ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

IMPLEMENTACIÓN DE UN SISTEMA PROTOTIPO DE ADMINISTRACIÓN DE INVENTARIO DE BIENES UTILIZANDO UNA APLICACIÓN MÓVIL CON SISTEMA OPERATIVO ANDROID Y TECNOLOGÍA NFC PARA LA ESCUELA POLITÉCNICA NACIONAL

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN "ELECTRÓNICA Y TELECOMUNICACIONES"

LAURA MARÍA VIZCAÍNO ARMAS

DIRECTOR: ING. TARQUINO FABIÁN SÁNCHEZ ALMEIDA, MBA.

Quito, agosto 2020

AVAL

Certifico que el presente trabajo fue desarrollado por Laura María Vizcaíno Armas, bajo mi supervisión.

ING. Tarquino Sánchez, MBA.
DIRECTOR DEL TRABAJO DE TITULACIÓN

DECLARACIÓN DE AUTORÍA

Yo, Laura María Vizcaíno Armas, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración dejo constancia de que la Escuela Politécnica Nacional podrá hacer uso del presente trabajo según los términos estipulados en la Ley, Reglamentos y Normas vigentes.

Laura María Vizcaíno Armas

DEDICATORIA

Dedico este trabajo a mis padres Nelson y Carmen, quienes nunca han dudado de mis capacidades, han sido mi guía y apoyo incondicional en la carrera universitaria y en la vida misma. Este es el resultado de su ayuda y de su confianza. Todo se lo debo a ustedes.

A mis hermanos Daniel y Carlos, quienes han sido un ejemplo a seguir y siempre me han brindado su soporte y valor para que culmine con éxito mi carrera.

Laura

AGRADECIMIENTO

Mi agradecimiento imperecedero a la Escuela Politécnica Nacional por abrirme sus puertas y permitirme formarme académicamente logrando con éxito culminar esta carrera, la que será mi carta de presentación en donde quiera que deba enfrentarme.

Un agradecimiento sincero al Ingeniero Tarquino Sánchez, tutor del presente trabajo; quien con su sabia experiencia supo guiarme hasta lograr terminar con este proyecto.

Mi agradecimiento formal y eterno a todos y cada de uno de los docentes de tan prestigiosa casona del saber, a las autoridades, personal administrativo y de servicio de esta.

Agradezco a mis compañeros y compañeras de estudio ya que con ellos supimos enfrentar momentos de alegría, sufrir algunos derrotas y lograr muchos éxitos.

Mi gratitud a mis amigas inseparables de la residencia con quienes vivimos grandes y dulces momentos y fueron un apoyo incondicional en mi vida universitaria; para ellas que estuvieron junto a mi brindándome su apoyo y su valiosa amistad; a ellas les digo que las llevaré siempre en lo más profundo de mi ser.

Mi agradecimiento a Washo quien ha sido un gran apoyo y compañero de vida.

A mis hermanos y mis sobrinos quienes siempre alegran mis días con sus sonrisas.

De manera especial agradezco a mi padres ya que nada de lo logrado hubiese sido posible sin su ayuda y guía incondicional, quienes me enseñaron a no rendirme y siempre luchar por mis metas.

Laura

ÍNDICE DE CONTENIDO

AVAL I
DECLARACIÓN DE AUTORÍA II
DEDICATORIA III
AGRADECIMIENTO IV
ÍNDICE DE CONTENIDO V
RESUMEN IX
ABSTRACT X
1. INTRODUCCIÓN 1
1.1 OBJETIVOS 1
1.2 ALCANCE 2
1.3 MARCO TEÓRICO 2
1.3.1 NEAR FIELD COMMUNICATION (NFC) 2
1.3.1.1 Definición 2
1.3.1.2 Estandarización de la tecnología NFC 3
1.3.1.3 Foro NFC 3
1.3.1.4 Modos de Operación 4
1.3.1.4.1 Modo Lector/Escritor 4
1.3.1.4.2 Modo Punto a Punto 4
1.3.1.4.3 Modo Emulación de Tarjeta 4
1.3.1.4.4 Modo de Carga Inalámbrica 4
1.3.1.5 Modos de Comunicación [12] 4
1.3.1.5.1 Modo Activo 4
1.3.1.5.2 Modo Pasivo 5
1.3.1.6 Etiquetas NFC [13] 5
1.3.1.6.1 Etiqueta Tipo 1 5
1.3.1.6.2 Etiqueta Tipo 2 5
1.3.1.6.3 Etiqueta Tipo 3 5
1.3.1.6.4 Etiqueta Tipo 4 6
1.3.1.6.5 Etiqueta Tipo 5 6
1.3.1.7 Formato NDEF (NFC Data Exchange Format) 6

1.3.1.8 Ventajas de NFC 7
1.3.1.9 Limitaciones de NFC 7
1.3.1.10 Aplicaciones Comunes [18] 8
1.3.2 ANDROID 8
1.3.2.1 Arquitectura de la plataforma [21] 8
1.3.2.1.1 Kernel de Linux 10
1.3.2.1.2 Capa de Abstracción de Hardware 10
1.3.2.1.3 Tiempo de Ejecución 10
1.3.2.1.4 Bibliotecas C/C++ Nativas 10
1.3.2.1.5 Marco de Trabajo de la API de Java 11
1.3.2.1.6 Aplicaciones del Sistema 11
1.3.2.2 Android Studio 11
1.3.2.2.1 Estructura del proyecto 12
1.3.2.2.2 Interfaz de Usuario 13
1.3.2.2.3 Sistema de compilación de Gradle 15
1.3.2.2.4 Administración de dependencias 16
1.3.2.2.5 Componentes de la aplicación [24] 16
1.3.2.2.5 Ciclo de Vida de una Actividad [25] 17
1.3.3 TIPOS DE APLICACIONES MÓVILES [26] 18
1.3.4 BASE DE DATOS 18
1.3.4.1 Gestor de base de datos 19
1.3.4.2 Modelos de bases de datos [28] 19
1.3.4.2.1 Base de datos jerárquicas 19
1.3.4.2.2 Bases de datos en red. 19
1.3.4.2.3 Bases de datos relacionales 19
1.3.4.2.3 Bases de datos orientadas a objetos 19
1.3.4.3 Structure Query Language 20
1.3.5 APLICACIÓN WEB 20
1.3.6 NGIN-X 21
1.3.7 API 21
1.3.7.1 Web Services 21
1.3.7.2 Métodos HTTP [35] 22
1.3.7.3 Formato de Datos 22
1.3.8 SISTEMA DE CONTROL DE VERSIONES 22

1.3.8.1 GIT 23
2. METODOLOGÍA 23
2.1 REQUERIMIENTOS DEL SISTEMA 23
2.2 ARQUITECTURA DEL PROTOTIPO 24
2.3 FUNCIONAMIENTO DEL PROTOTIPO 25
2.3.1 ETIQUETA NFC 26
2.3.2 APLICACIÓN WEB 27
2.3.3 APLICACIÓN MÓVIL 27
2.3.4 BASE DE DATOS 28
2.3.5 SERVIDOR WEB 28
2.3.6 GITHUB 28
2.4 ESTRUCTURA DE LA BASE DE DATOS 29
2.5 DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN WEB 30
2.5.1 INTERFAZ DE INICIO 31
2.5.2 INTERFAZ HOME 33
2.5.3 INTERFAZ RESUMEN 33
2.5.4 INTERFAZ USUARIOS – ADMINISTRAR GRUPOS 34
2.5.5 INTERFAZ USUARIOS – ADMINISTRAR USUARIOS 37
2.5.6 INTERFAZ BIENES - ADMINISTRAR BIENES 39
2.5.7 INTERFAZ BIENES - AÑADIR BIEN 39
2.5.8 PERFIL 41
2.6 DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN MÓVIL 45
2.6.1 ACTIVIDAD LOGIN 48
2.6.2 ACTIVIDAD MENÚ 51
2.6.3 ACTIVIDAD ESCRIBIR ETIQUETA 53
2.6.4 ACTIVIDAD LEER ETIQUETA 60
2.6.5 ACTIVIDAD INVENTARIO 65
2.6.6 ACTIVIDAD LECTURA 66
3. PRUEBAS Y RESULTADOS 68
3.1 COMPONENTES DE HARDWARE Y SOFTWARE IMPLEMENTADOS 68
3.2 ESCENARIO DE PRUEBAS 70
3.3 PRUEBAS DE FUNCIONALIDAD 70
3.3.1 PRUEBAS DE APLICACIÓN WEB 70
3.3.1.1 Añadir un Nuevo Grupo 70

3.3.1.2 Añadir un Nuevo Usuario 72
3.3.1.3 Añadir Nuevo Bien 74
3.3.1.4 Editar Cuenta 75
3.3.2 PRUEBAS DE APLICACIÓN MÓVIL 77
3.3.2.1 Login 77
3.3.2.2 Escribir Etiqueta 80
3.3.2.3 Leer Etiqueta 83
3.3.2.4 Error de Conexión 84
3.4 TIEMPOS DE OCURRENCIAS DE EVENTOS 86
3.5 ANÁLISIS DE RESULTADOS 87
3.5.1 LOGIN 87
3.5.2 REGISTRO EN BASE DE DATOS 87
3.5.3 VISUALIZACIÓN DE INFORMACIÓN 88
3.5.4 LECTURA Y ESCRITURA DE LAS ETIQUETAS 88
3.5.5 ERRORES DE CONEXIÓN 88
4. CONCLUSIONES Y RECOMENDACIONES 89
4.1 CONCLUSIONES 89
4.2 RECOMENDACIONES 91
5. REFERENCIAS BIBLIOGRÁFICAS 92
ANEXOS 95

RESUMEN

El presente proyecto muestra un prototipo de administración de inventario de bienes utilizando una aplicación móvil, una aplicación web y tecnología Near Field Communication (NFC). El sistema está compuesto por: servidor web, base de datos, aplicación desarrollada para Android y aplicación web. Se utilizan etiquetas NFC como un identificador de cada bien, las que están asociadas a un código que se encuentra en la base de datos juntamente con las características del bien.

En el capítulo uno se presentan los objetivos, alcance y conceptos generales acerca de la tecnología NFC, Sistema Operativo Android, Base de Datos, lenguajes de programación y softwares utilizados.

En el capítulo dos se describe la arquitectura y funcionamiento del sistema, como se desarrollaron las aplicaciones y su comunicación con la base de datos.

En el capítulo tres se presentan las pruebas realizadas con las etiquetas NFC y el correcto funcionamiento de estas con la aplicación móvil.

Finalmente, en el capítulo cuatro se describe las experiencias obtenidas a lo largo del trabajo, presentado a través de conclusiones y recomendaciones.

PALABRAS CLAVE: NFC, Aplicación Web, Aplicación Móvil, Inventario.

ABSTRACT

This project shows a prototype of inventory management of goods using a mobile application, a web application and Near Field Communication (NFC) technology. The system is made up of: web server, database, application developed for Android and web application. NFC tags are used as an identifier for each asset, those that are associated with a code found in the database identified with the characteristics of the asset.

Chapter one presents the objectives, scope and general concepts about NFC technology, Android Operating System, Database, programming languages and software used.

Chapter two describes the architecture and operation of the system, how the applications were developed and how they communicate with the database.

Chapter three presents the tests carried out with the NFC tags and the correct operation of these with the mobile application.

Finally, chapter four describes the experiences obtained throughout the work, presented through conclusions and recommendations.

KEYWORDS: NFC, Web Application, Mobile Application, Inventory.

1. INTRODUCCIÓN

El control y organización de inventarios es una necesidad que toda entidad privada o pública necesita automatizar para los procesos de registro, actualización y movimiento de los bienes muebles y la infraestructura tecnológica de forma rápida y sencilla. El objetivo principal de tener un adecuado registro de inventarios es contar con información suficiente y útil para: minimizar pérdidas, mantener un nivel de inventario óptimo y comenzar a utilizar la tecnología con la consecuente disminución de gastos operativos.[1]

El problema por resolver con el control y registro de los bienes es identificar la ubicación de estos, sobre todo cuando por algún motivo cambian de lugar a otro distinto sin ninguna notificación por parte del custodio, lo que trae como consecuencia el desconocimiento de la ubicación del bien cuando se lo requiera; o en otros casos no se conoce donde se encuentran cuando se desea cambiar el custodio o realizar un traspaso administrativo. Por otro lado, se tiene un deficiente sistema de registro; pues en muchos casos se realiza una identificación física por inspección y posterior registro, y en otros casos, no hay una constatación física, esto sin duda generaría dificultades relacionados con la pérdida de bienes, falta de registro, cambio de información y desconocimiento de la ubicación; lo que a su vez genera dificultades administrativas y financieras.

El presente proyecto pretende brindar como solución al problema mencionado un prototipo, en el cual el custodio de los bienes pueda visibilizar la información de cada bien y añadir un nuevo bien a su cargo en cualquier momento mediante el uso de una aplicación móvil bajo sistema operativo Android [2] ,mediante etiquetas NFC ubicadas en cada bien. También se podrá visualizar en una aplicación web la información de los bienes para gestionar la información en cuanto a: actualización de registros, cambio de ubicación, mantenimiento, etc. refrescando la información para el Responsable Administrativo de la Oficina de Control de Bienes de la Institución.

1.1 OBJETIVOS

El objetivo general de este Proyecto Técnico es: Implementar un sistema prototipo de administración de inventario de bienes utilizando una aplicación móvil con sistema operativo Android y tecnología NFC para la Escuela Politécnica Nacional.

Los objetivos específicos del Proyecto Técnico son:

- Implementar la base de datos mediante el gestor de base de datos.

- Implementar el software para la aplicación web y desarrollar la misma.
- Desarrollar la aplicación móvil que leerá y escribirá las etiquetas NFC y permitirá la visualización de características de los bienes.
- Establecer la comunicación entre la aplicación web con la base de datos y la aplicación móvil con la base de datos.
- Implementar en una muestra de 10 bienes las etiquetas NFC y comprobar su funcionamiento con la lectura de esta en el móvil y el sistema en conjunto.

1.2 ALCANCE

Para el desarrollo del presente trabajo de titulación se investigará las características y funcionalidades de la comunicación NFC, se utilizará una base de datos donde se registrará la información de cada bien asociado a cada identificador de las etiquetas NFC juntamente con una aplicación que correrá en el Sistema Operativo Android la cual permitirá la lectura y escritura de la etiqueta y la visualización de características de los bienes almacenados, verificar sus estados e información de cada uno. La aplicación dará soporte únicamente a celulares que tengan implementada la tecnología NFC ya que esta se utilizará para la lectura de la etiqueta NFC.

Se manejará una aplicación web desde donde se administrará la base de datos.

Se colocará una etiqueta NFC en cada bien dentro de una muestra de 10 bienes del Laboratorio de Informática de la Facultad de Ingeniería Eléctrica para asociar su identificador con el bien y realizar diversas pruebas.

1.3 MARCO TEÓRICO

La presente sección tiene como objetivo presentar conceptos básicos y la información teórica para entender la problemática y la solución propuesta en este trabajo.

1.3.1 NEAR FIELD COMMUNICATION (NFC)

1.3.1.1 Definición

Near Field Communication (NFC) es un estándar internacional que define una tecnología inalámbrica de corto alcance que puede hacer transmisión de datos bidireccionalmente, lo que quiere decir que la transmisión y recepción de datos se hace al mismo tiempo. NFC se basa en la tecnología inalámbrica RFID (Radio Frecuency Identification). [3]

Sony empezó el desarrollo de la tecnología NFC en el año 2002, orientada en el uso de teléfonos móviles para el intercambio de datos. Esta tecnología reside en la unificación de dispositivos móviles u otros dispositivos con identificación por radiofrecuencia, que brinda una comunicación entre dispositivos electrónicos. Generalmente esta comunicación se da a una distancia de 5 a 20 cm con una frecuencia de 13.56 MHz y con una tasa de transferencia de 424 Kb/s. [2]

NFC es compatible con cientos de millones de tarjetas y lectores sin contacto ya implementados en todo el mundo.

1.3.1.2 Estandarización de la tecnología NFC

ISO/IEC (International Organization for Standardization / International Electro-technical Commission), ETSI (European Telecommunications Standards Institute) y ECMA (European Association for Standardizing Information and Communication Systems) han estandarizado la tecnología NFC. El estándar ISO/IEC 18092/ECMA – 340 define los modos de comunicación utilizando dispositivos inductivos acoplados que funcionan a la frecuencia central de 13.56 MHz, también define los modos de comunicación activo y pasivo, especifica esquemas de modulación, codificaciones, velocidades de transferencia y formato de trama de la interfaz RF, así como esquemas de inicialización y condiciones requeridas para el control de colisión de datos durante la inicialización. Además, ISO/IEC 18092:2013 define un protocolo de transporte que incluye la activación del protocolo y los métodos de intercambio de datos.[4] [5]

1.3.1.3 Foro NFC

El foro NFC se lanzó en al año 2004 como una asociación sin fines de lucro, iniciado por compañías de comunicaciones móviles. La misión del Foro es seguir con el desarrollo de la tecnología NFC mediante el impulso de especificaciones garantizando la interoperabilidad entre dispositivos y servicios, y enseñar a las empresas, proveedores de servicios y desarrolladores sobre cómo utilizar la tecnología NFC. [6]

Después de solo 18 meses de haber sido fundado el Foro describió formalmente la arquitectura de la tecnología NFC. Se han publicado 26 especificaciones hasta la fecha. Estas especificaciones proveen una “hoja de ruta” que permite a las partes interesadas crear nuevos productos orientados al consumidor.[7]

1.3.1.4 Modos de Operación

Entre los estándares que desarrolló el Fórum NFC se encuentran los tres modos de operación de NFC basados en los posibles casos de interacción de dispositivos NFC. Estos modos de operación están basados en los estándares ISO/IEC 18092 NFC IP-1 e ISO/IEC 14443, a continuación, se detallan: [8] [9]

1.3.1.4.1 Modo Lector/Escritor

El dispositivo NFC es capaz de leer etiquetas definidas en el Fórum NFC, puede ser una etiqueta colocada en un póster inteligente NFC. Este modo en la interfaz RF cumple con los esquemas ISO 14443 y FeliCa. [10]

1.3.1.4.2 Modo Punto a Punto

Se puede intercambiar datos entre dos dispositivos NFC. Un ejemplo, compartir parámetros de configuración de una conexión WiFi o intercambiar datos como tarjetas de negocios o fotos digitales. Está basado en el estándar ISO/IEC 18092.

1.3.1.4.3 Modo Emulación de Tarjeta

El dispositivo NFC se muestra para un lector externo como una etiqueta NFC tradicional. Este modo permite pagos sin contacto y emisión de boletos mediante dispositivos NFC sin cambiar la infraestructura existente.

1.3.1.4.4 Modo de Carga Inalámbrica

Dispositivos pequeños IoT como auriculares Bluetooth, rastreador de ejercicios o relojes inteligentes se pueden cargar con transferencia sin contacto de hasta 1 W de potencia. [11]

1.3.1.5 Modos de Comunicación [12]

NFC realiza intercambio de datos entre dos dispositivos cercanos, los que se conocen con el nombre de *Iniciator* (de donde se origina la comunicación) y *Target* (el receptor). La comunicación se basa en el principio de inducción electromagnética, dos circuitos inductivos comparten energía y se trasmite información a poca distancia. NFC puede trabajar con dos modos de comunicación: activa y pasiva.

1.3.1.5.1 Modo Activo

En este modo un campo de radiofrecuencia es generado por cada dispositivo para realizar la comunicación. El Iniciator comienza la comunicación a lo que el target responde utilizando una modulación autogenerada por su propio campo RF. Para este modo es necesario que los dos dispositivos tengan su propia fuente de alimentación.

1.3.1.5.2 Modo Pasivo

En este modo solo el Iniciator genera el campo de radiofrecuencia a lo que el target mediante acoplamiento inductivo toma energía para generar la modulación. Este modo es importante para dispositivos que funcionan que utilizan batería como teléfonos inteligentes o Personal Digital Assistant, los que necesitan ahorrar energía.

1.3.1.6 Etiquetas NFC [13]

Las especificaciones para las etiquetas NFC Tipo 1/2/3/4 se definen por el Fórum NFC y brindan la información técnica para implementar el modo lector/escritor y la funcionalidad de control asociada del dispositivo NFC para interactuar con las etiquetas. La especificación técnica TNEP 1.0 admite el intercambio bidireccional de mensajes NDEF basado en el protocolo de comunicación usado por las etiquetas de tipo 2, 3, 4 y 5. El propósito de estas especificaciones es definir como los mensajes NDEF son leídos desde una etiqueta NFC y escritos en una etiqueta NFC.

1.3.1.6.1 Etiqueta Tipo 1

Define como un dispositivo habilitado para NFC en modo Lector/Escritor, detecta, lee y escribe un mensaje NDEF en una etiqueta NFC Fórum Tipo 1. La comunicación con este tipo de etiqueta se basa en la tecnología NFC-A. Estas etiquetas pueden ser leídas y ser reescritas, aunque se permite que funcionen solamente en modo de lectura. La memoria disponible es de 96 Bytes y expandible a 2 kB; la velocidad de comunicación es de 106 Kbps.

1.3.1.6.2 Etiqueta Tipo 2

Define como un dispositivo habilitado para NFC en modo Lector/Escritor, detecta, lee y escribe un mensaje NDEF en una etiqueta NFC Fórum Tipo 2. La comunicación con este tipo de etiqueta se basa en la tecnología NFC-A. Poseen las mismas descripciones de las etiquetas tipo 1 pero su capacidad de memoria es mayor. La memoria disponible es de 48 Bytes y expandible a 2 kB; la velocidad de comunicación es de 106 Kbps.

1.3.1.6.3 Etiqueta Tipo 3

Define como un dispositivo habilitado para NFC en modo Lector/Escritor, detecta, lee y escribe un mensaje NDEF en una etiqueta NFC Fórum Tipo 3. La comunicación con este tipo de etiqueta se basa en la tecnología NFC-F que es compatible con el estándar japonés (JIS) X 6319-4. Estas etiquetas son configuradas previamente para que sean de lectura, para ser reescritas o solamente de lectura. La capacidad de memoria es variable, pero

teóricamente el límite de memoria es 1 MB por servicio; la velocidad de comunicación es de 212 Kbps o 424 Kbps.

1.3.1.6.4 Etiqueta Tipo 4

Define como un dispositivo habilitado para NFC en modo Lector/Escritor, detecta, lee y escribe un mensaje NDEF en una etiqueta NFC Fórum Tipo 4. La comunicación con este tipo de etiqueta se basa en el protocolo de intercambio da datos ISO (ISO-DEP) que es compatible con el estándar ISO/IEC 14443. Este protocolo se basa en tecnología NFC-A o NFC-B. Posee las mismas características de las etiquetas tipo 3 pero tolera una mayor cantidad de almacenamiento. La memoria disponible es variable, de hasta 32 kB por servicio; la velocidad de comunicación es de hasta 424 Kbps. [14] [15]

1.3.1.6.5 Etiqueta Tipo 5

Define como un dispositivo habilitado para NFC en modo Lector/Escritor, detecta, lee y escribe un mensaje NDEF en una etiqueta NFC Fórum Tipo 5. La comunicación con este tipo de etiqueta se basa en la tecnología NFC-V.

1.3.1.7 Formato NDEF (NFC Data Exchange Format)

Como se vio anteriormente, el Foro NFC es responsable del desarrollo de especificaciones NFC basadas en estándares para permitir la interoperabilidad entre diferentes dispositivos y etiquetas. Una de las especificaciones creadas por el Foro NFC es el Formato de Intercambio de Datos NFC (NDEF). La especificación NDEF define el formato de encapsulación de mensajes para intercambiar información entre dos dispositivos habilitados para NFC o dispositivos con la etiqueta NFC. NDEF es un formato de mensaje binario ligero que se puede utilizar para encapsular varias cargas útiles personalizadas. El uso de esta capa de abstracción permite crear el contenido de la etiqueta estándar compatible con todos los tipos de etiquetas del Foro NFC, ignorando el hecho de que cada tipo de etiqueta tiene diferentes configuraciones de hardware y diseños de memoria. [16]

Un mensaje NDEF está compuesto por uno o más registros NDEF. Cada registro NDEF está compuesto por la carga útil (Payload) y un encabezado (Header). En el encabezado se almacenan; la longitud de los datos, su tipo y su identificador. [16] En la Figura 1.1 se puede apreciar esta estructura.

Figura 1.1. Estructura de un mensaje NDEF [16]

1.3.1.8 Ventajas de NFC

Debido a las características de la tecnología NFC existen muchas posibilidades de aplicación y garantizan algunas ventajas, como:

- Mayor seguridad, la transferencia de datos al ser de corto alcance dificulta que estos puedan ser interceptados por terceros.
- Comunicación ágil y rápida, no es necesario realizar previas configuraciones para la comunicación, basta con acercar los dispositivos.
- Ofrece una variedad de aplicaciones para diferentes áreas debido a sus modos de operación.
- Brinda una interacción intuitiva con el usuario final, ya que este no debe tener conocimientos técnicos acerca de la tecnología.
- Es compatible con tarjetas inteligentes e infraestructura RFID.
- La integración de la tecnología NFC en teléfonos inteligentes deriva en hacer uso también de las bondades que posee el móvil juntamente con NFC para desarrollar una amplia gama de aplicaciones.

1.3.1.9 Limitaciones de NFC

Un problema difícil de detectar son las escuchas anónimas, al ser una comunicación que se realiza mediante señales de radiofrecuencia un atacante puede utilizar una antena para capturar las señales transmitidas.

RFID jamming, en este escenario el atacante no escucha, sino que modifica los datos que se transmiten vía el interfaz NFC. De esta manera también sólo puede interferir para que el receptor no entienda los datos que envía el otro dispositivo.

Modificar datos, aquí el atacante envía datos manipulados. Codificaciones como Manchester con ASK facilitan este problema, por lo que es mejor utilizar una función criptográfica hash.

Un dispositivo activo solamente puede comunicarse con un dispositivo pasivo al mismo tiempo, la comunicación broadcast no es permitida.[17]

1.3.1.10 Aplicaciones Comunes [18]

Pagos móviles, pasando el teléfono por un sensor es rápido hacer una transacción, para lo que se requiere un terminal NFC que contenga los datos bancarios de forma segura.

Tarjetas de acceso, en universidades o empresas se utiliza NFC para acceder a las instalaciones además de ser un control de asistencia.

Libros, en las bibliotecas etiquetas NFC pueden incluirse en las páginas de libros para descargar archivos que complementen lo que se está investigando. También puede utilizarse en lugares donde no existen bibliotecas públicas colocando etiquetas NFC en lugares públicos para abrir un enlace que permita la descarga de libros que no se encuentran bajo leyes de propiedad intelectual.

En Hospitales, el médico utilizando su móvil puede acceder a la información de cada paciente utilizando tecnología NFC.

1.3.2 ANDROID

Android es un Sistema Operativo construido en el kernel abierto de Linux. Para optimizar recursos de capacidad de memoria y hardware en un dispositivo móvil, Android utiliza una máquina virtual personalizada. La plataforma Android evoluciona en cuanto los desarrolladores trabajen en la creación de nuevas aplicaciones por ejemplo agregando nuevas tecnologías como NFC ya que es de código abierto. [19]

Android Inc. fue quien inició el desarrollo de Android, luego adquirido por Google y actualmente por Open Handset Alliance. Google ha divulgado la mayoría de código fuente bajo la licencia del software Apache que es de código abierto.[20]

1.3.2.1 Arquitectura de la plataforma [21]

En la Figura 1.2 se muestran los componentes principales de Android.

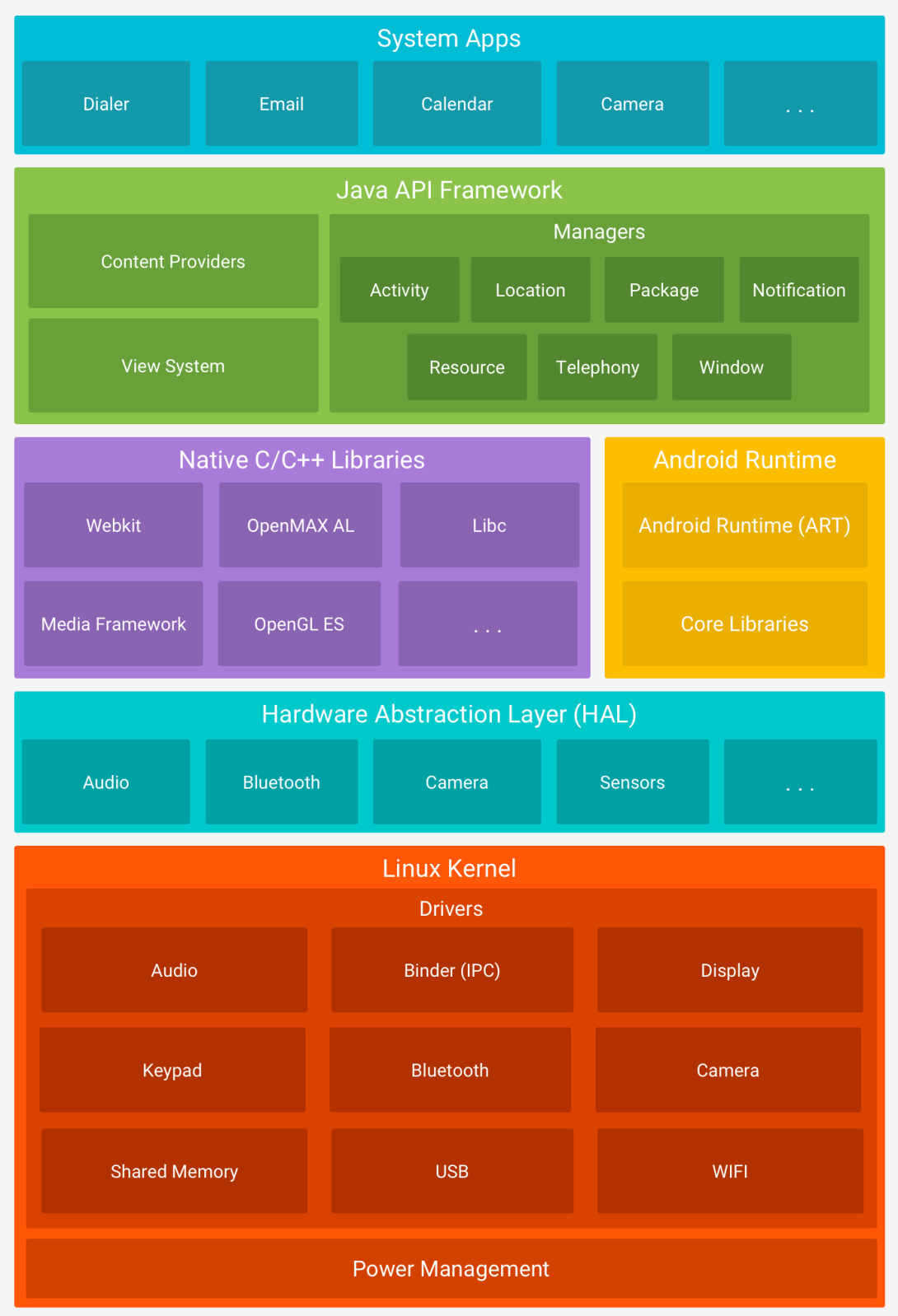


Figura 1.2. Arquitectura de la plataforma Android [21]

1.3.2.1.1 Kernel de Linux

El kernel de Linux es la base de la plataforma Android. Así, funcionalidades subyacentes del tiempo de ejecución como la generación de subprocesos y la administración de memoria de bajo nivel se basan en el kernel de Linux. Actúa como una capa de abstracción entre el hardware y la pila de software.

El uso del kernel de Linux permite que Android fructifique funciones de seguridad, también al ser un kernel conocido permite facilidad en desarrollo de dispositivos para los fabricantes.

1.3.2.1.2 Capa de Abstracción de Hardware

La capa de abstracción de hardware (HAL) ofrece interfaces estándares que aprovechan las capacidades de hardware del dispositivo al marco de trabajo de la API de Java. La HAL se compone de algunos módulos de biblioteca los que realizan una interfaz para un tipo específico de hardware, como el módulo de la cámara o de Bluetooth. Cuando el marco de trabajo de una API realiza una petición para acceder al hardware del dispositivo, el sistema Android carga el módulo de biblioteca correspondiente.

1.3.2.1.3 Tiempo de Ejecución

En dispositivos que tienen Android 5.0 (nivel de API 21) o versiones posteriores, cada aplicación ejecuta sus propios procesos y solicitudes del tiempo de ejecución de Android (ART). El ART ejecuta varias máquinas virtuales en dispositivos de memoria baja ejecutando archivos DEX, un formato de código de bytes diseñado especialmente para Android y perfeccionado para ocupar un espacio de memoria pequeño. Antes de Android 5.0, Dalvik funcionaba como el entorno de ejecución del sistema operativo.

Android incluye un conjunto de bibliotecas de entorno de ejecución centrales que tienen la mayoría de las funcionalidades del lenguaje de programación Java; se incluyen algunas funciones del lenguaje Java 8, que usa el marco de trabajo de la API de Java.

1.3.2.1.4 Bibliotecas C/C++ Nativas

Componentes de Android, como ART y HAL, se basan en código nativo que requiere bibliotecas nativas escritas en C y C++. La plataforma Android proporciona API del marco de trabajo de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas a las aplicaciones.

.

1.3.2.1.5 Marco de Trabajo de la API de Java

Todas las funciones del Sistema Operativo Android están disponibles mediante APIs escritas en el lenguaje Java. Las APIs son los fundamentos necesarios para crear aplicaciones de Android, estas simplifican la reutilización de componentes y servicios del sistema, como los siguientes:

- Un sistema de vista extensivo para la compilación de las interfaces de usuario donde se incluyen listas, cuadrículas, cuadros de texto, botones y un navegador web integrable.
- Un administrador de recursos que ofrece acceso a recursos sin código, como strings localizadas, gráficos y archivos de diseño.
- Un administrador de notificaciones que permite que todas las aplicaciones manifiesten alertas personalizadas en la barra de estado.
- Un administrador de actividad que gestiona el ciclo de vida de las aplicaciones.
- Proveedores de contenido que permiten que las aplicaciones accedan a datos desde otras aplicaciones, como la aplicación de Contactos.

1.3.2.1.6 Aplicaciones del Sistema

Android incluye un conjunto de aplicaciones centrales como correo electrónico, mensajería SMS, calendarios, navegación en Internet y contactos, entre otros. Estas aplicaciones no tienen un estado especial entre las aplicaciones que el usuario elije instalar; entonces, una aplicación externa también puede ser el navegador web, el sistema de mensajería SMS o el teclado predeterminado del usuario (considerando excepciones, como la aplicación Ajustes del Sistema). Las aplicaciones centrales del sistema brindan capacidades importantes a las cuales los desarrolladores pueden tener acceso desde sus propias aplicaciones.

1.3.2.2 Android Studio

Android Studio es el Entorno de Desarrollo Integrado (IDE) oficial para el desarrollo de aplicaciones para Android. Además del editor de código Android Studio ofrece más funciones que aumentan la productividad en el desarrollo de aplicaciones, como:

- Sistema de compilación flexible basado en Gradle.
- Emulador vertiginoso y lleno de funciones

- Mismo ambiente para el desarrollo de aplicaciones para todos los dispositivos que tengan sistema operativo Android.
- Facilidad en cuanto a inserción de código y recursos sin reiniciar la ejecución de la aplicación.
- Integración con GitHub y plantillas de código para realizar la compilación de aplicaciones comunes.
- Herramientas para identificar problemas de rendimiento, usabilidad y compatibilidad de la versión.
- Compatible con C++ y NDK. [22]

1.3.2.2.1 Estructura del proyecto

En la figura 1.3 se muestra cómo se organiza un proyecto en Android Studio. Cada proyecto de Android Studio contiene módulos de código fuente y archivos de recursos. Los siguientes son tipos de módulos:

- Módulos de aplicaciones para Android
- Módulos de biblioteca
- Módulos de Google App Engine

Android Studio muestra los archivos del proyecto como se mira en la Figura 1.3. Los archivos de compilación se encuentran en la parte superior de Gradle Scripts y cada módulo de aplicación contiene:

- Manifests: donde está el archivo AndroidManifest.xml.
- Java, donde están los códigos fuente Java.
- Res, donde están los archivos sin código, como diseños XML, imágenes de mapa de bits, básicamente el diseño visual de la aplicación. [23]

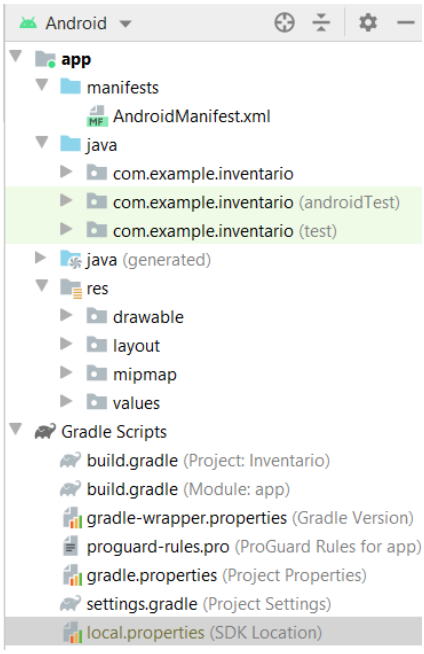


Figura 1.3. Estructura del proyecto dentro de Android Studio. Fuente: Android Studio

1.3.2.2.2 Interfaz de Usuario

Como se puede ver en la Figura 1.4 la ventana principal de Android Studio consta de varias áreas que se explican a continuación.

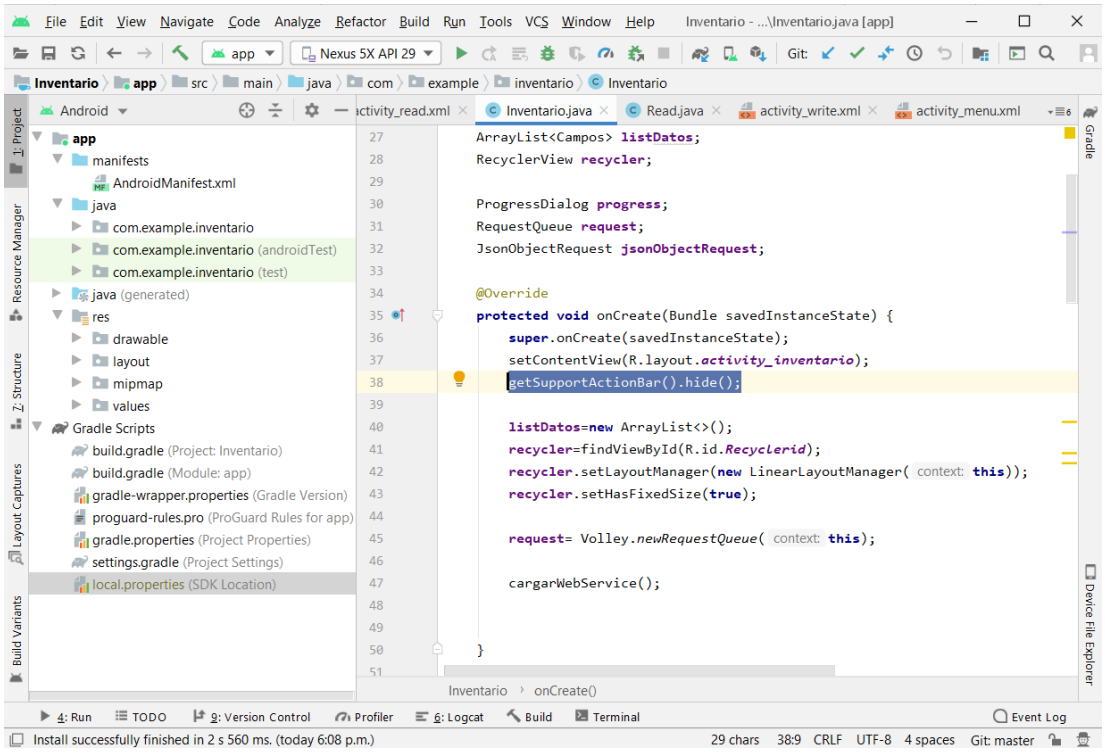


Figura 1.4. Interfaz de Usuario de Android Studio. Fuente: Android Studio

La Figura 1.5 representa la barra de herramientas que permite ejecutar algunas acciones, como ejecutar la aplicación, depurar la aplicación sincronizar el proyecto con los archivos de Gradle, también iniciar herramientas de Android.

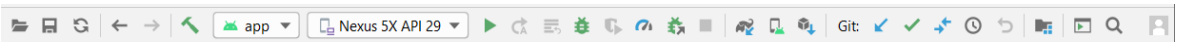


Figura 1.5. Barra de Herramientas. Fuente: Android Studio

En la Figura 1.6 se aprecia la barra de navegación que permite examinar el proyecto, así como abrir archivos para editar.

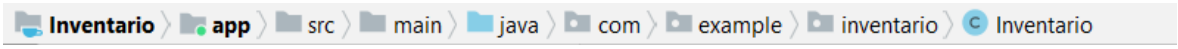
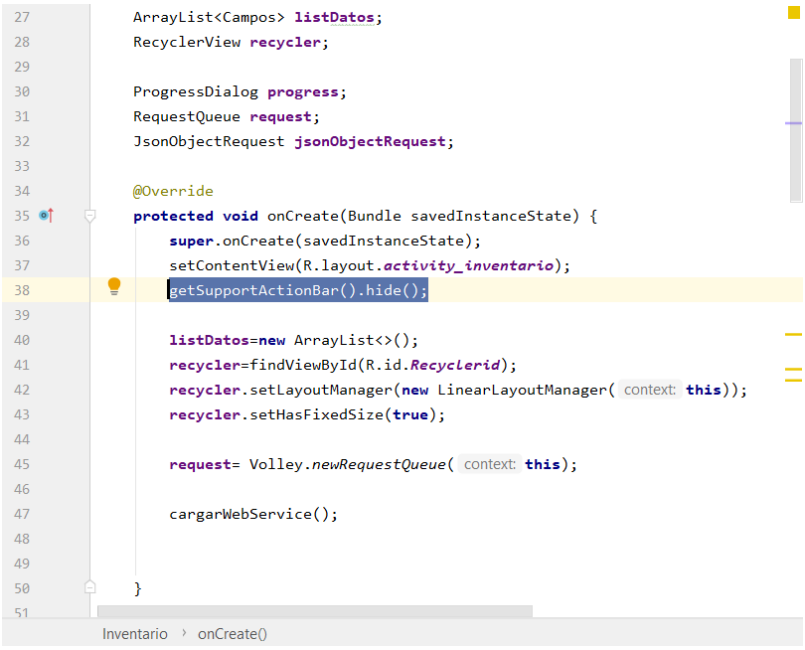


Figura 1.6. Barra de navegación. Fuente: Android Studio

En la Figura 1.7 se mira el editor, esta es el área donde se puede escribir y modificar el código.



```
27    ArrayList<Campos> listDatos;
28    RecyclerView recycler;
29
30    ProgressDialog progress;
31    RequestQueue request;
32    JsonObjectRequest jsonObjectRequest;
33
34    @Override
35    protected void onCreate(Bundle savedInstanceState) {
36        super.onCreate(savedInstanceState);
37        setContentView(R.layout.activity_inventario);
38        getSupportActionBar().hide();
39
40        listDatos=new ArrayList<>();
41        recycler=findViewById(R.id.Recyclerid);
42        recycler.setLayoutManager(new LinearLayoutManager( context: this));
43        recycler.setHasFixedSize(true);
44
45        request= Volley.newRequestQueue( context: this);
46
47        cargarWebService();
48
49
50    }
51
Inventario › onCreate()
```

Figura 1.7. Editor de texto. Fuente: Android Studio

Las ventanas de herramientas ofrecen acceso a acciones determinadas como administración de proyectos, búsqueda y control de versiones, entre otras. Un ejemplo se puede observar en la Figura 1.8.

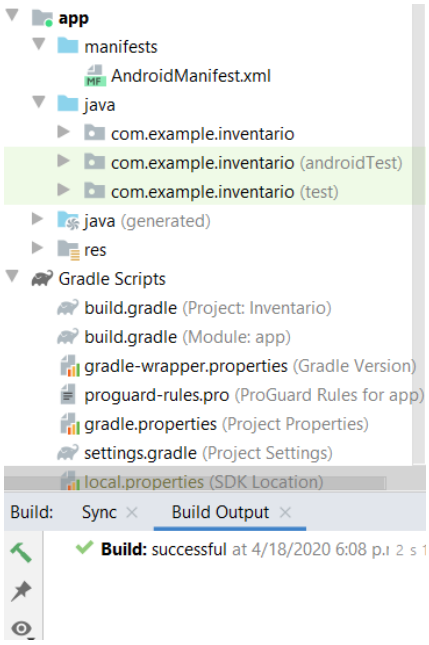


Figura 1.8. Ventanas de herramientas. Fuente: Android Studio

En la Figura 1.9 se observa la barra de estado, la cual muestra el estado del proyecto, además de advertencias y mensajes.

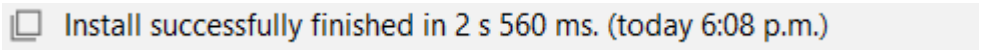


Figura 1.9. Barra de estado. Fuente: Android Studio

1.3.2.2.3 Sistema de compilación de Gradle

Para el sistema de compilación Android Studio usa Gradle. Este sistema de compilación utiliza una herramienta desde el menú y ejecuta esta herramienta independientemente del código fuente. Se puede utilizar funciones del sistema para:

- Personalizar, configurar y extender el proceso de compilación.
- Crear varios APK para la aplicación.
- Volver a usar códigos y recursos en conjuntos de archivos fuente.

Los archivos de compilación de Android Studio se denominan build.gradle. Son archivos de texto sin formato que usan la sintaxis Groovy para configurar la compilación con elementos que proporciona el complemento de Android para Gradle. Los archivos de compilación son generados automáticamente cuando se importa un proyecto existente. [22]

1.3.2.2.4 Administración de dependencias

En el archivo build.gradle se encuentran las dependencias del proyecto. Gradle busca las dependencias y hace que estén disponibles en la compilación. [22]

1.3.2.2.5 Componentes de la aplicación [24]

Existen componentes importantes para el desarrollo de una aplicación en Android Studio. A continuación, se describen:

- ***Actividades*:** Representa el punto de entrada de interacción con el usuario. Las actividades representan una pantalla individual y utilizan un layout que define su aspecto. Una aplicación puede estar formada por varias actividades pero que son independientes entre sí.
- ***Servicios:*** Permite mantener la ejecución de una aplicación en segundo plano.
- ***Receptores de emisiones:*** Reciben y reaccionan a los anuncios de difusión. Las aplicaciones reciben esas transmisiones a medida que se anuncian y, a veces, las aplicaciones deben reaccionar en consecuencia. La mayoría de las transmisiones se originan en el sistema operativo, como un cambio de zona horaria, anuncio de batería baja o pantalla apagada. Las transmisiones no tienen interfaces de usuario; sin embargo, pueden crear una notificación en la barra de estado de Android.
- ***Proveedores de contenido:*** Pone a disposición de otras plataformas los datos de una aplicación según sea necesario. También, puede almacenar y recuperar datos, y ponerlos a disposición de otras aplicaciones. Al usar proveedores de contenido, la aplicación puede compartir datos con otros, y otras aplicaciones pueden incluso modificarla si su aplicación permite dichos cambios.

Además, se usan Intents, un archivo Manifest y recursos de aplicaciones:

- ***Intents:*** Las actividades, servicios y receptores de emisiones se activan por intents. Para actividades y servicios, los intents definen las acciones a realizar. Por ejemplo, un servicio puede hacer una solicitud para abrir una página web.
- ***Manifest:*** Antes de que el sistema operativo Android inicie un componente, el sistema debe saber que el componente de la aplicación existe al leer el archivo manifiest de la aplicación, AndroidManifest.xml. Este archivo incluye información sobre todos los componentes de la aplicación correspondiente. También ofrece otras funciones como; identificar los permisos de usuario que requiere la aplicación,

como el acceso a Internet, declarar niveles mínimos de API requeridos por la aplicación, declarar las características de hardware y software utilizadas por la aplicación, como la cámara y los servicios Bluetooth, NFC.

- ***Recurso de aplicación:*** Una aplicación de Android se compone de varios recursos que están separados del código fuente, como imágenes, archivos de audio y todo lo relacionado con la presentación visual de la aplicación. Para una aplicación de Android, se debe definir animaciones, menús, estilos, colores y el diseño de las interfaces de usuario dentro de los archivos XML relacionados.

1.3.2.2.5 Ciclo de Vida de una Actividad [25]

Una actividad pasa a diferentes estados de su ciclo de vida mientras el usuario navega por la aplicación. La clase Actividad ofrece un conjunto de devoluciones de llamada que permiten saber a la Actividad un cambio de estado, por ejemplo, que el sistema está creando, deteniendo o reanudando una actividad.

Dentro de los métodos de devolución de llamada de un ciclo de vida se establece cual es el comportamiento de la actividad cuando el usuario deje de usar la aplicación o la reanude. En la Figura 1.10 se visualiza una representación del ciclo de vida.

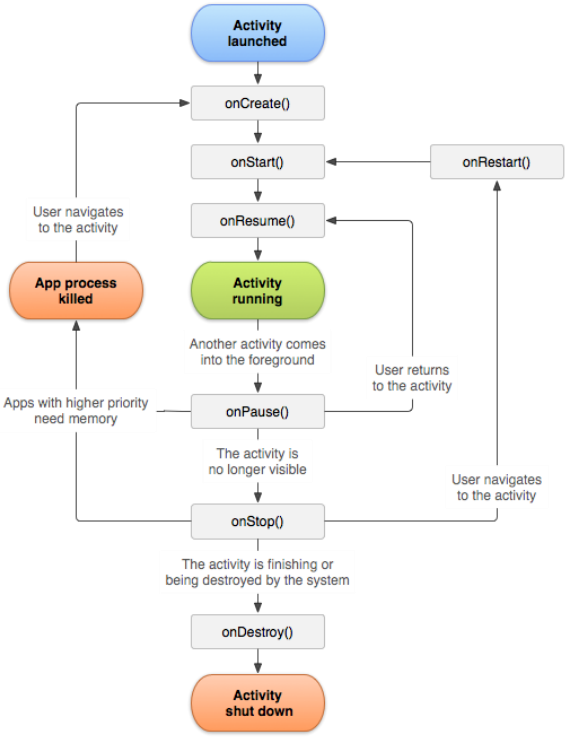


Figura 1.10. Ciclo de Vida de un Activity [25]

A continuación, se detallan las devoluciones de llamada.

- onCreate(): se implementa cuando el sistema crea la actividad por primera vez.
- onStarted(): La actividad reside en el estado onStarted() luego de estar en onCreate(). Esta llamada hace que el usuario mire la actividad.
- onResume(): Cuando finaliza el método onStarted() la actividad entra en el estado onResume(), aquí la actividad pasa a primer plano, en este estado el usuario interactúa con la aplicación, se permanece en este estado hasta que ocurra un evento que le quite de foco.
- onPause(): Cuando el usuario deja la actividad o cuando la actividad ya no está en primer plano se llama a este método.
- onDestroy(): Se llama a este método cuando la actividad está finalizando.

También existen métodos protegidos, se detalla uno que será de utilidad para el presente proyecto.

- onNewIntent(): Punto de entrada para actividades que ya se ejecutan en otro lugar, por lo que no se puede llamar al método onCreate().

1.3.3 TIPOS DE APLICACIONES MÓVILES [26]

Aplicaciones nativas: Son aplicaciones desarrolladas y optimizadas para un sistema operativo específico y la plataforma de desarrollo del fabricante.

Aplicaciones web: Hace uso de CSS y HTML para crear diseños web que se adapten a cualquier tamaño de pantalla, es decir, responsive web design. La desventaja es que no se tiene una aplicación como tal que se pueda descargar de una tienda en línea.

Aplicaciones híbridas: Estas aplicaciones tienen características de las aplicaciones nativas y de las aplicaciones web. Utiliza estándares de desarrollo web frontend y funciones de hardware del móvil.

1.3.4 BASE DE DATOS

Una base de datos es una estructura computarizada compartida e integrada que guarda un conjunto de datos y metadatos. Datos son los datos sin elaborar y metadatos son datos acerca de los datos. [27]

1.3.4.1 Gestor de base de datos

Con las bases de datos, el componente fundamental para la explotación de estas son los Sistemas Gestores de Bases de Datos. Estos son un conjunto de programas que administran la estructura de la base de datos e inspeccionan el acceso a los datos guardados en ella. Estos cumplen con la función de ser intermediarios entre el usuario y la base datos. [27]

1.3.4.2 Modelos de bases de datos [28]

Existen diversos modelos de bases de datos de acuerdo con la estructura que se utilice, algunos de las más comunes son:

1.3.4.2.1 Base de datos jerárquicas

Los datos se organizan a través de una estructura basada en nodos interconectados. Cada nodo puede tener un único padre y cero, uno o varios hijos. Así, se tiene una estructura en forma de árbol invertido en el que todos sus nodos dependen en última demanda de un nodo denominado *raíz*.

1.3.4.2.2 Bases de datos en red.

Este modelo posee ciclos en la estructura de la base de datos, lo que quiere decir que no existe un único padre para cada nodo, lo cual permite una mayor eficacia en lo que a la redundancia de datos se refiere.

1.3.4.2.3 Bases de datos relacionales

Es el modelo de bases de datos más utilizado en la actualidad. Este modelo hace uso de un esquema basado en tablas, esto resulta en una forma más fácil de comprender y utilizar las bases de datos para su análisis y consulta. Las tablas están integradas por un número de registros que son equivalentes a filas y campos que son equivalentes a columnas.

1.3.4.2.3 Bases de datos orientadas a objetos.

Este modelo desarrolla las bondades de las bases de datos relacionales, de tal modo que estas puedan contener objetos, permitiendo así una integración más fácil con la propia arquitectura de los programas empleados para el manejo de la base de datos, en caso de que estos hayan sido desarrollados mediante programación orientada a objetos.

En la Figura 1.11 se visualiza una comparación gráfica entre los modelos de bases de datos antes mencionados.

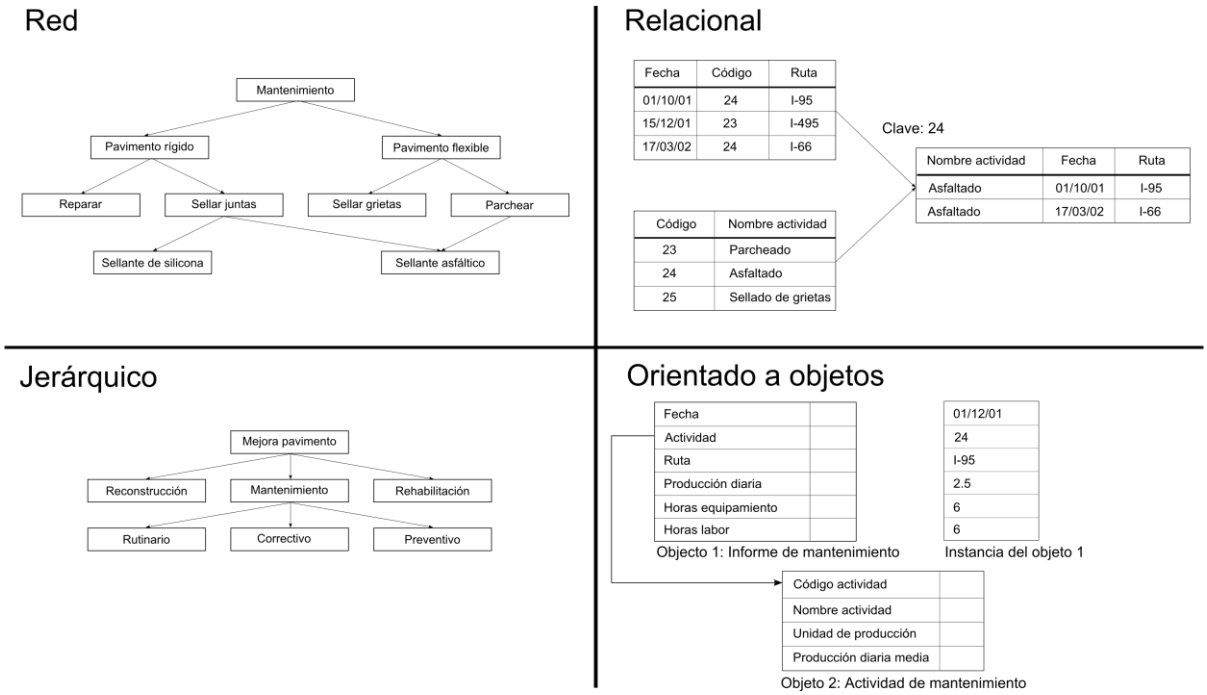


Figura 1.11. Comparación entre modelos de bases de datos [28]

1.3.4.3 Structure Query Language

SQL (Structure Query Language) es un lenguaje de consulta establecido como el lenguaje de alto nivel estándar para sistemas gestores de bases de datos relacionales. [27]

1.3.5 APLICACIÓN WEB

Una aplicación web es aquella aplicación a la que los usuarios finales pueden acceder mediante un servidor Web a través de Internet o de una intranet mediante un navegador. También se puede decir que es una aplicación software que está codificando en lenguajes que son soportados por los navegadores Web.

La arquitectura de una aplicación generalmente está estructurada como una aplicación de tres capas. La primera capa viene dada por el navegador Web, la segunda capa es un motor capaz de usar una tecnología Web dinámica, por ejemplo, PHP (Hypertext Preprocessor), esta capa comunica una base de datos con una aplicación web visible en el navegador web. La tercera capa es una base de datos. A través de peticiones el navegador web se comunica con la capa en medio que ofrece servicios haciendo uso de consultas y actualizaciones a la base de datos y que finalmente proporciona una interfaz de usuario. [29]

1.3.6 NGIN-X

Es un servidor web HTTP (Hypertext Transfer Protocol) de alto rendimiento que trabaja con varios lenguajes y tecnologías de desarrollo. Nginx es más rápido que otros servidores. Al hacer uso de sockets asíncronos, Nginx no genera procesos tantas veces como recibe solicitudes. Un proceso por núcleo es suficiente para manejar miles de conexiones, lo que lleva a una carga de CPU y consumo de memoria mucho más ligeros. [30] Entre sus módulos más frecuentes se puede referir a PHP-FPM. Es un modo de configuración del intérprete de PHP diseñado para incrementar el rendimiento. Esencialmente permite cachear peticiones que están en la memoria por un tiempo, de tal forma que, si se vuelven a originar, se envía el contenido de caché. Este proceso permite disminuir la carga del servidor y este atiende a un mayor número de usuarios con los mismos recursos. [31]

1.3.7 API

API (Application Programming Interfaces) es un conjunto de constantes, funciones y protocolos que permiten el desarrollo de aplicaciones. Así una API permite la comunicación entre dos aplicaciones de software a través de un ligado de normas.[32] La API se encuentra en el servidor, hace posible que el usuario final se comunique con el servidor para enviar o recibir información requerida por el usuario.

1.3.7.1 Web Services

Un Servicio Web es un componente al que se puede tener acceso mediante protocolos Web estándar. Un Servicio Web puede abarcar un conjunto de procedimientos a los que se puede llamar desde cualquier lugar de Internet, esta invocación es independiente de la plataforma que se utilice y del lenguaje de programación en el que se esté manejando el servicio.

El WC3 (World Wide Web Consortium) define un servicio Web como un sistema software diseñado para soportar interacciones máquina a máquina a través de la red. Es decir, los servicios Web brindan una forma estándar de interoperar entre aplicaciones software que se ejecutan en diferentes plataformas. [33]

Los tipos de Web Services son SOAP (Simple Object Access Protocol) y REST (Representational State Transfer). SOAP es un estándar de comunicación entre Servicios Web y aplicaciones, sus mensajes son definidos en formato XML (Extensible Markup Language) que viajan por lo general a través de HTTP. REST se integra de mejor manera con HTTP que SOAP porque no requiere de archivos XML, SOAP permite una rápida

implementación y consumo debido a que cuenta con una infraestructura liviana; además REST puede enviar y recibir datos como JSON (JavaScript Object Notation), XML o texto sin formato. [34]

1.3.7.2 Métodos HTTP [35]

Para realizar una acción, un conjunto de métodos de petición es definido por HTTP. A continuación, se describen algunos:

- GET: Pide una representación de un recurso determinado.
- POST: Envía una entidad a un recurso determinado.
- PUT: Sustituye las representaciones existentes del recurso de destino con la carga útil de la petición.
- DELETE: Borra un recurso determinado.
- CONNECT: establece una conexión de red hacia el servidor identificado por el recurso en forma de túnel.
- OPTIONS: Describe opciones de comunicación para el recurso de destino.

1.3.7.3 Formato de Datos

Los formatos más comunes para transferir la información en el uso de APIs son JSON y XML. JSON es un formato de texto ligero, su sintaxis está basada en el lenguaje JavaScript. JSON usa conformidades para el procesamiento de datos que son comunes a lenguajes como C, C++, C#, Java, Python, etc.; esto convierte a JSON en un lenguaje ideal para el intercambio de datos entre aplicaciones que se dispongan en diferentes escenarios. Dos estructuras básicas conforman un mensaje JSON; la primera es una estructura nombre – valor, y la segunda es una lista ordenada de valores representados en forma de arreglos o vectores. Las anteriores estructuras se usan para conformar los mensajes, el cliente y servidor interpreta y decodifica estos mensajes conforme al protocolo interno definido, para lo que es necesario el uso de librerías que permitan interpretar la sintaxis propia del lenguaje JSON.[36]

1.3.8 SISTEMA DE CONTROL DE VERSIONES

Un sistema de control de versiones es una aplicación que almacena en repositorios las versiones de un software generadas en el transcurso de su desarrollo y evolución. [37] De esta manera se puede recuperar versiones específicas cuando el archivo ya ha sido

modificado, comparar cambios a lo largo del tiempo, mirar quien modificó el archivo mejorando su desarrollo. Utilizar un sistema de control de versiones también significa que, si se pierde o se perjudica un archivo, la versión que estaba correcta se la podrá recuperar fácilmente.

1.3.8.1 GIT

Git es un sistema de control de versiones gratuito y de código abierto y uno de los más populares. Con Git varios desarrolladores pueden trabajar al mismo tiempo con sus propias copias de un archivo obtenidas de un repositorio. [38]

2. METODOLOGÍA

2.1 REQUERIMIENTOS DEL SISTEMA

El objetivo del proyecto es implementar un prototipo de administración de bienes mediante etiquetas NFC y el uso de una aplicación móvil. Este sistema tiene los siguientes elementos fundamentales; una base de datos, una aplicación web que permite administrar la base de datos, una aplicación móvil que lee y escribe las etiquetas NFC y un servidor web que recupera la información de la base datos.

Dentro de la base de datos para la información de cada bien se requieren los siguientes campos; nombre, tipo de bien, código NFC, serial, código de inventario, custodio, ubicación, fecha de ingreso, fecha de la compra, fecha de último mantenimiento, fecha de garantía, marca, procesador, estado y características. Al carecer de espacio disponible en la etiqueta para todos los campos, en esta se escribirán los campos considerados más importantes; código NFC, nombre, custodio, ubicación, código de inventario, serial y fecha de último mantenimiento.

Para el desempeño del objetivo del prototipo se cumple con:

- Iniciar sesión con credenciales en aplicación web y aplicación móvil.
- Lectura de una etiqueta NFC y asociar su identificador con el que está en la base de datos para visualizar los campos antes definidos mediante la aplicación móvil.
- Escritura de una etiqueta NFC para asociar con un identificador y añadir un bien con los campos antes definidos mediante la aplicación móvil.
- Visualizar el inventario de la base de datos en la aplicación web y aplicación móvil.

- Administración de la aplicación web para un manejo más rápido en cuanto a usuarios e inventario.
- Almacenamiento de datos en una Base de Datos con una tabla para bienes y otra tabla para usuarios.
- Uso de un servidor web para la recuperación de datos para la aplicación web y aplicación móvil.
- Lectura de una etiqueta NFC sin hacer uso de credenciales.

El usuario mediante el uso de sus credenciales, esto es usuario y contraseña, las que se gestionan en la aplicación web, ingresa a la aplicación móvil y la aplicación web. Mediante solicitudes HTTP se indica que es lo que se requiere. En la aplicación móvil el usuario escoge leer una etiqueta, escribir una etiqueta o visualizar el inventario, el servidor web responde a estas solicitudes consultando o insertando información en la base de datos.

En la aplicación web el usuario solicita gestionar la información de usuarios en cuanto a su visualización, creación, eliminación y edición. De la misma manera mediante el servidor web se realiza las consultas correspondiente a la base de datos, la que devuelve la información requerida.

2.2 ARQUITECTURA DEL PROTOTIPO

Para una mejor comprensión del prototipo en la Figura 2.1 se presenta su diagrama de despliegue.

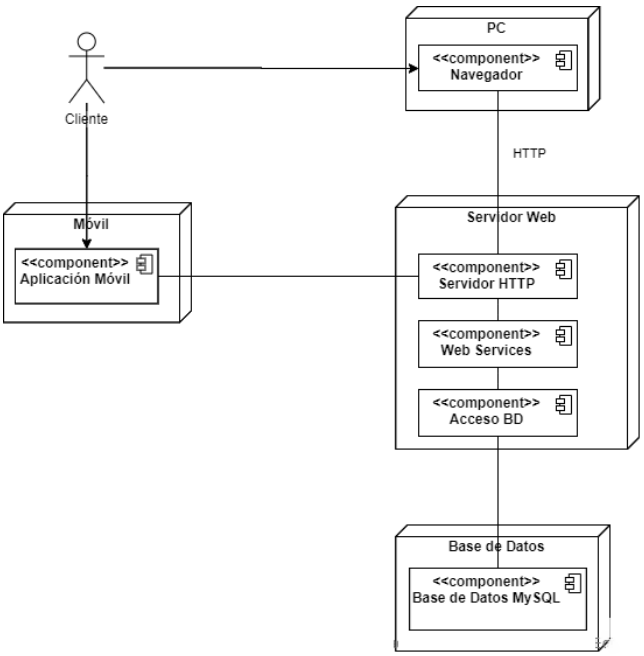


Figura 2.1. Diagrama de despliegue del prototipo. Fuente: Autor

El sistema consta de la etiqueta NFC, la cual se lee mediante la aplicación móvil para asociar su identificador en la base de datos y mediante el servicio web devolver los campos asociados al identificador. Para añadir un nuevo bien se escribe la etiqueta NFC con sus respectivos campos por medio de la aplicación móvil y mediante el web service se almacena en la base de datos.

También el usuario también puede acceder a la aplicación web mediante un navegador que se comunica con el servidor web para devolver el inventario completo y gestiona la información de los usuarios. Las dos aplicaciones se comunican con la misma base de datos, de la que se insertan y consultan los datos.

2.3 FUNCIONAMIENTO DEL PROTOTIPO

A continuación, se detalla la comunicación, que lenguajes, que software y que tecnologías se utilizan para el completo funcionamiento del sistema. Una apreciación gráfica se encuentra en la Figura 2.2.

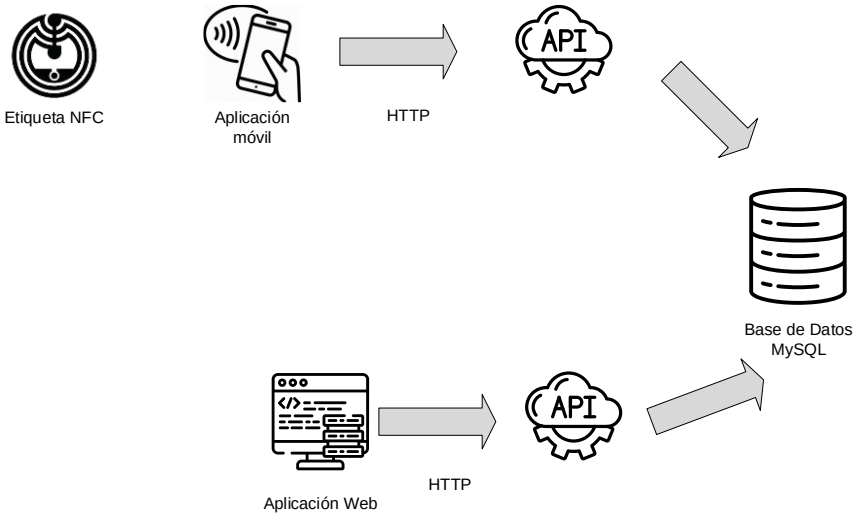


Figura 2.2. Funcionamiento del Sistema. Fuente: Autor

2.3.1 ETIQUETA NFC

Para el proyecto se seleccionó etiquetas que pertenecen a la serie NTAG, específicamente NTAG215 y NTAG213, este tipo de etiquetas tienen una compatibilidad universal con teléfonos móviles inteligentes que en el presente trabajo se utilizará como lector. Pertenecen a la norma ISO1443-3A, el Fórum NFC las define como etiquetas tipo 2, la duración garantizada del chip es de 10 años, se pueden leer o programar hasta 100 000 veces y no admiten encriptación. [39] Las etiquetas NTAG215 seleccionadas no trabajan sobre superficies metálicas, por lo que para el caso de bienes con superficies de metal se seleccionaron las etiquetas NTAG 213, diseñadas para esta función. Las características técnicas de las etiquetas se muestran en la Tabla 2.1.

Tabla 2.1. Características de la etiqueta NTAG215 y NTAG213[40]

Característica	**NTAG215**	**NTAG213 Anti-metal**
Memoria Total	540 bytes	180 bytes
Memoria Disponible	504 bytes	144 bytes
Longitud del texto	490 caracteres	130 caracteres
Retención de datos	10 años	
Resistencia a la lectura/escritura	1000 000 ciclos	
Encriptación	No	
Número de serie (UID)	Si (7 bytes)	
Compatibilidad	Compatible con teléfonos inteligentes NFC y lectores NFC ISO14443-A	

A continuación, se describen los campos de las características:

Memoria total: Es la memoria total, parte de la memoria no se puede utilizar ya que es utilizada para escribir la IUD de la etiqueta y otra información.

Memoria disponible: Es el total de memoria que si se puede escribir en la etiqueta.

Longitud del texto: Es el número máximo de caracteres para escribir un mensaje de texto en la etiqueta NFC.

Encriptación: una función de seguridad en el chip, que puede ayudar a prevenir la duplicación.

Número de serie: la etiqueta contiene en su interior un número único (llamado UID).

Compatibilidad universal: la etiqueta de los tipos NTAG son los únicos compatibles con todos los dispositivos.

NTAG213, NTAG215 y NTAG216 han sido desarrollados por NXP® Semiconductors como circuitos integrados de etiqueta NFC estándar para ser utilizados en aplicaciones de mercado, en combinación con dispositivos NFC o dispositivos de acoplamiento de proximidad compatibles con NFC. NTAG213, NTAG215 y NTAG216, generalmente llamados NTAG21x, están diseñados para cumplir totalmente con las especificaciones NFC Forum Tipo 2 Tag y ISO / IEC14443 Tipo A. [41]

2.3.2 APLICACIÓN WEB

Para el desarrollo de la aplicación web se utiliza el editor de texto Visual Studio Code que es libre y de código abierto. Para la estructuración de la aplicación se utiliza el lenguaje estándar de desarrollo web HTML (HyperText Markup Language) y se hace uso del framework, Bootstrap para el diseño de la aplicación. Bootstrap crea interfaces web mediante CSS y JavaScript y su principal característica es el de adaptar la interfaz al tamaño de pantalla del dispositivo desde donde se visualiza. Para la comunicación con la base de datos mediante el servidor se utiliza el lenguaje de programación PHP.

2.3.3 APLICACIÓN MÓVIL

Según IDC (International Data Corporation) la cuota de teléfonos inteligentes de Android se acerca al 86.6% en 2020. [42] Esto significa que Android se convierte en el sistema operativo más utilizado del mundo antes que IOS, sistema operativo para móviles de la

compañía Apple. De esta manera se ha seleccionado el sistema operativo Android para la realización de la aplicación del presente proyecto.

En cuanto al desarrollo de la aplicación se ha optado por realizar una aplicación nativa, al hacer uso de la tecnología NFC incorporada en el teléfono inteligente es mejor este tipo de aplicación ya que tiene un acceso completo al dispositivo en cuanto a hardware se refiere. Se utiliza Java como lenguaje de programación y Android Studio como IDE (Integrated Development Environment).

Para la comunicación con la base de datos se usa una API remota de tipo Servicios Web, sistema de comunicación en una red, usando el protocolo HTTP (Hypertext Transfer Protocol). La API devuelve la información en el formato para transferir información, JSON (Javascript Object Notation) y está escrita con lenguaje PHP (Hypertext Preprocessor).

2.3.4 BASE DE DATOS

Se utiliza el lenguaje SQL (Structured Query Language) para el desarrollo de una base de datos relacional, como gestor de la base de datos se utiliza MySQL y MySQL Workbench como herramienta visual de la base de datos.

2.3.5 SERVIDOR WEB

Se hace uso de un servidor que brinda servicio de hosting y una dirección IP estática. Se hace uso del terminal de Windows para el acceso remoto al servidor mediante el protocolo SSH (Secure Shell) y con la contraseña que el hosting brinda. En el servidor que maneja el sistema operativo Linux se instala NGINX, MySQL, PHP-FPM y GIT.

2.3.6 GITHUB

GitHub es un servicio comercial de alojamiento de repositorios Git remotos. [43] GitHub proporciona una interfaz web para crear repositorios o clonar repositorios existentes. En el presente proyecto se utiliza esta herramienta para crear un repositorio del proyecto de la aplicación web y de la aplicación móvil con el propósito de facilitar el desarrollo de estas y de sus modificaciones. GitHub brinda el uso de llaves SSH asociadas a la cuenta de GitHub, con el protocolo SSH, se puede conectar y autenticar en servidores y servicios remotos. Con las claves SSH se puede conectarse a GitHub sin proporcionar un nombre de usuario o contraseña en cada visita. Esto hace más fácil el desarrollo de las aplicaciones, al hacer uso de un terminal y guardar cambios, estos también se actualizan en el servidor. Se generó una clave SSH para el computador personal y para el servidor.

2.4 ESTRUCTURA DE LA BASE DE DATOS

La base de datos se elabora con las siguientes tablas con sus respectivas columnas:

- **User_groups:** id, group_name, group_level, group status.
- **Users:** id, name, username, password, user_level, status.
- **Products:** id, nombre, codigo_nfc, serial, codigo_inventario, custodio, fecha_ingreso, fecha_compra, ubicacion, fecha_ultimo_mantenimiento, fecha_garantia, marca, procesador, estado, caracteristicas, tipo.

En la Figura 2.3 se muestra el diagrama entidad-relación generado por MySQL. En el diagrama se puede observar las respectivas tablas y columnas asignadas a la base de datos con sus respectivas características. La tabla users utiliza los datos de la tabla user_groups.

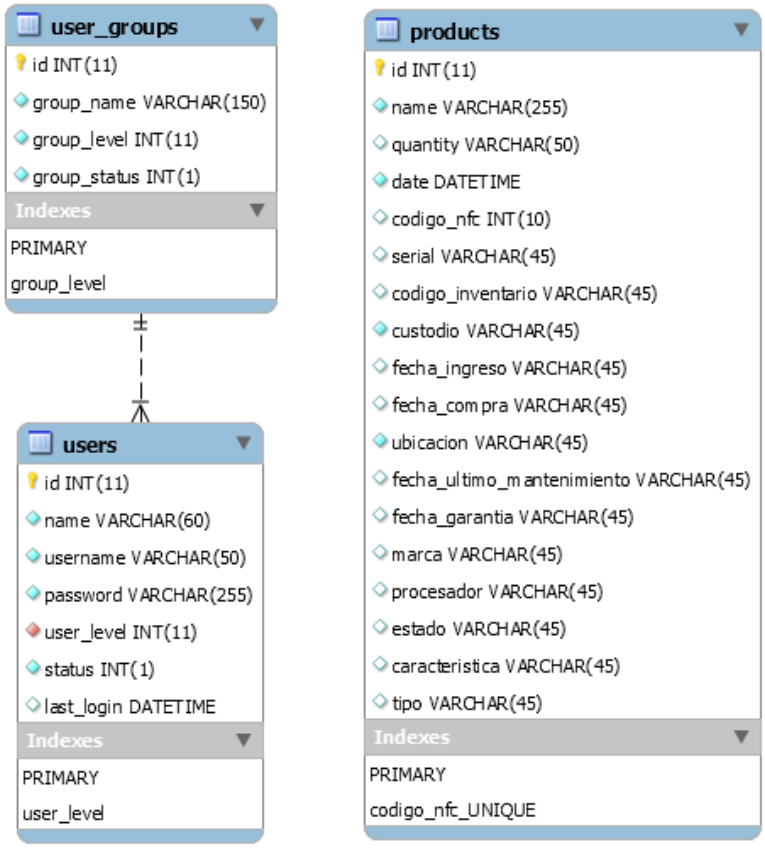


Figura 2.3. Diagrama entidad - relación de la base de datos. Fuente: MySQL Workbench

La conexión hacia la base de datos se realiza utilizando los siguientes parámetros; generando una llave SSH en el computador personal, la IP del servidor, el usuario y puerto MySQL server, así como el usuario y contraseña, como se indica en la Figura 2.4.

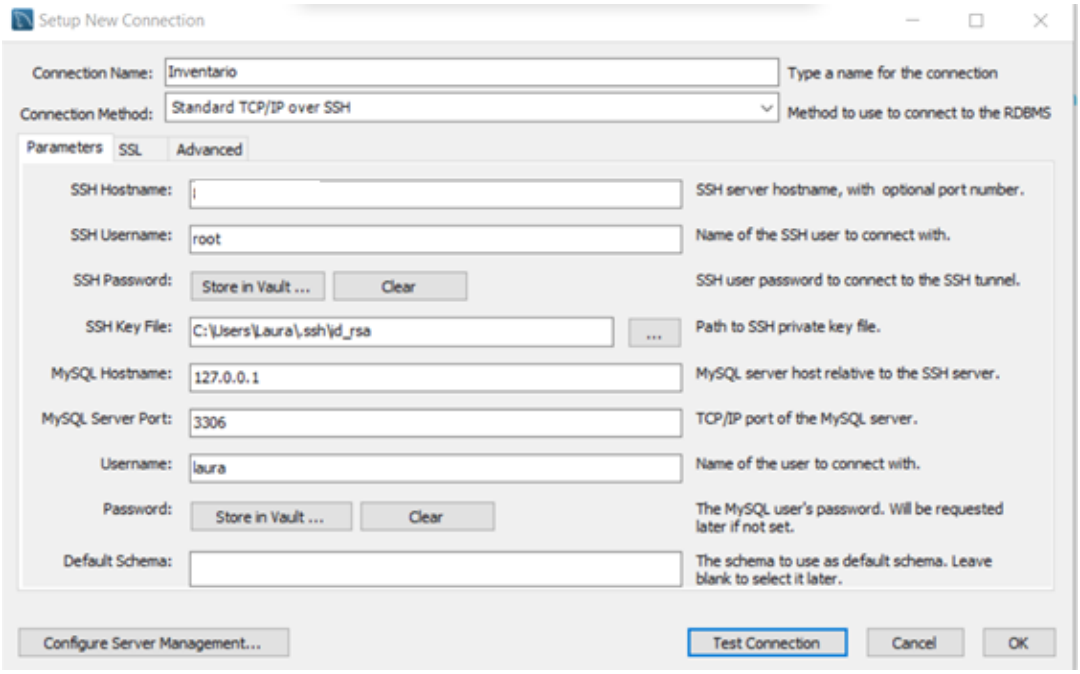


Figura 2.4. Conexión a la base de datos. Fuente: MySQL Workbench

2.5 DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN WEB

La aplicación web permite la administración de la base de datos en cuanto a crear, visualizar, modificar y eliminar la información tanto de bienes como de usuarios. La importancia de la aplicación web radica en que el usuario final no tendrá que manejar directamente el lenguaje SQL para administrar los datos. La aplicación web se accede desde cualquier navegador web con acceso a Internet mediante un usuario y una contraseña.

La aplicación web está conformada por varios archivos desarrollados con lenguaje de programación PHP dentro de HTML. En cada interfaz se incluye el archivo *load.php* este archivo incluye a archivos de configuración como; *config.php*, database.php, functions.php, sql.php. A continuación, se detalla que realiza cada uno.

- config.php, contiene la información necesaria para conectarse a la base de datos: host, usuario, contraseña y nombre de la base de datos.

- *functions.php*, este archivo contiene las funciones necesarias para cada requerimiento de la aplicación, como; eliminar caracteres especiales en un string para usar en una sentencia SQL, eliminar caracteres HTML, verificar que los campos de entrada no estén vacíos, redirigir a una página.
- *database.php,* en este archivo se definen las funciones para abrir y cerrar la conexión con la base de datos.
- *sql.php,* aquí se definen funciones para modificaciones de la base de datos, como borrar, buscar, insertar y actualizar datos de bienes y usuarios mediante sentencias SQL.

2.5.1 INTERFAZ DE INICIO

En la interfaz de inicio se accede mediante un usuario y contraseña almacenados en la base de datos, se visualiza en la figura 2.5.

Figura 2.5. Interfaz de inicio. Fuente: Autor

La aplicación web inicializa con el archivo *index.php*, nombre comúnmente utilizado por las páginas web para la primera interfaz. En este se hace un llamado al archivo *load.php* que como se explicó antes contiene todas las funcionalidades para el correcto funcionamiento de la aplicación.

Al completar los datos el botón *Iniciar sesión* lleva a un proceso de autenticación, que se describe en el diagrama de flujo de la Figura 2.6.

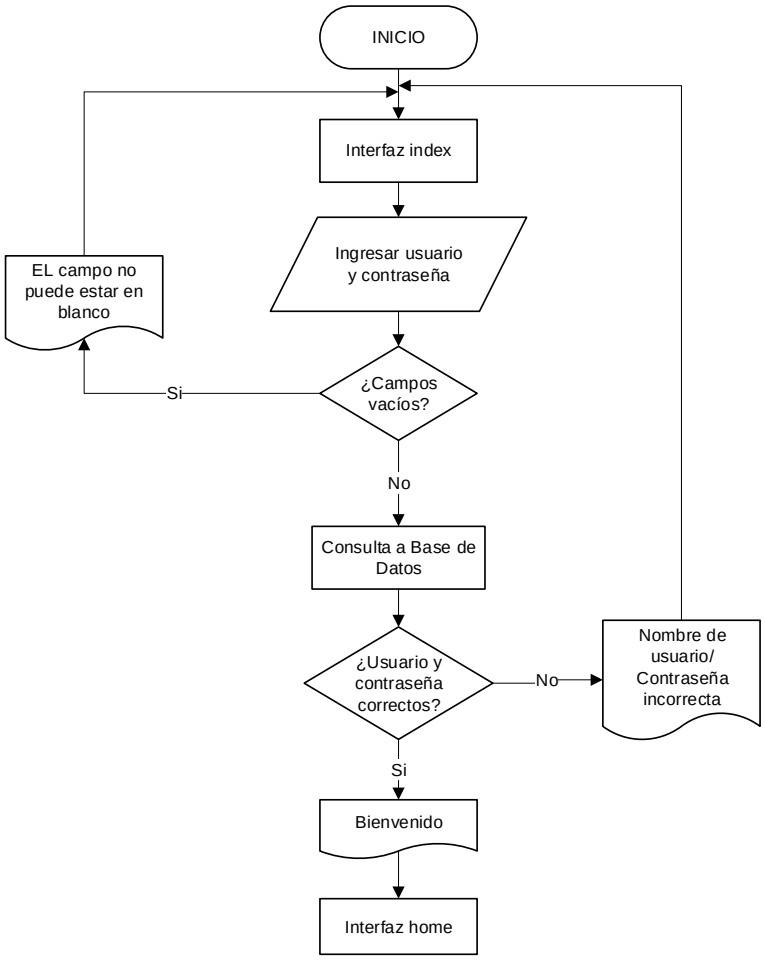


Figura 2.6. Diagrama de Flujo - Autenticación. Fuente: Autor

Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *auth.php*. En la Figura 2.7 se muestra cómo se realiza este proceso de autenticación, se consulta en la base de datos los campos, se realiza la comprobación y si es correcta, se redirecciona a la página *Home*.

```
function authenticate($username='', $password='') {
  global $db;
  $username = $db->escape($username);
  $password = $db->escape($password);
  $sql  = sprintf("SELECT id,username,password,user_level FROM users WHERE username ='%s' LIMIT 1", $username);
  $result = $db->query($sql);
  if($db->num_rows($result)){
    $user = $db->fetch_assoc($result);
    $password_request = sha1($password);
    if($password_request === $user['password'] ){
      return $user['id'];
    }
  }
 return false;
}
```

Figura 2.7. Función para autenticación. Fuente: Autor

La aplicación está diseñada para que los usuarios añadidos sean asignados a un grupo. Existe un grupo de usuarios administradores que tienen acceso a la gestión de usuarios y bienes. Y existe otro grupo de usuarios especiales que tienen acceso solamente a la gestión de los bienes. A continuación, se detallan estos aspectos.

2.5.2 INTERFAZ HOME

Después de iniciar sesión se muestra la interfaz home que se mira en la Figura 2.8, en esta interfaz se muestra el menú de opciones, la fecha actual, y acceso al perfil. Usuarios administradores y usuarios especiales tienen acceso a esta página.

Figura 2.8. Interfaz Home. Fuente: Autor

2.5.3 INTERFAZ RESUMEN

En esta opción del menú se visualiza la cantidad de usuarios, la cantidad de bienes y los bienes añadidos recientemente como se visualiza en la Figura 2.9. Solo usuarios administradores tienen acceso a esta página.

Figura 2.9. Interfaz Resumen. Fuente: Autor

Los usuarios y bienes se cuentan mediante la función *count_by_id* del archivo *sql.php* y que se muestra en la figura 2.10.

```
function count_by_id($table){
  global $db;
  if(tableExists($table))
  {
    $sql    = "SELECT COUNT(id) AS total FROM ".$db->escape($table);
    $result = $db->query($sql);
     return($db->fetch_assoc($result));
  }
}
```

Figura 2.10. Función count_by_id. Fuente: Autor

2.5.4 INTERFAZ USUARIOS – ADMINISTRAR GRUPOS

En esta opción como se mira en la Figura 2.11 el administrador puede mirar la información de cada grupo, así también puede añadir, editar y eliminar los grupos; los grupos definen lo que el usuario es capaz de administrar y visualizar. Solo usuarios administradores tienen acceso a esta página.

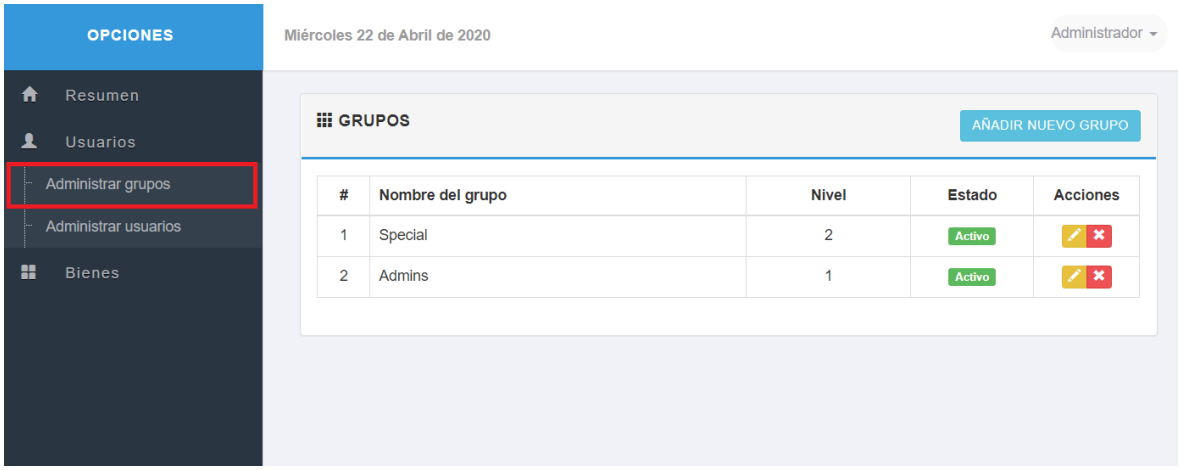


Figura 2.11. Interfaz Usuarios - Administrar grupos. Fuente: Autor

Al presionar el botón *AÑADIR NUEVO GRUPO* se muestra la interfaz que se muestra en la Figura 2.12.

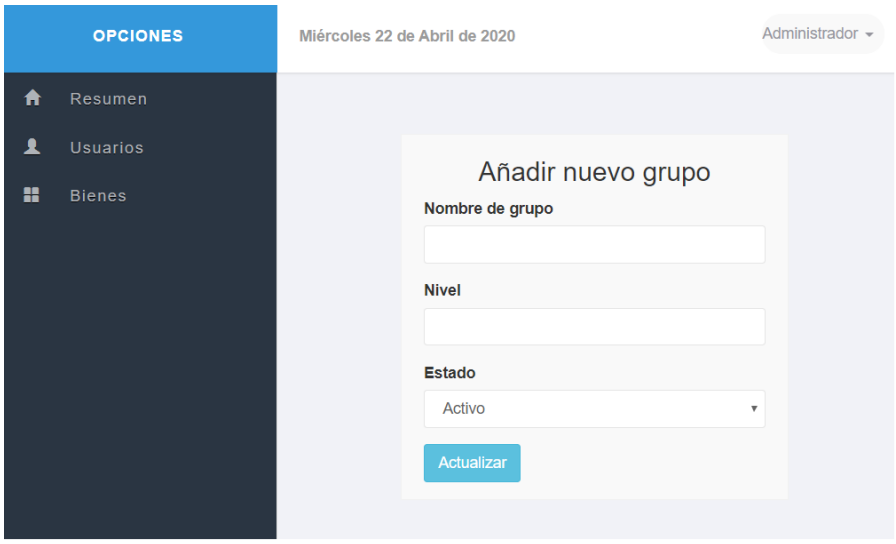


Figura 2.12. Interfaz Añadir Grupo. Fuente: Autor

En el diagrama de flujo de la Figura 2.13 se muestra el proceso en caso de añadir un nuevo grupo.

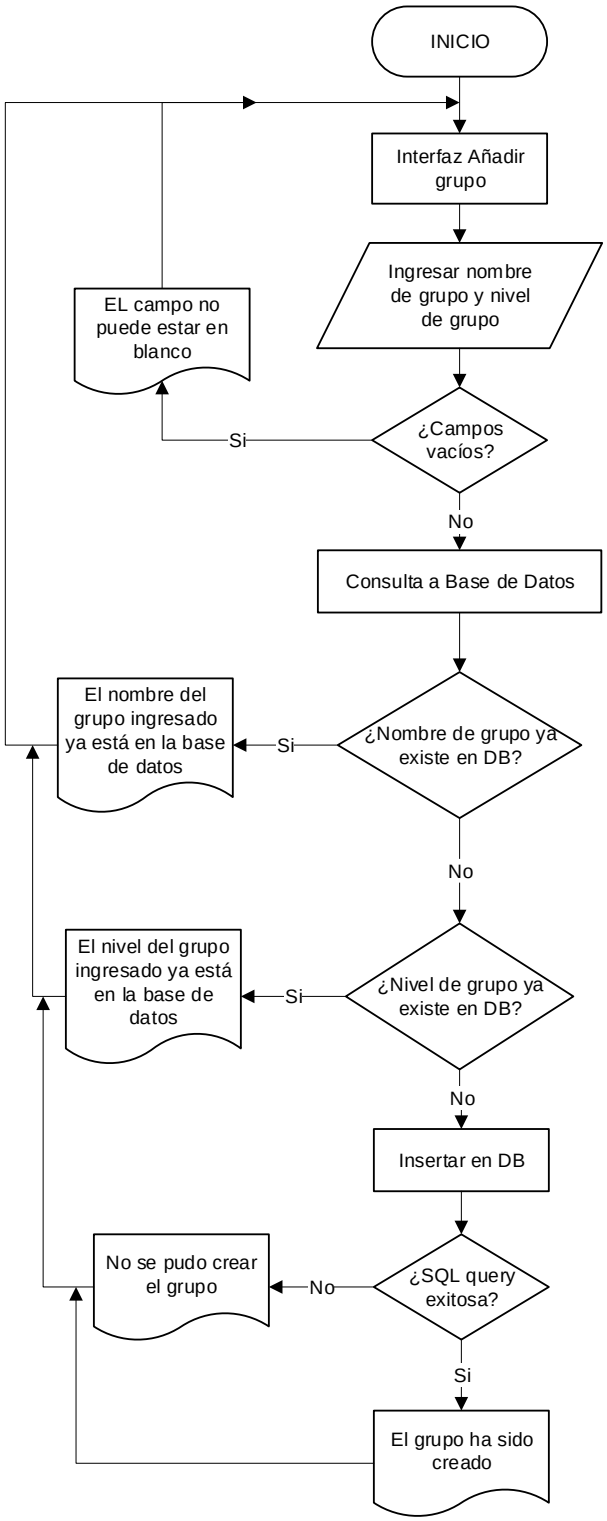


Figura 2.13. Diagrama de Flujo - Interfaz Añadir Grupo. Fuente: Autor

En la Figura 2.14 se muestran las sentencias SQL para la inserción en la base de datos de los nuevos datos. Se inserta en la base de datos los campos, se realiza la comprobación y si es correcta, se muestra el mensaje correspondiente. Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *add_group.php*.

```php
$query  = "INSERT INTO user_groups (";
$query .="group_name,group_level,group_status";
$query .=") VALUES (";
$query .=" '{$name}', '{$level}','{$status}'";
$query .=")";
if($db->query($query)){
  $session->msg('s',"El grupo ha sido creado ");
  redirect('add_group.php', false);
} else {
  $session->msg('d',' No se pudo crear el grupo.');
  redirect('add_group.php', false);
}
```

Figura 2.14. Añadir un nuevo grupo. Fuente: Autor

2.5.5 INTERFAZ USUARIOS – ADMINISTRAR USUARIOS

Como se muestra en la figura 2.15, en esta interfaz el administrador puede mirar la información de cada usuario, así como añadir, editar y eliminar un usuario; al editar el usuario se puede definir a que grupo pertenece y establecer una contraseña. Solo usuarios administradores tienen acceso a esta página.

Figura 2.15. Interfaz Usuarios - Administrar usuarios. Fuente: Autor

Al presionar el botón AÑADIR NUEVO USUARIO se muestra la interfaz que se muestra en la Figura 2.16, donde se piden los datos necesarios.

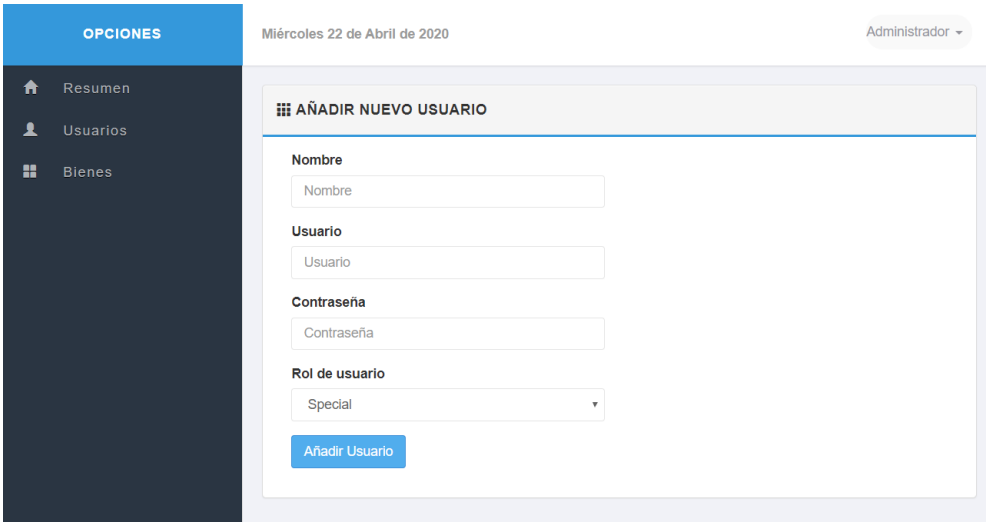


Figura 2.16. Interfaz Añadir Usuario. Fuente: Autor

A continuación, se muestra el diagrama de flujo en la Figura 2.17 del funcionamiento de añadir un nuevo usuario.

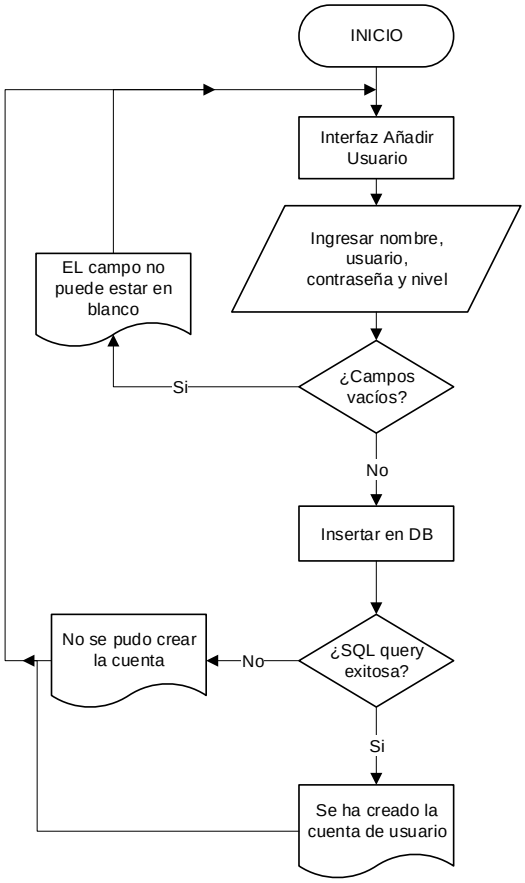


Figura 2.17. Diagrama de Flujo - Interfaz Añadir Usuario. Fuente: Autor

Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *add_user.php*. Para añadir un usuario se utilizan las sentencias SQL que se muestra en la Figura 2.18. se inserta en la base de datos los campos, se realiza la comprobación y si es correcta, se muestra el mensaje correspondiente.

```
$query = "INSERT INTO users (";
$query .="name,username,password,user_level,status";
$query .=") VALUES (";
$query .=" '{$name}', '{$username}', '{$password}', '{$user_level}','1'";
$query .=")";
if($db->query($query)){
  //sucess
  $session->msg('s',"¡Se ha creado la cuenta de usuario! ");
  redirect('add_user.php', false);
} else {
  //failed
  $session->msg('d','No se pudo crear la cuenta.');
  redirect('add_user.php', false);
}
```

Figura 2.18. Añadir un usuario. Fuente: Autor

2.5.6 INTERFAZ BIENES - ADMINISTRAR BIENES

En esta parte de las opciones se visualiza el inventario con los campos requeridos, también se puede añadir, editar y eliminar un bien como se visualiza en la Figura 2.19. Aquí el usuario también puede exportar a Excel la tabla del inventario completo. Usuarios administradores y usuarios especiales tienen acceso a esta página.

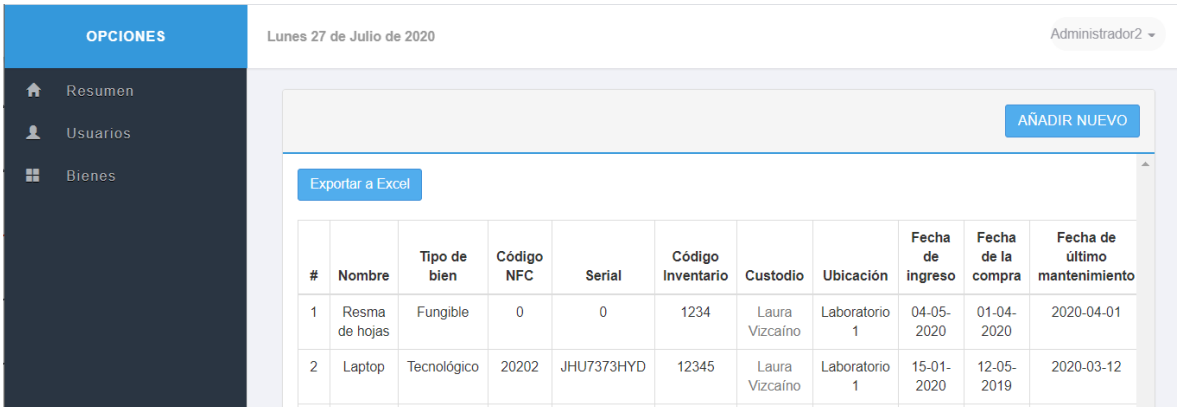


Figura 2.19. Interfaz Administrar Bienes. Fuente: Autor

2.5.7 INTERFAZ BIENES - AÑADIR BIEN

Finalmente, para el menú de opciones como se puede ver en la Figura 2.20, en esta interfaz se puede añadir un nuevo bien completando todos los campos requeridos. Usuarios

administradores y usuarios especiales tienen acceso a esta página. Se ha añadido esta opción como una alternativa cuando las etiquetas NFC no son necesarias, en el caso de bienes fungibles como, por ejemplo, una resma de papel, y estos bienes también deben ser inventariados. Los campos por completarse son los mismos que tiene la aplicación móvil, excepto el campo para el código NFC ya que no es necesario en estos casos.

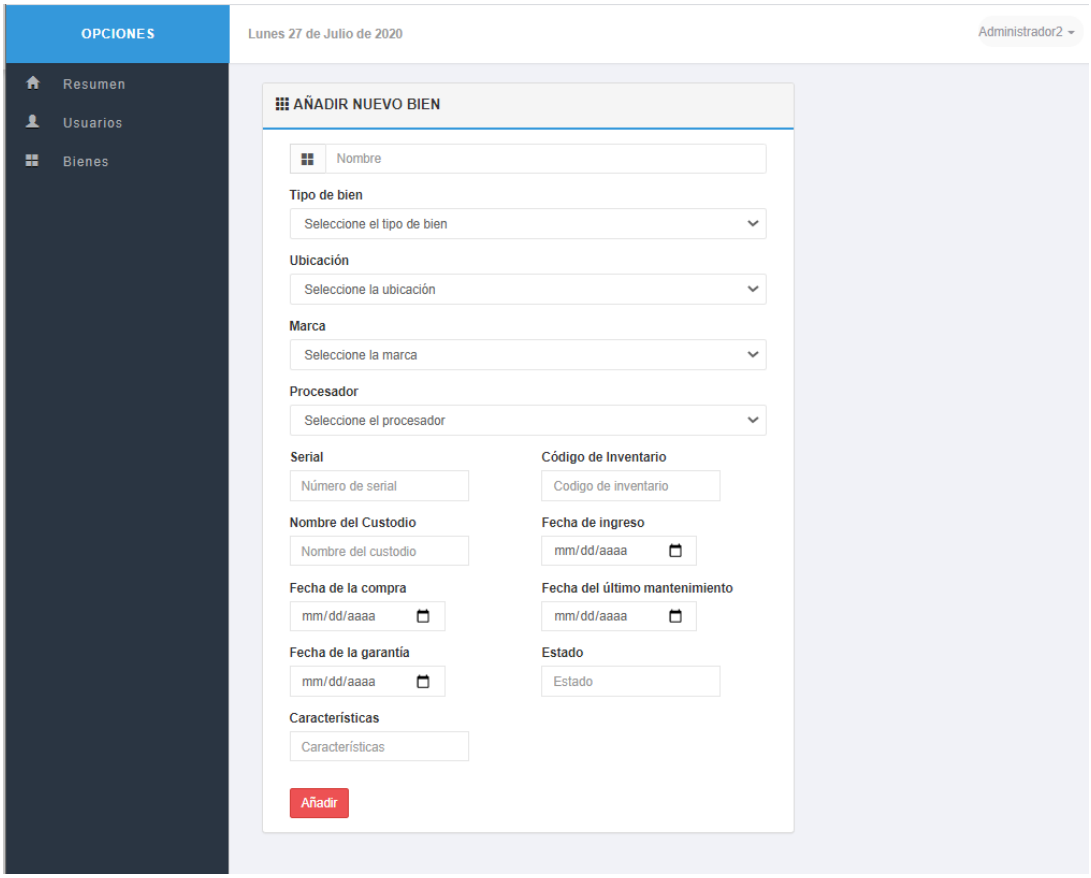


Figura 2.20. Interfaz Añadir Bien. Fuente: Autor

Al presionar en el botón *AÑADIR* se realiza el proceso mostrado en el diagrama de flujo de la Figura 2.21.

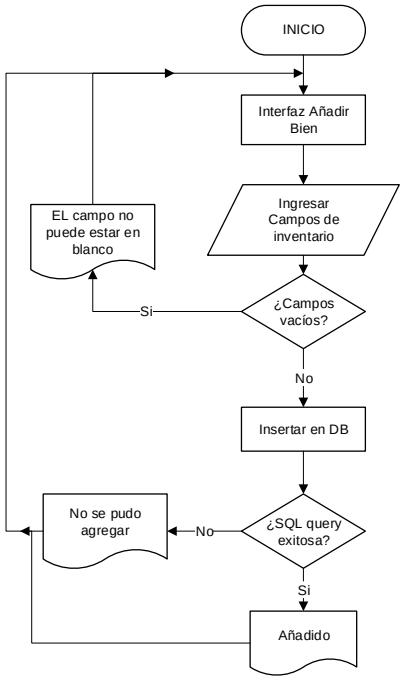


Figura 2.21. Diagrama de Flujo - Añadir Bien. Fuente: Autor

Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *add_product.php*. En la figura 2.22 se muestra las sentencias SQL para añadir un nuevo bien.

```
$date    = make_date();
$query  = "INSERT INTO products (";
$query .=" name,tipo,codigo_nfc,serial,codigo_inventario,custodio,ubicacion,fecha_ingreso,fecha_compra,f
$query .=") VALUES (";
$query .=" '{$p_name}', '{$p_tipo}', '{$p_nfc}', '{$p_ser}', '{$p_cod}', '{$p_cus}', '{$p_ubi}','{$p_ing
$query .=")";
$query .=" ON DUPLICATE KEY UPDATE name='{$p_name}'";
if($db->query($query)){
  $session->msg('s',"Añadido");
  redirect('add_product.php', false);
} else {
  $session->msg('d',' No se pudo agregar!');
  redirect('product.php', false);
}
```

Figura 2.22. Añadir un nuevo bien. Fuente: Autor

2.5.8 PERFIL

Como se puede mirar en la Figura 2.23 en la parte superior derecha donde aparece el nombre de usuario existe la opción de editar la información de este. Aquí también se puede salir de la aplicación y redirigir a la página de autenticación. Usuarios administradores y usuarios especiales tienen acceso a esta opción.

Figura 2.23. Perfil. Fuente: Autor

En *Ajustes* se muestra la interfaz que se mira en la Figura 2.24. Al hacer clic en el botón *Actualizar* se realiza el proceso que se muestra en el diagrama de flujo de la Figura 2.25 y se actualizan los campos nombre y usuario.

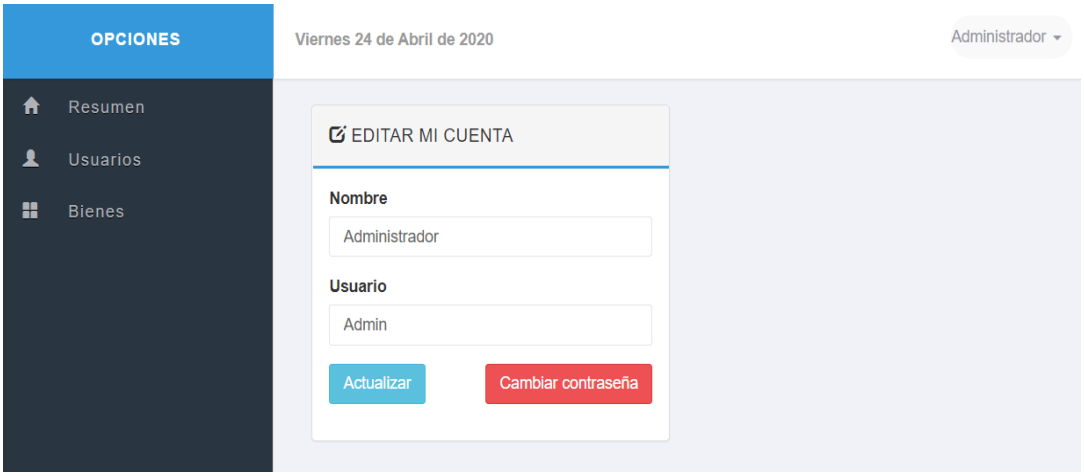


Figura 2.24. Interfaz Editar cuenta. Fuente: Autor

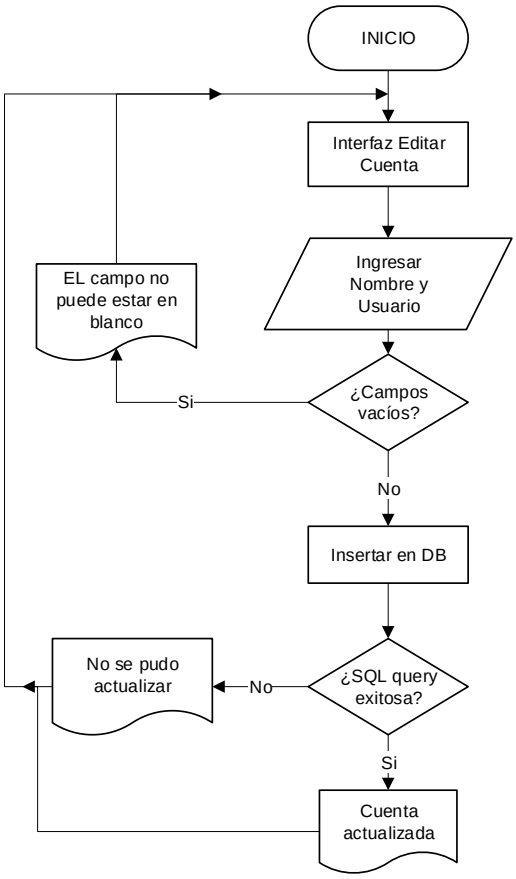


Figura 2.25. Diagrama de Flujo - Editar Cuenta. Fuente: Autor

Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *edit_account.php*. En la figura 2.26 se muestran las sentencias SQL para editar los campos nombre y usuario.

```
$sql = "UPDATE users SET name ='{$name}', username ='{$username}' WHERE id='{$id}'";
$result = $db->query($sql);
if($result && $db->affected_rows() === 1){
  $session->msg('s',"Cuenta actualizada");
  redirect('edit_account.php', false);
} else {
  $session->msg('d',' No se pudo actualizar');
  redirect('edit_account.php', false);
}
```

Figura 2.26. Código para actualizar la cuenta. Fuente: Autor

En el caso de hacer clic en el botón *Cambiar contraseña* se muestra la interfaz de la Figura 2.27, se debe ingresar la nueva y anterior contraseña y se realiza el proceso que se muestra en el diagrama de flujo que se mira en la Figura 2.28.

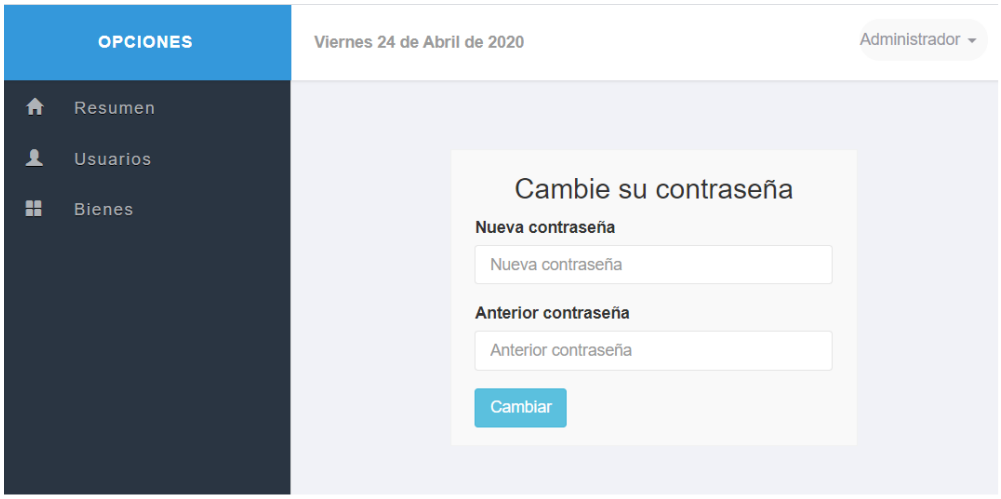


Figura 2.27. Interfaz Cambiar Contraseña. Fuente: Autor

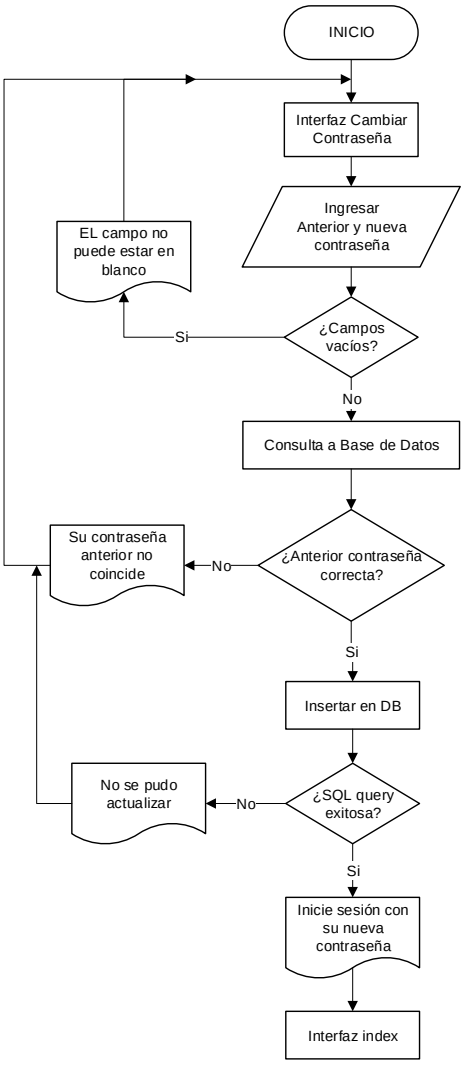


Figura 2.28. Diagrama de Flujo - Cambiar Contraseña. Fuente: Autor

Con el elemento HTML *form* se envía los datos mediante el método POST y la URI donde se procesa esta información con el archivo *change_password.php*. En la figura 2.29 se muestra el código cambiar la contraseña.

```
$id = (int)$_POST['id'];
$new = remove_junk($db->escape(sha1($_POST['new-password'])));
$sql = "UPDATE users SET password ='{$new}' WHERE id='{$db->escape($id)}'";
$result = $db->query($sql);
  if($result && $db->affected_rows() === 1):
    $session->logout();
    $session->msg('s',"Inicie sesión con su nueva contraseña.");
    redirect('index.php', false);
  else:
    $session->msg('d',' No se pudo actualizar!');
    redirect('change_password.php', false);
  endif;
```

Figura 2.29. Código para cambiar contraseña. Fuente: Autor

2.6 DISEÑO E IMPLEMENTACIÓN DE LA APLICACIÓN MÓVIL

La aplicación permite cinco casos de uso, primero iniciar sesión que permite acceder a la lectura de la etiqueta, escritura de la etiqueta y visualización de inventario; estas acciones se comunican con el servidor para devolver la información requerida. Además, sin iniciar sesión se incluye la opción de leer la etiqueta, en el caso de que no se disponga de acceso a Internet para mirar la información más importante del bien. Esto se muestra gráficamente en el diagrama de casos de uso de la Figura 2.30.

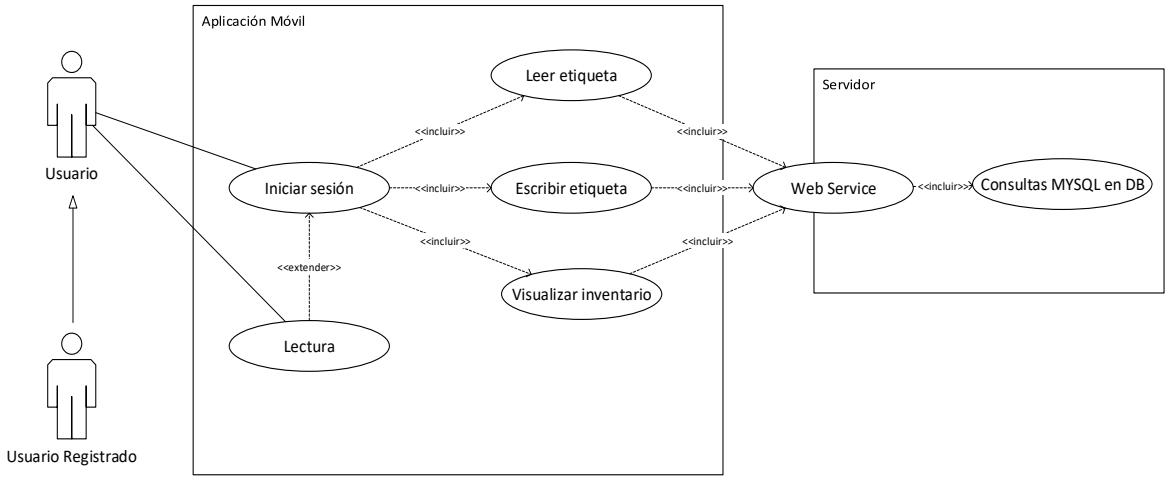


Figura 2.30. Diagrama de casos de uso de la aplicación móvil. Fuente: Autor

La función de la aplicación móvil es mediante la lectura de la etiqueta NFC mostrar la información correspondiente al bien donde se encuentra la etiqueta, y mediante la escritura

de la etiqueta, añadir un nuevo bien. La aplicación está compuesta por seis actividades cada una con su respectivo layout y cuatro archivos para web service con PHP.

Los web services para la presente aplicación se usan para la comunicación con la base de datos, estos son especificados en cada actividad con la dirección IP del servidor que brinda el servicio de hosting.

Cuando se crea el proyecto para la aplicación, Android Studio permite elegir el mínimo nivel de API para la misma como se indica en la Figura 2.31. En en este caso se eligió el nivel de API 21, con lo que la aplicación correrá en aproximandamente 94.1% de los dispositivos.

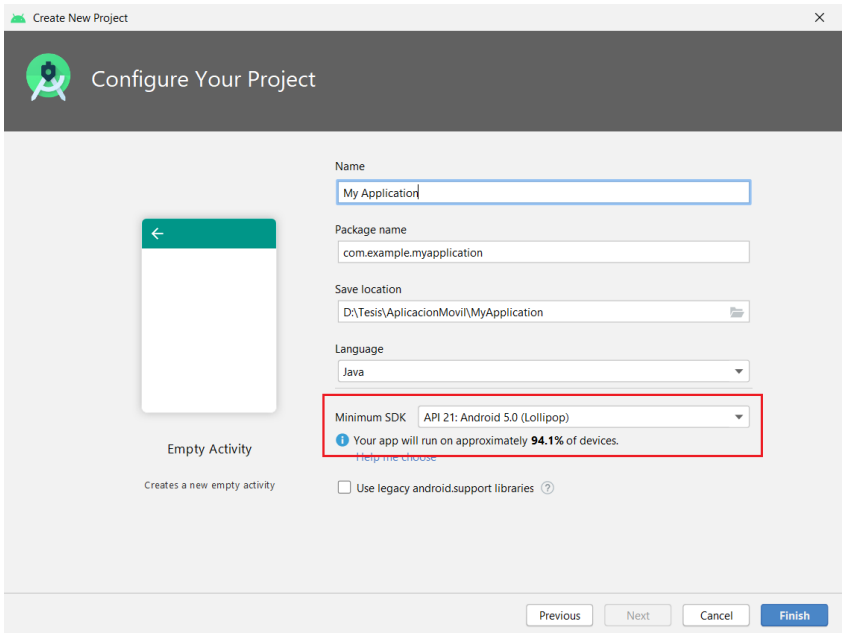


Figura 2.31. Escoger mínimo nivel de API. Fuente: Android Studio

Para el caso de las dos actividades que son de lectura y escritura de las etiquetas se utiliza el *Sistema de Despacho en Primer Plano*, este permite que una actividad intercepte un Intent NFC y reclame prioridad sobre las otras actividades que están registradas para manejar el mismo Intent. De esta manera, cuando la aplicación está en primer plano, no se le pedirá al usuario que elija qué aplicación abrir para hacer uso de la tecnología NFC incorporada en el móvil. Este método despacho en primer plano se muestra en la Figura 2.32.

```
private void enableForegroundDispatchSystem(){
    Intent intent = new Intent( packageContext: this, Write.class).addFlags(Intent.FLAG_RECEIVER_REPLACE_PENDING);

    PendingIntent pendingIntent = PendingIntent.getActivity( context: this,  requestCode: 0, intent,  flags: 0);

    IntentFilter[] intentFilters = new IntentFilter[] {};

    nfcAdapter.enableForegroundDispatch( activity: this,pendingIntent,intentFilters, techLists: null);
}

private void disableForegroundDispatchSystem(){

    nfcAdapter.disableForegroundDispatch( activity: this);

}
```

Figura 2.32. Sistema de Despacho en Primer Plano. Fuente: Autor

Cuando la actividad empieza se llama al método *onResume*, luego se habilita el sistema de despacho en primer plano en el adaptador NFC que indica que cualquier Intent debe ser enviado a la actividad donde se está ejecutando. Cuando la actividad recibe un Intent entrante, se llama al método *onNewIntent*. Cuando el usuario abandona la aplicación o el dispositivo entra en modo de suspensión, se llama al método *onPause*, lo que significa que la aplicación ya no está en uso activo; por lo tanto, es una buena práctica deshabilitar el sistema de envío en primer plano o pueden ocurrir comportamientos inesperados. Si el usuario vuelve a la aplicación, se llama nuevamente al método *onResume*, y con eso, el sistema de despacho en primer plano se inicia nuevamente. [16] Este proceso se realiza como se indica en la Figura 2.33.

```
@Override
protected void onResume() {
    super.onResume();
    enableForegroundDispatchSystem();
}

@Override
protected void onPause() {
    super.onPause();
    disableForegroundDispatchSystem();
}
```

Figura 2.33. Métodos onResume y onPause. Fuente: Autor

Para el correcto funcionamiento de la aplicación en el archivo AndroidManifest.xml se deben especificar los permisos para el uso de Internet y NFC, como se visualiza en la Figura 2.34.

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.NFC"></uses-permission>
```

Figura 2.34. Permisos en Manifest. Fuente: Autor

En el archivo de dependencias de Gradle se debe implementar la librería Volley, esta se utiliza para optimizar el envío de peticiones HTTP desde la aplicación Android hacia servidores externos, como se observa en la Figura 2.35. Volley se integra fácilmente con cualquier protocolo y es compatible con JSON. [44]

```
implementation 'com.android.volley:volley:1.1.1'
```

Figura 2.35. Implementar Volley. Fuente: Autor

2.6.1 ACTIVIDAD LOGIN

En la Figura 2.36 se mira la primera interfaz de la aplicación, en esta interfaz el usuario debe iniciar sesión con las mismas credenciales que se usa en el caso de la aplicación web, las que se encuentran en la base de datos.

Figura 2.36. Interfaz Login. Fuente: Autor

Para esta actividad se utiliza el archivo para web service, *wsJSONLogin.php*, en este archivo se hace la conexión con la base de datos, mediante un condicional se verifica cualquier error de conexión existente mostrando un mensaje si este se diera. Mediante el método POST se envían los datos y se usa una sentencia para consultar en la base de datos, mediante un condicional se valida si se encontró el registro con los datos enviados y se lo asigna a un array, se convierte el array a una cadena JSON y se cierra la conexión existente. En la Figura 2.37 se muestra el código para la consulta.

```
$username=$_POST['username'];
$password=sha1($_POST['password']);

$sentencia=$conexion->prepare("SELECT * FROM users WHERE username=? AND password=?");
$sentencia->bind_param('ss',$username,$password);
$sentencia->execute();

$resultado = $sentencia->get_result();
if ($fila = $resultado->fetch_assoc()) {
        echo json_encode($fila,JSON_UNESCAPED_UNICODE);
}
$sentencia->close();
$conexion->close();
```

Figura 2.37. Código para consultar credenciales. Fuente: Autor

Para el código de Android Studio se debe crear las variables para editar texto y un botón, en el método *onCreate* se vinculan las variables declaradas con los controles del layout. Se crea un método llamado *validarUsuario* agregando un parámetro string destinado a la ruta del web service donde se especifica el método POST y la URL para el web service. Si el usuario y contraseña existen, mediante un Intent se lanza la siguiente actividad que para el caso es el Menú que se verá adelante. En la Figura 2.38 se muestra el código para este proceso.

```
private void validarUsuario(String URL){
    StringRequest stringRequest=new StringRequest(Request.Method.POST, URL, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            if(!response.isEmpty()){
                Intent intent=new Intent(getApplicationContext(),Menu.class);
                startActivity(intent);
            }else{
                Toast.makeText( context: Login.this, text: "Usuario o contraseña incorrectos",Toast.LENGTH_SHORT).show();
            }
        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText( context: Login.this,error.toString(),Toast.LENGTH_SHORT).show();
        }
    }){
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String,String> parametros=new HashMap<~>();
            parametros.put("username",username.getText().toString());
            parametros.put("password",password.getText().toString());
            return parametros;
        }
    };

    RequestQueue requestQueue=Volley.newRequestQueue( context: this);
    requestQueue.add(stringRequest);
}
```

Figura 2.38. Método validarUsuario. Fuente: Autor

En el método *onCreate* se genera el evento *click* asociando este a la variable del botón *INICIAR*, dentro de este se llama al método *validarUsuario* colocando dentro de este la URL del servicio. En el diagrama de flujo de la Figura 2.39 se muestra el proceso indicado.

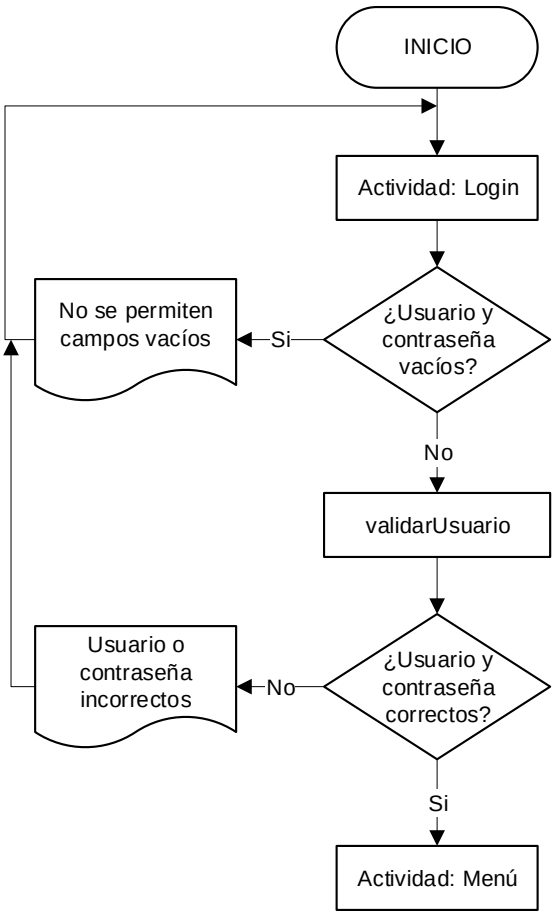


Figura 2.39. Diagrama de Flujo - Actividad Login. Fuente: Autor

2.6.2 ACTIVIDAD MENÚ

Al acceder con el usuario y contraseña correctas la aplicación lleva a la interfaz del menú como se muestra en la Figura 2.40. Aquí el usuario elige que acción realizar entre; leer la etiqueta, escribir sobre la etiqueta o visualizar el inventario completo.

Figura 2.40. Interfaz Menú. Fuente: Autor

Para esta actividad no se tiene un archivo para web service ya que funciona como una interfaz que, mediante un Intent en Android Studio, lanza la actividad. En la Figura 2.41 se muestra el código donde cada botón está asociado al evento *click* que llevara al usuario a la acción que desee realizar.

```
buttonWrite.setOnClickListener((v) → {
        Intent i = new Intent( packageContext: Menu.this, Write.class);
        startActivity(i);
});

buttonInventario.setOnClickListener((v) → {
        Intent i = new Intent( packageContext: Menu.this, Inventario.class);
        startActivity(i);
});

buttonRead.setOnClickListener((v) → {
        Intent i = new Intent( packageContext: Menu.this, Read.class);
        startActivity(i);
});
```

Figura 2.41. Código - Interfaz Menú. Fuente: Autor

2.6.3 ACTIVIDAD ESCRIBIR ETIQUETA

Al elegir la opción *ESCRIBIR ETIQUETA* se muestra la interfaz que se mira en la Figura 2.42, aquí se muestran los campos a completarse con el correspondiente identificativo que serán escritos sobre la etiqueta y guardados en la base de datos.

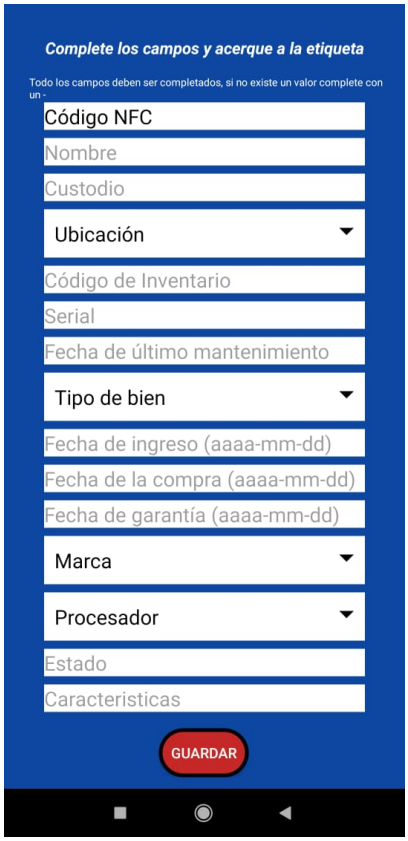


Figura 2.42. Interfaz Escribir Etiqueta. Fuente: Autor

En esta actividad se utiliza el archivo *wsJSONRegistro.php* para web service. Aquí se crean las credenciales para acceder a la base de datos; host, nombre, usuario y contraseña con los que se realiza la conexión con la base de datos. Se crea un objeto JSON array que se encarga de almacenar la información que retorna el web service. Se hace verificaciones de los datos que llegan al archivo PHP mediante el método *GET*, en este caso la información de cada bien, si los datos no son recibidos el web service devuelve que no retornó, esto puede ser en caso de que haya un problema de conexión. La información se almacena en variables locales para realizar un *INSERT* a la base de datos, luego se hace nuevamente una consulta a la tabla *products* donde se insertaron los datos, esta consulta es necesaria para verificar que efectivamente se realizó el registro, esta información se almacena en el array *JSON* que luego se codifica mediante *json_encode* para que se almacene en un

formato *JSON* y se devuelva la información. En la Figura 2.43 se muestra el código en PHP de este proceso.

```php
$conexion=new mysqli($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);

$insert="INSERT INTO products(name, codigo_nfc, serial, fecha_garantia, codigo_inventario, custodio, fecha_ingreso, ubicacion,fecha

if($conexion->query($insert)===TRUE){

    $resultado = $conexion->query("SELECT * FROM products WHERE codigo_nfc = {$codigo_nfc}");

    if($registro=mysqli_fetch_array($resultado)){
        $json['products'][]=$registro;
    }
    mysqli_close($conexion);
    echo json_encode($json);
}
else{
    echo "El error es : ".mysqli_error($conexion);
    $resulta["codigo_nfc"]=0;
    $resulta["name"]='No Registra';
    $json['products'][]=$resulta;
    echo json_encode($json);
}
```

Figura 2.43. Web Service - Añadir nuevo bien. Fuente: Autor

Para el código en Java de Android Studio, se implementan los métodos necesarios para trabajar con *Volley*. Se crea un objeto tipo *RequestQueue* y *JsonObjectRequest* propios de Volley que permiten establecer la conexión directamente con el web service. En el método *onNewIntent* se asocia el evento *click* con el botón *Guardar* donde se llama a los métodos *cargarWebService, createNdefMessage* y *writeNdefMessage,* como se mira en la Figura 2.44*.* Esto quiere decir que cuando el usuario presiona *Guardar* se realiza primero la inserción de los datos en la base de datos mediante *cargarWebService* y luego se realiza la escritura de la etiqueta mediante *createNdefMessage* y *writeNdefMessage* donde se escogen y se escriben los datos que van se van a guardar en la etiqueta que como se aclaró antes son los más importantes debido a la limitada memoria de la etiqueta NFC.

```java
protected void onNewIntent(final Intent intent) {
    super.onNewIntent(intent);

    if (intent.hasExtra(NfcAdapter.EXTRA_TAG)){
        Toast.makeText( context: this,  text: "NfcIntent", Toast.LENGTH_SHORT).show();

        btnWrite.setOnClickListener((v) -> {
                cargarWebService();
                Tag tag = intent.getParcelableExtra(NfcAdapter.EXTRA_TAG);
                NdefMessage ndefMessage = createNdefMessage( content: allcampos.getText()+ "");
                writeNdefMessage(tag, ndefMessage);

        });
    }
}
```

Figura 2.44. Método onNewIntent – Actividad Escribir. Fuente: Autor

En el método *cargarWebService* se define la URL del web service con la dirección IP del servidor con los parámetros necesarios para completar todos los datos de cada bien como se indica en la Figura 2.45. La URL se debe enviar a Volley para ser leída y procese la información que se debe enviar utilizando el objeto *JsonObjectRequest* indicando que se utiliza el método GET y la URL. Cuando se realizó exitosamente el registro se indica que la inserción se realiza correctamente y se limpian los campos en el layout. En el caso de no poder registrar se muestra el error en pantalla.

```
private void cargarWebService(){

    progreso=new ProgressDialog( context: this);
    progreso.setMessage("Cargando...");
    progreso.show();

    String url="http://192.168.1.12/appNFC/wsJSONRegistro.php?codigo_nfc="+ txtTagContent.getText().toString()+
            "&name="+editTextName.getText().toString()+
            "&custodio="+editTextCustodio.getText().toString()+
            "&ubicacion="+editTextUbicacion.getText().toString()+
            "&codigo_inventario="+editTextCodigoInventario.getText().toString()+
            "&serial="+editTextSerial.getText().toString()+
            "&fecha_ultimo_mantenimiento="+editTextDateMant.getText().toString()+
            "&fecha_ingreso="+editTextDateIngreso.getText().toString()+
            "&fecha_compra="+editTextDateCompra.getText().toString()+
            "&fecha_garantia="+editTextDateGar.getText().toString()+
            "&marca="+editTextMarca.getText().toString()+
            "&procesador="+editTextProcesador.getText().toString()+
            "&estado="+editTextEstado.getText().toString()+
            "&caracteristicas="+editTextCaracteristicas.getText().toString()+
            "&tipo="+editTextTipo.getText().toString();


    url=url.replace( target: " ", replacement: "%20");

    jsonObjectRequest=new JsonObjectRequest(Request.Method.GET,url,  jsonRequest: null, listener: this, errorListener: this);
    request.add(jsonObjectRequest);
```

Figura 2.45. Método cargarWebService – Actividad Escribir. Fuente: Autor

Los intentos de NFC llevan un valor adicional con ellos que es una representación virtual de la etiqueta y se puede obtener utilizando la clave *NfcAdapter.EXTRA_TAG*. Se puede obtener información sobre la etiqueta, como el ID de la etiqueta y su tipo de contenido, a través de este objeto.

En el método *onNewIntent*, se recupera la instancia de la etiqueta y luego se usa otras clases proporcionadas por Android para leer, escribir y recuperar fácilmente más información sobre la etiqueta. Estas clases son las siguientes:

- *android.nfc.tech.ndef*: esta clase proporciona métodos para recuperar y modificar el objeto *NdefMessage* en una etiqueta
- *android.nfc.tech.NdefFormatable:* esta clase proporciona métodos para formatear etiquetas que pueden formatearse como NDEF

Lo primero que se hace al escribir una etiqueta es llamar al método *get (Tag tag)* desde la clase *Ndef*, que devolverá una instancia de la misma clase. Luego, se necesita abrir una conexión con la etiqueta llamando al método *connect ().* Con una conexión abierta, ahora se puede escribir un mensaje NDEF en la etiqueta llamando al método *writeNdefMessage (NdefMessage ndefMessage)*. Comprobar si la etiqueta se puede escribir o no siempre es una buena práctica para evitar excepciones no deseadas. Se puede hacer esto llamando al método isWritable (). Cuando todo está hecho, se llama al método *close ()* para liberar la conexión previamente abierta. En la Figura 2.46 se mira el código de este proceso.

Si el método *get (Tag tag)* devuelve nulo, significa que la etiqueta no está formateada según el formato NDEF, y se debe formatearla correctamente.

```
private void writeNdefMessage(Tag tag, NdefMessage ndefMessage){
    try {
        if (tag == null){
            Toast.makeText( context: this,  text: "El objeto tag no puede ser nulo",Toast.LENGTH_SHORT).show();
            return;
        }

        Ndef ndef = Ndef.get(tag);

        if (ndef == null){
            //Formatear la etiqueta con el formato NDEF y escribir el mensaje
            formatTag(tag, ndefMessage);
        }else {
            ndef.connect();

            if (!ndef.isWritable()){
                Toast.makeText( context: this,  text: "La etiqueta no se puede escribir",Toast.LENGTH_SHORT).show();
                ndef.close();
                return;
            }

            ndef.writeNdefMessage(ndefMessage);
            ndef.close();

            Toast.makeText( context: this,  text: "La etiqueta puede ser escrita",Toast.LENGTH_SHORT).show();
        }

    }catch (Exception e){
        Log.e( tag: "writeNdefMessage",e.getMessage());
    }
}
```

Figura 2.46. Método writeNdefMessage. Fuente: Autor

Para formatear una etiqueta con el formato NDEF, se usa la clase *NdefFormatable* de la misma manera que se lo hace con la clase *Ndef*. Sin embargo, en este caso, se quiere formatear la etiqueta y escribir un mensaje. Esto se logra llamando al método *formatTag()* que se mira en la Figura 2.47. Entonces, se debe llamar al método *get (Tag tag)*, luego abrir una conexión llamando a connect (), formatear la etiqueta y escribir el mensaje llamando al método de formato *formatTag().* Finalmente, se cierra la conexión con el método close ().

Si el método *get (Tag tag)* devuelve nulo, significa que la API de Android no puede formatear automáticamente la etiqueta a NDEF.

```
private void formatTag (Tag tag , NdefMessage ndefMessage){
    try {
        NdefFormatable ndefFormatable = NdefFormatable.get(tag);

        if (ndefFormatable == null){
            Toast.makeText( context: this,  text: "La etiqueta no es formateable",Toast.LENGTH_SHORT).show();
            return;
        }

        ndefFormatable.connect();
        ndefFormatable.format(ndefMessage);
        ndefFormatable.close();

        Toast.makeText( context: this,  text: "Etiqueta escrita",Toast.LENGTH_SHORT).show();

    }catch (Exception e){
        Log.e( tag: "formatTag",e.getMessage());
    }
}
```

Figura 2.47. Método formatTag. Fuente: Autor

Para escribir un string en una etiqueta se debe tomar en cuenta que el payload no corresponde directamente con el texto que se escribe. El payload enviado a la etiqueta es una concatenación de bytes que contiene el lenguaje del string, el string mismo y un byte adicional, el primero, que es un operador AND bit a bit en el tamaño del lenguaje (para indicar la codificación de caracteres utilizada). Esto se hace con el método *createTextRecord* que se visualiza en la Figura 2.48.

```
private NdefRecord createTextRecord(String content)
{
    try {
        byte[] language;
        language = Locale.getDefault().getLanguage().getBytes( charsetName: "UTF-8");

        final byte[] text = content.getBytes( charsetName: "UTF-8");
        final int languageSize = language.length;
        final int textLength = text.length;
        final ByteArrayOutputStream payload = new ByteArrayOutputStream( size: 1 + languageSize + textLength);

        payload.write((byte) (languageSize & 0x1F));
        payload.write(language,  off: 0, languageSize);
        payload.write(text,  off: 0, textLength);

        return new NdefRecord(NdefRecord.TNF_WELL_KNOWN, NdefRecord.RTD_TEXT, new byte[0], payload.toByteArray());

    }catch (UnsupportedEncodingException e){
        Log.e( tag: "createTextRecord", e.getMessage());
    }
    return null;
}
```

Figura 2.48. Método createTextRecord [16]

El método *createNdefMessage* que se muestra en la Figura 2.49 utiliza el método *createTextRecord* para crear el mensaje NDEF que será escrito con el método *writeNdefMessage.*

```
private NdefMessage createNdefMessage(String content){

    NdefRecord ndefRecord = createTextRecord(content);

    NdefMessage ndefMessage = new NdefMessage(new NdefRecord[]{ ndefRecord });

    return ndefMessage;
}
```

Figura 2.49. Método createNdefMessage. Fuente: Autor

Cuando se lee el texto de una etiqueta, se necesita realizar la operación inversa y obtener el lenguaje del string y el string mismo de la matriz de bytes.

En la aplicación, se usa el lenguaje predeterminado del sistema para formatear el string, pero se puede usar cualquier otro lenguaje independiente del lenguaje del usuario.

En la Figura 2.50 se muestra el diagrama de flujo de esta actividad.

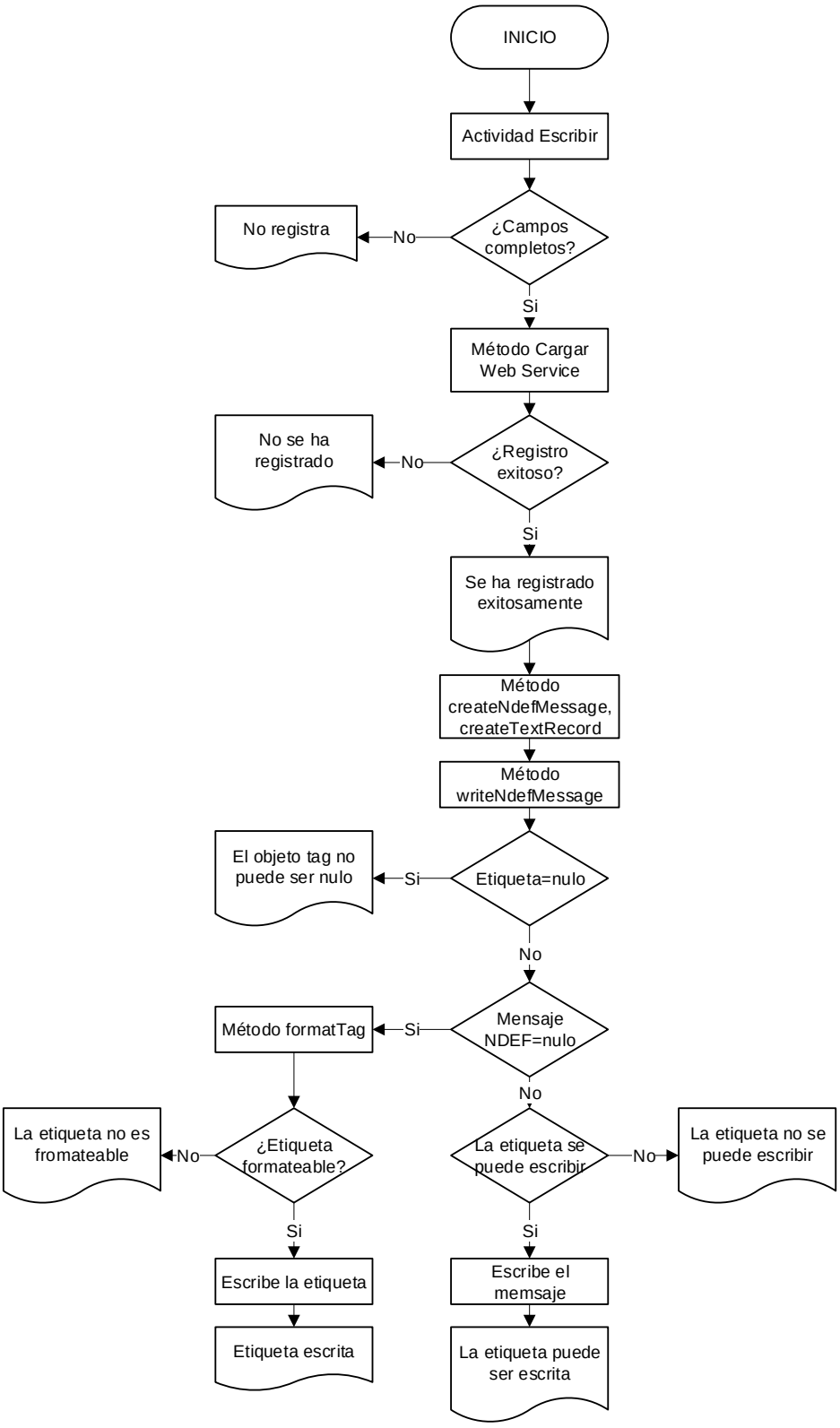


Figura 2.50. Diagrama de Flujo - Actividad Escribir Etiqueta. Fuente: Autor

2.6.4 ACTIVIDAD LEER ETIQUETA

En la interfaz de la actividad Leer Etiqueta en cada uno de los espacios se mostrarán los campos de la información de cada bien asociados al identificador del bien, como se visualiza en la Figura 2.51.

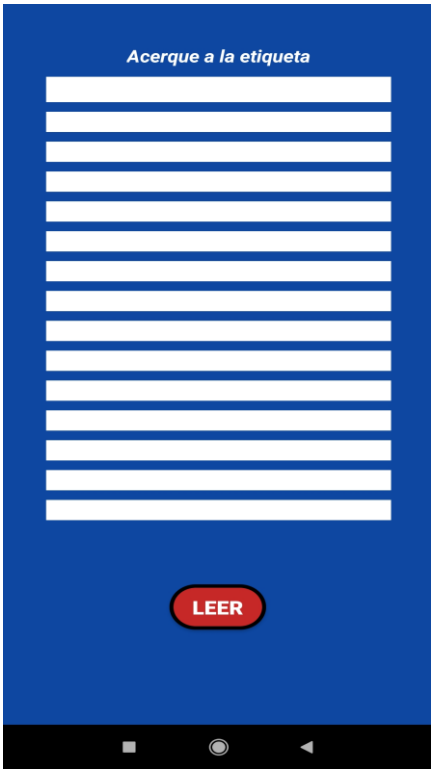


Figura 2.51. Interfaz Leer Etiqueta. Fuente: Autor

Para esta actividad se utiliza el archivo para web service, *wsJSONConsultarbien.php* donde se consulta la información registrada en la base de datos para presentarla en la pantalla de la aplicación. Este archivo es similar al utilizado para el registro de un bien solo que en lugar de un *INSERT* se hace la consulta directa en la tabla correspondiente. Se tiene las credenciales de acceso a la base de datos, se recibe un parámetro que sería el código NFC, se conecta a la base de datos, se realiza la consulta que devuelve todos los campos del bien asociado al código NFC y con base a esto se crea el objeto JSON. El código correspondiente se visualiza en la Figura 2.52.

```php
$json=array();

    if(isset($_GET["codigo_nfc"])){
        $codigo_nfc=$_GET["codigo_nfc"];

        $conexion = mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);

        $consulta="select name, codigo_nfc, serial, fecha_garantia, codigo_inventario, custodio, fecha_ingreso, ubicacion,
        $resultado=mysqli_query($conexion,$consulta);

        if($registro=mysqli_fetch_array($resultado)){
            $json['products'][]=$registro;
        }else{
            $resultar["codigo_nfc"]=0;
            $resultar["name"]='no registra';
            $json['products'][]=$resultar;
        }

        mysqli_close($conexion);
        echo json_encode($json);
    }
    else{
        $resultar["success"]=0;
        $resultar["message"]='Ws no Retorna';
        $json['products'][]=$resultar;
        echo json_encode($json);
    }
?>
```

Figura 2.52. Web Service - Consultar Bien. Fuente: Autor

Para el código fuente en Java se utilizan los objetos *RequestQueue* y *JsonObjectRequest*. En el método *onCreate* se hace referencia al evento *click* con el botón LEER en el que se llama a los métodos *readTextFromMessage* y *cargarWebService* como se mira en la Figura 2.53. El método *readTextFromMessage* lee la etiqueta escoge el valor del campo código NFC y este valor se busca en a base de datos mediante el método *cargarWebService.*

```java
protected void onNewIntent(final Intent intent) {
    super.onNewIntent(intent);

    if (intent.hasExtra(NfcAdapter.EXTRA_TAG)){
        Toast.makeText( context: this,  text: "NfcIntent", Toast.LENGTH_SHORT).show();

        buttonReadTag.setOnClickListener((v) -> {
                Parcelable[] parcelables = intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
                if (parcelables !=null && parcelables.length > 0){
                    readTextFromMessage((NdefMessage)parcelables[0]);
                    cargarWebService();
                }else {
                    Toast.makeText( context: Read.this,  text: "Ningún mensaje NDEF encontrado", Toast.LENGTH_SHORT).show();
                }
        });
    }
}
```

Figura 2.53. Método onNewIntent - Actividad Leer. Fuente: Autor

En el método cargarWebService que se visualiza en la Figura 2.54 se define un string que tenga la URL para el llamado al web service. En el objeto *jsonObjectRequest* se envían los parámetros necesarios, estos son el método GET y la URL y al RequestQueue se agrega este objeto *JSON*. En caso de error se muestra el error y en el caso de éxito se muestra la

consulta. Para presentar en pantalla la información se crea una clase Java llamada campos, esta clase va a contener la información de los bienes donde se generan los métodos get y set con los que se puede mapear la información para presentarla en pantalla. Los métodos get y set, son métodos que se usan en las clases para mostrar (get) o modificar (set) el valor de un atributo.

```
private void cargarWebService(){
    progreso=new ProgressDialog( context: this);
    progreso.setMessage("Consultando...");
    progreso.show();
    String url = "http://192.168.1.12/appNFC/wsJSONConsultarBien.php?codigo_nfc="+codigo.getText().toString();

    jsonObjectRequest=new JsonObjectRequest(Request.Method.GET,url, jsonRequest: null, listener: this, errorListener: this);
    request.add(jsonObjectRequest);
}
```

Figura 2.54. Método cargarWebService - Actividad Leer

Se utiliza un objeto JSONArray para recibir el identificador de lo que retorna el web service. Se crea un objeto JSONObject que se encarga de llenar cada objeto dependiendo de los campos requeridos, luego se recorre el array para asignar cada dato como se mira en la Figura 2.55.

```
public void onResponse(JSONObject response) {
    progreso.hide();
    Toast.makeText( context: this, text: "Mensaje: "+response,Toast.LENGTH_SHORT).show();

    Campos campo =new Campos();

    JSONArray json=response.optJSONArray( name: "products");
    JSONObject jsonObject=null;

    try {
        jsonObject=json.getJSONObject( index: 0);
        campo.setTxtname(jsonObject.optString( name: "name"));
        campo.setTxtserial(jsonObject.optString( name: "serial"));
        campo.setTxtfecha_garantia(jsonObject.optString( name: "fecha_garantia"));
        campo.setTxtcodigo_inventario(jsonObject.optString( name: "codigo_inventario"));
        campo.setTxtcustodio(jsonObject.optString( name: "custodio"));
        campo.setTxtfecha_ingreso(jsonObject.optString( name: "fecha_ingreso"));
        campo.setTxtubicacion(jsonObject.optString( name: "ubicacion"));
        campo.setTxtfecha_ultimo_mantenimiento(jsonObject.optString( name: "fecha_ultimo_mantenimiento"));
        campo.setTxttipo(jsonObject.optString( name: "tipo"));
        campo.setTxtfecha_compra(jsonObject.optString( name: "fecha_compra"));
        campo.setTxtmarca(jsonObject.optString( name: "marca"));
        campo.setTxtprocesador(jsonObject.optString( name: "procesador"));
        campo.setTxtestado(jsonObject.optString( name: "estado"));
```

Figura 2.55. Código para obtener los datos de la consulta. Fuente: Autor

El sistema Android usa Intents para transportar datos entre actividades, aplicaciones y eventos. La misma lógica se aplica en NFC, en la detección de una etiqueta, la instancia de la etiqueta se puede obtener con el Intent. Los mensajes NDEF también se pueden obtener desde allí.

Un mensaje NDEF contiene uno o más registros Ndef (NdefRecords). Ya que los datos reales están en NdefRecords se debe obtener el primer NdefRecord de NdefMessage y su contenido. Como el sistema Android usa los primeros registros para determinar el tipo de etiqueta; se supone que es del primer registro del que se quiere obtener los datos. Entonces, en el método *readTextFromMessage,* solo se devuelve el primer registro encontrado. Los datos en sí pueden obtenerse a través del método getPayload presente en la instancia del mensaje NDEF. Este método devuelve el byte [ ], que es una representación sin formato de los datos. Este proceso se puede mirar en la Figura 2.56.

```
private void readTextFromMessage(NdefMessage ndefMessage) {

    NdefRecord [] ndefRecords = ndefMessage.getRecords();

    if(ndefRecords != null && ndefRecords.length>0){
        NdefRecord ndefRecord = ndefRecords[0];

        String tagContent = getTextFromNdefRecord(ndefRecord);

        txtTagContent.setText(tagContent);

    }else{
        Toast.makeText( context: this,  text: "Ningún registro NDEF encontrado", Toast.LENGTH_SHORT).show();
    }

    String total =txtTagContent.getText().toString();
    String[] parte = total.split( regex: ",");

    parte1=parte[0];
    codigo.setText(" "+parte1);
}
```

Figura 2.56. Método readTextFromMessage

Como se vio en la Actividad Escribir, escribir un string requiere que se cree una matriz de bytes, que contenga el bit de estado que indica la codificación del texto, la longitud del código del lenguaje y el string en sí. Para tomar un string del array bytes [ ], se necesita hacer exactamente lo contrario y separar la matriz. Se hace esto en el método *getTextFromNdefRecord.* Primero, se obtiene el tamaño del lenguaje y se determina la codificación correcta que se utilizará. La información del lenguaje está presente en el mensaje porque es posible que se necesite formatear correctamente el texto en el lenguaje en que fue escrito. El método *getTextFromNdefRecord* se mira en la Figura 2.57.

```
public String getTextFromNdefRecord (NdefRecord ndefRecord){
    String tagContent = null;
    try {
        byte[] payload = ndefRecord.getPayload();
        String textEncoding = ((payload[0] & 128) == 0) ? "UTF-8" : "UTF-16";
        int languageSize = payload[0] & 0063;
        tagContent = new String(payload,  offset: languageSize + 1,
                length: payload.length - languageSize - 1, textEncoding);
    }catch (UnsupportedEncodingException e ){
        Log.e( tag: "getTextFromNdefRecord", e.getMessage(),e);
    }
    return tagContent;
}
```

Figura 2.57. Método getTextFromNdefRecord [16]

En el diagrama de flujo de la Figura 2.58 se muestra el proceso de la actividad Leer Etiqueta.

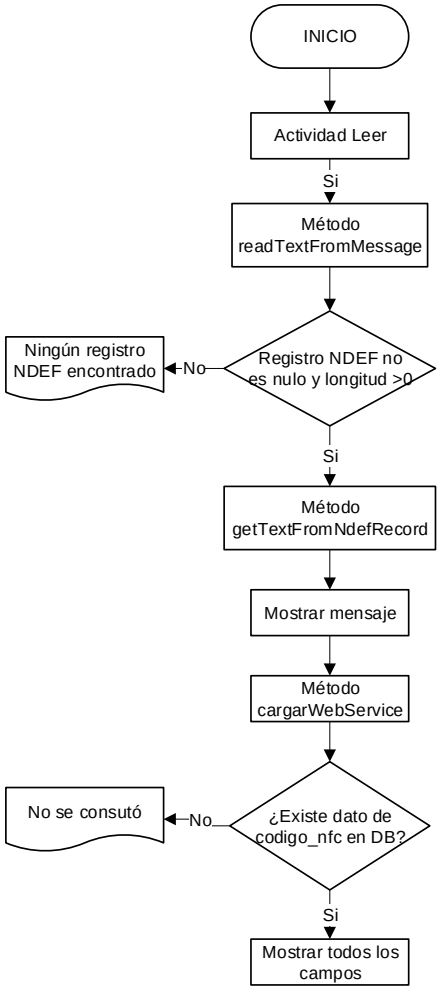


Figura 2.58. Diagrama de Flujo - Actividad Leer Etiqueta. Fuente: Autor

2.6.5 ACTIVIDAD INVENTARIO

En esta interfaz se visualiza el inventario completo que se encuentra en la base de datos. (Figura 2.59). Para el layout de esta actividad se usa el componente RecyclerView, este se utiliza para mostrar una lista de desplazamiento de elementos de grandes conjuntos de datos.[45]

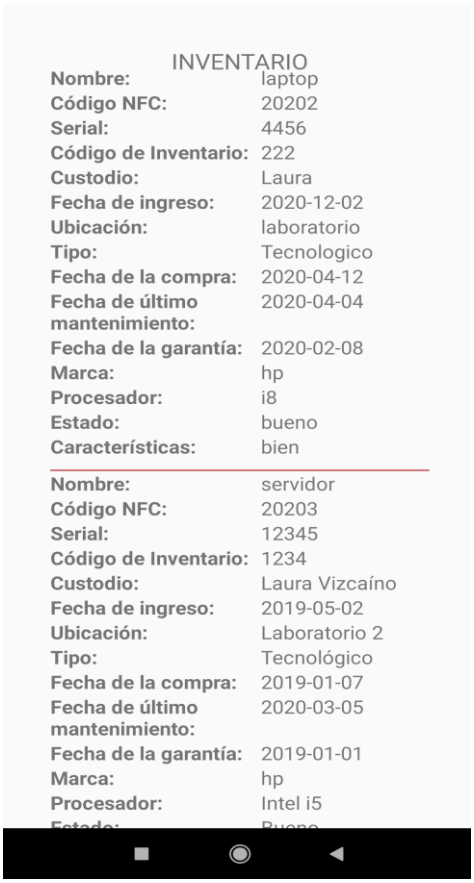


Figura 2.59. Interfaz Inventario. Fuente: Autor

Para esta actividad se utiliza el archivo para web service, *wsJSONConsultarLista.php*. Este archivo tiene la misma estructura de las actividades anteriores, la diferencia aquí es que se llama directamente a la tabla *products* para que devuelva todos los campos, como se mira en la Figura 2.60.

```php
$json=array();

    $conexion = mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$database_localhost);

    $consulta="select name, codigo_nfc, serial, fecha_garantia, codigo_inventario, custodio, fecha_ingreso, ubicacion,fecha_ultimo
    $resultado=mysqli_query($conexion,$consulta);

    while($registro=mysqli_fetch_array($resultado)){
        $json['products'][]=$registro;
    }
    mysqli_close($conexion);
    echo json_encode($json);
?>
```

Figura 2.60. Web Service - Consultar Inventario. Fuente: Autor

Para el código en Java en el método onCreate se llama al método cargarWebService que se mira en la Figura 2.61 y en donde se hace la conexión con el web service mediante un string que identifique la URL del servidor. Con el objeto JsonObjectRequest se envía los parámetros necesarios para la conexión, el método GET y la URL. Si existe un error, este se muestra en pantalla, en caso contrario se muestran todos los campos de cada bien en pantalla.

```java
private void cargarWebService() {

    progress=new ProgressDialog( context: this);
    progress.setMessage("Consultando...");
    progress.show();

    String url = "http://192.168.1.12/appNFC/wsJSONConsultarLista.php";

    jsonObjectRequest= new JsonObjectRequest(Request.Method.GET,url, jsonRequest: null, listener: this, errorListener: this);
    request.add(jsonObjectRequest);
}
```

Figura 2.61. Método cargarWebService - Actividad Inventario

2.6.6 ACTIVIDAD LECTURA

Para el caso de la Actividad Leer Etiqueta y acceder a esta es necesario el uso de las credenciales y por consiguiente acceso a Internet. Se desarrolla esta actividad en el escenario de que el usuario solo desee mirar la información escrita en la etiqueta y que pueda hacerlo sin acceso a Internet. En la Figura 2.62 se muestra la interfaz de esta actividad.

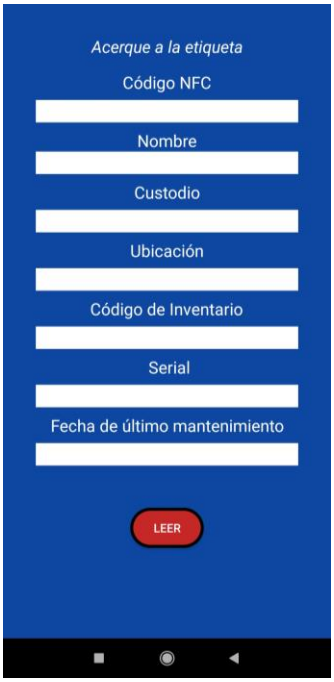


Figura 2.62. Interfaz Lectura

El código fuente de esta actividad utiliza los mismos métodos de la actividad Leer Etiqueta, excepto el método cargarWebService ya que no se requiere acceso al servidor. En la Figura 2.63 se muestra el diagrama de flujo de esta actividad.

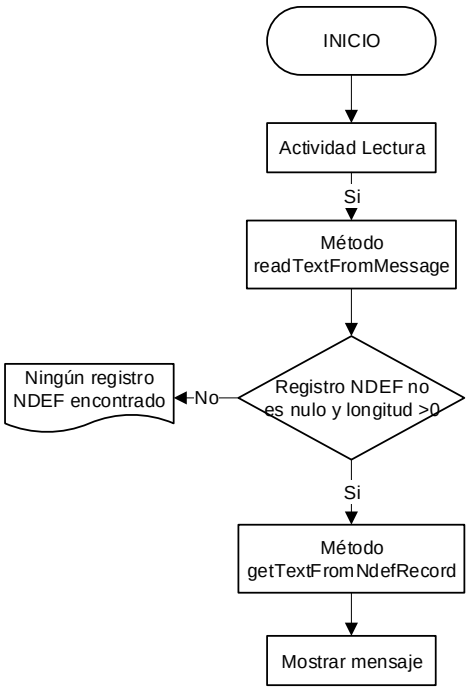


Figura 2.63. Diagrama de flujo - Actividad Lectura

En la Figura 2.64 se muestra el diagrama de clases de la aplicación móvil.

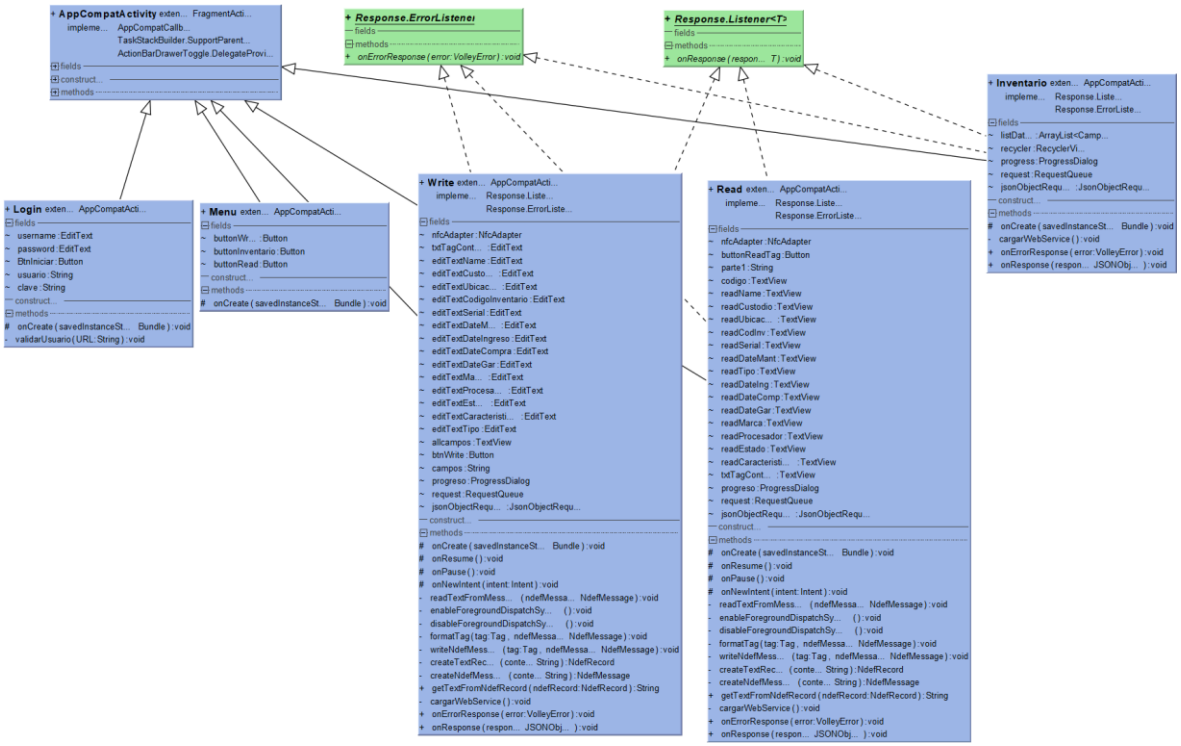


Figura 2.64. Diagrama de clases de la aplicación móvil. Fuente: Android Studio

3. PRUEBAS Y RESULTADOS

En este capítulo se muestran las pruebas realizadas con el prototipo implementado, así como los resultados obtenidos de las mismas.

3.1 COMPONENTES DE HARDWARE Y SOFTWARE IMPLEMENTADOS

Los componentes de hardware y software implementados para las pruebas de funcionamiento se detallan en la Tabla 3.1.

Tabla 3.1. Características de hardware y software implementados. Fuente: Autor

Equipo/Software	Características	
Servidor Linux	Sistema Operativo	Ubuntu – 16.04 – amd64
	RAM, MiB	1024
	Tamaño disco, MiB	30720
	PHP, versión	7.3.14-6
	MySQL, versión	14.14 Distrib 5.7.29

	GIT, versión	2.7.4
	Plataforma	NGINX
Entorno de Desarrollo Integrado	Android Studio, versión	3.6
Editor de Texto	Microsoft Visual Studio Code, versión	1.44.2
Etiquetas NFC	NTAG215, NTAG213	
Terminal celular	Marca	Xiaomi
	Modelo	Mi 9T
	Versión de Android	10
	Nivel de API	29

Las pruebas se realizan con el terminal descrito en la tabla, sin embargo, la aplicación es funcional con cualquier dispositivo que tenga la tecnología NFC incorporada y su mínimo nivel de API sea 21 esto es Android 5.0 (Lollipop).

En la Figura 3.1 se mira las etiquetas NTAG215 y en la Figura 3.2 se observa las etiquetas NTAG213 antimetal, las que se utilizan para las pruebas de funcionamiento.

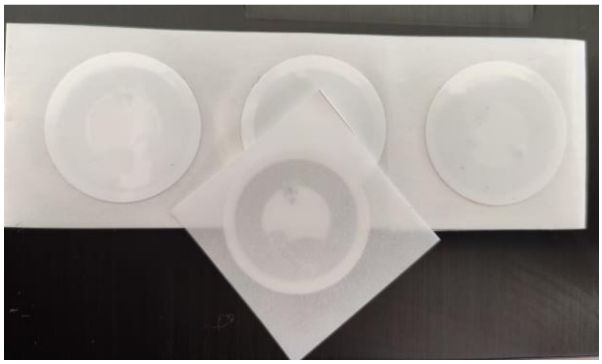


Figura 3.1. Etiquetas NTAG215

Figura 3.2. Etiquetas NTAG213 antimetal

3.2 ESCENARIO DE PRUEBAS

Las pruebas se realizan en un ambiente con las siguientes características:

- La aplicación móvil debe estar correctamente instalada en el celular.
- La aplicación web y la aplicación móvil necesitan tener disponibilidad de Internet.
- Las etiquetas NTAG213 antimetal deben colocarse sobre las superficies metálicas.
- Las etiquetas NTAG215 no deben colocarse sobre superficies metálicas.
- Las etiquetas utilizadas deben ser compatibles con Android.
- El celular que se utiliza debe tener incorporada la tecnología NFC.

3.3 PRUEBAS DE FUNCIONALIDAD

A continuación, se presentan los resultados de las acciones de la aplicación web y la aplicación móvil.

3.3.1 PRUEBAS DE APLICACIÓN WEB

3.3.1.1 Añadir un Nuevo Grupo

Una de las acciones a realizarse en la aplicación web como se explicó en el capítulo dos es añadir un nuevo grupo, aquí existen tres escenarios que se describen a continuación.

Primero, si en Nivel se completa con uno que ya existe como se muestra en la Figura 3.3, la aplicación devolverá un mensaje de que el nivel de grupo ya existe y no se registra. El campo nivel se asocia a los usuarios y define lo que pueden visualizar y administrar.

Figura 3.3. Añadir grupo (Escenario 1). Fuente: Autor

Segundo, cuando no se completan los campos, la aplicación devuelve que los campos no pueden estar vacíos como se mira en la Figura 3.4.

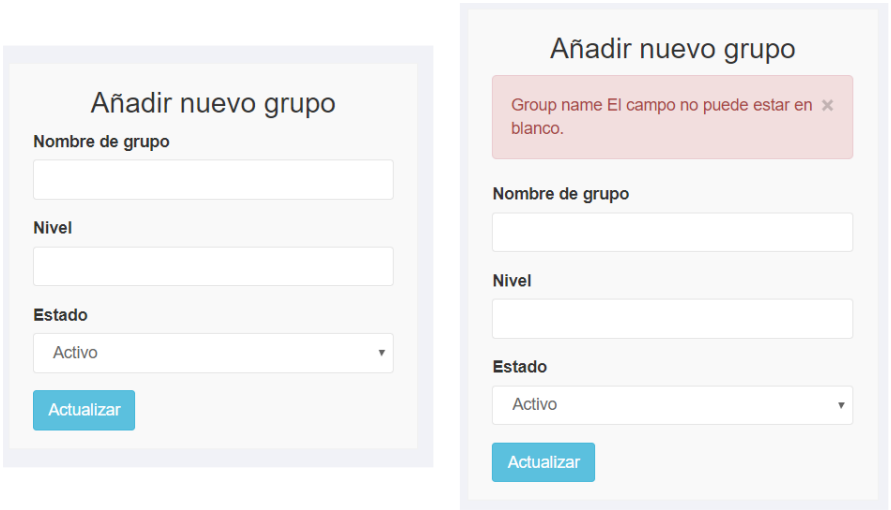


Figura 3.4. Añadir grupo (Escenario 2). Fuente: Autor

Tercero, cuando se añaden los campos correctamente, se registra con éxito en la base de datos el nuevo grupo como se puede ver en la Figura 3.5. Luego se limpian los campos y se redirige a la misma página.

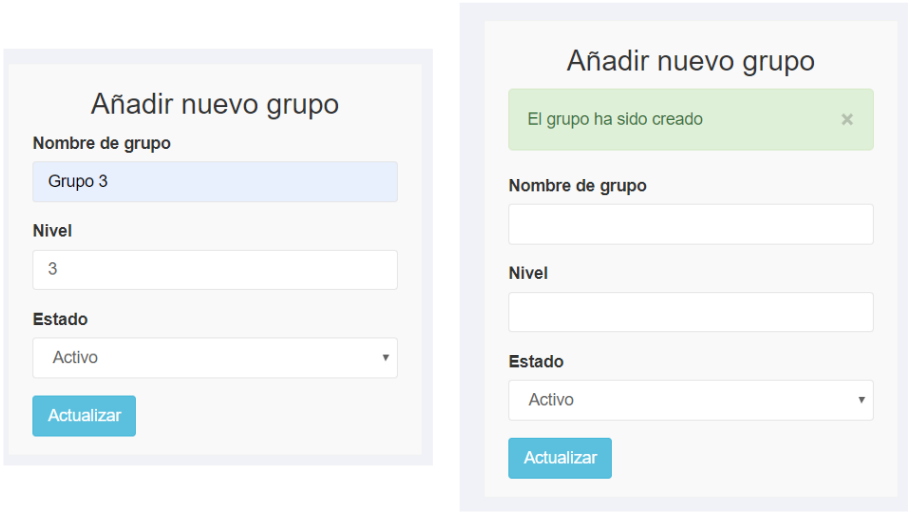


Figura 3.5. Añadir grupo (Escenario 3). Fuente: Autor

Los cambios realizados se miran en la página Usuarios – Administrar grupos con lo que se comprueba que se realizó correctamente la acción como se mira en la Figura 3.6. Aquí también se puede editar o eliminar los grupos, conforme se requiera.

Figura 3.6. Grupo añadido. Fuente: Autor

3.3.1.2 Añadir un Nuevo Usuario

En la aplicación web también se puede administrar a los usuarios, agregar un nombre, un usuario y una contraseña que es son los campos que se utilizan para ingresar al sistema, y el rol de usuario, es decir a que grupo pertenece.

Aquí existen dos escenarios, primero si los campos no están completos, la aplicación web devuelve que no pueden estar en blanco y no se registran (Figura 3.7).

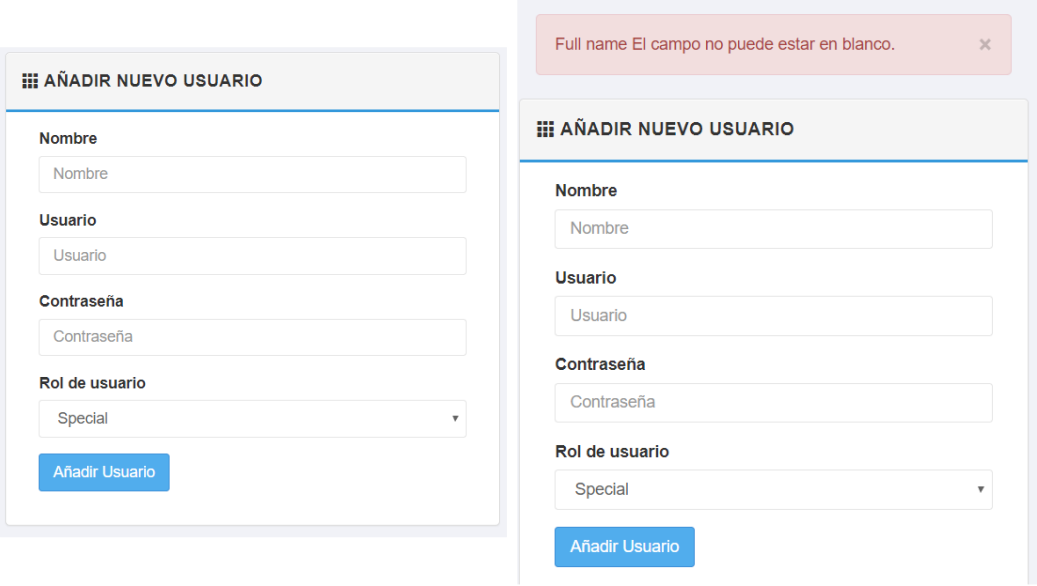


Figura 3.7. Añadir usuario (Escenario 1). Fuente: Autor

Segundo, si todos los campos están completos se registra con éxito el nuevo usuario en la base de datos, y se limpian los campos como se mira en la Figura 3.8.

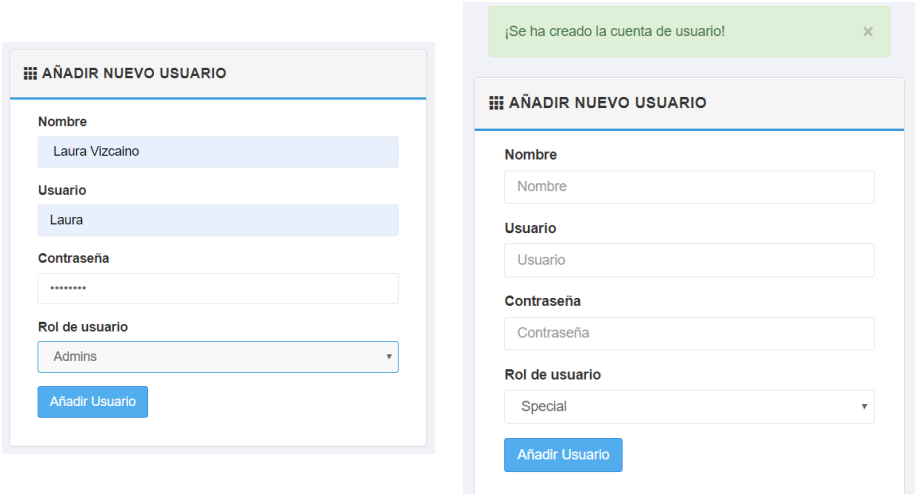


Figura 3.8. Añadir usuario (Escenario 2). Fuente: Autor

La correcta inserción del nuevo usuario se pude comprobar en la página Usuarios – Administrar usuarios como se visualiza en la Figura 3.9. Aquí también el administrador puede eliminar o editar la información de cada usuario.

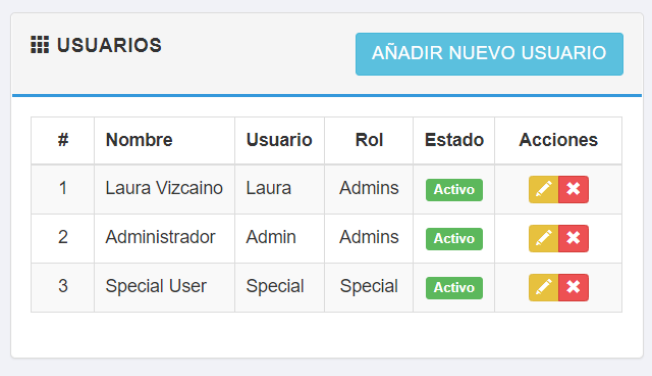


Figura 3.9. Usuario añadido. Fuente: Autor

3.3.1.3 Añadir Nuevo Bien

La aplicación web también tiene la función de añadir un nuevo bien, como se explicó en el capítulo dos, esta parte es funcional para bienes que no necesiten una etiqueta NFC. Todos los campos están disponibles para completarse, sin embargo, si no existe un dato de alguno se puede dejar en blanco, excepto los más importantes como Nombre, Tipo de Bien, Código de Inventario, Nombre del Custodio, Ubicación y Fecha de Ingreso.

Existe dos escenarios, el primero, si no se completa un campo importante como se muestra en la Figura 3.10, la aplicación devuelve que no se puede registrar.

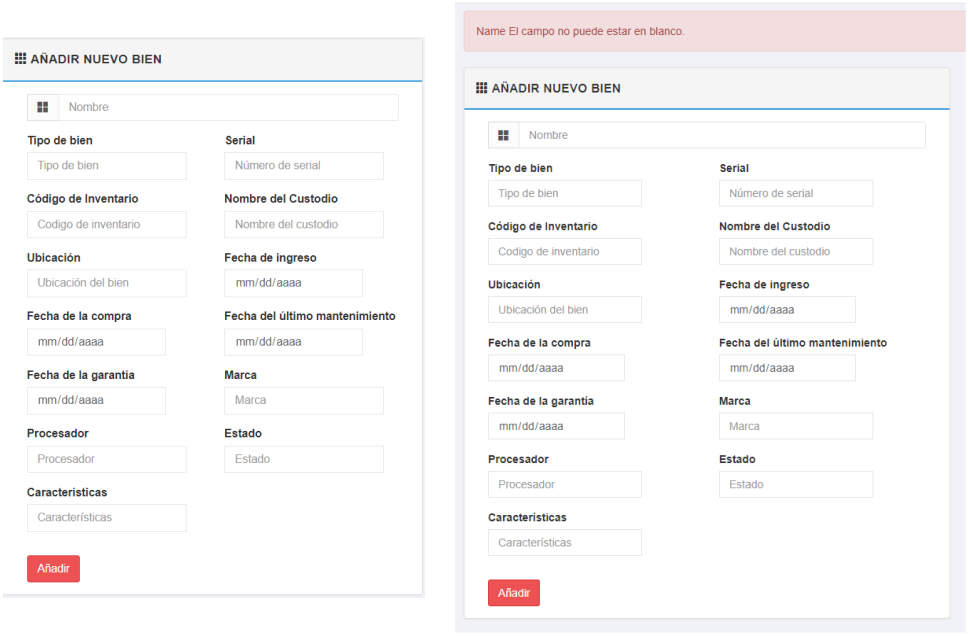


Figura 3.10. Añadir bien (Escenario 1). Fuente: Autor

En el segundo caso, si se completan todos los campos importantes se registra exitosamente el nuevo bien, la aplicación devuelve un mensaje de Añadido y se limpian los campos, como se muestra en la Figura 3.11.

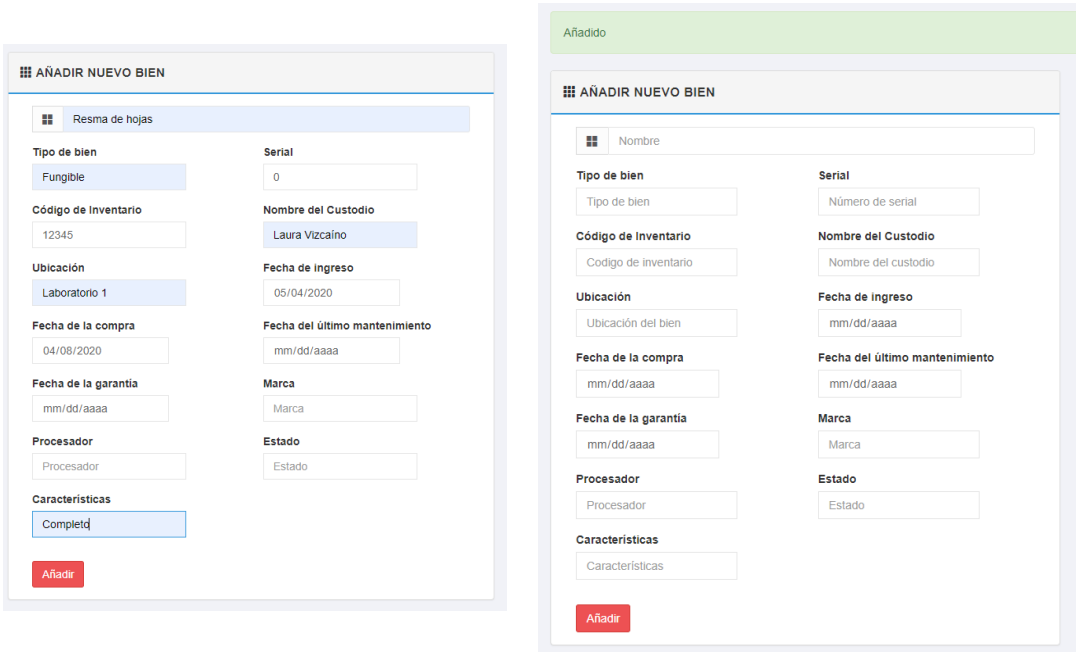


Figura 3.11. Añadir bien (Escenario 2). Fuente: Autor

Se puede comprobar el registro mirando la página Bienes- Administrar Bienes donde aparece el nuevo bien como se aprecia en la Figura 3.12. Aquí se visualiza una lista de todo el inventario como de los bienes que tienen un código NFC y los que no, igualmente se puede eliminar y editar cada ítem de la lista.

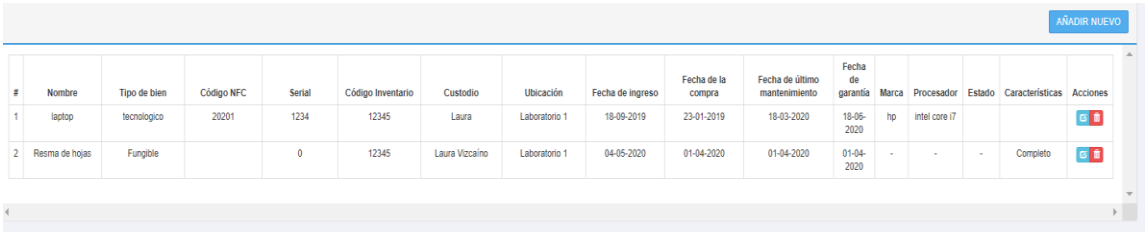


Figura 3.12. Bien añadido. Fuente: Autor

3.3.1.4 Editar Cuenta

En la aplicación también se puede editar la cuenta, el nombre y el usuario, como se mira en la Figura 3.13, si un campo no está completo la aplicación no actualiza y muestra el mensaje que el campo no puede estar en blanco.

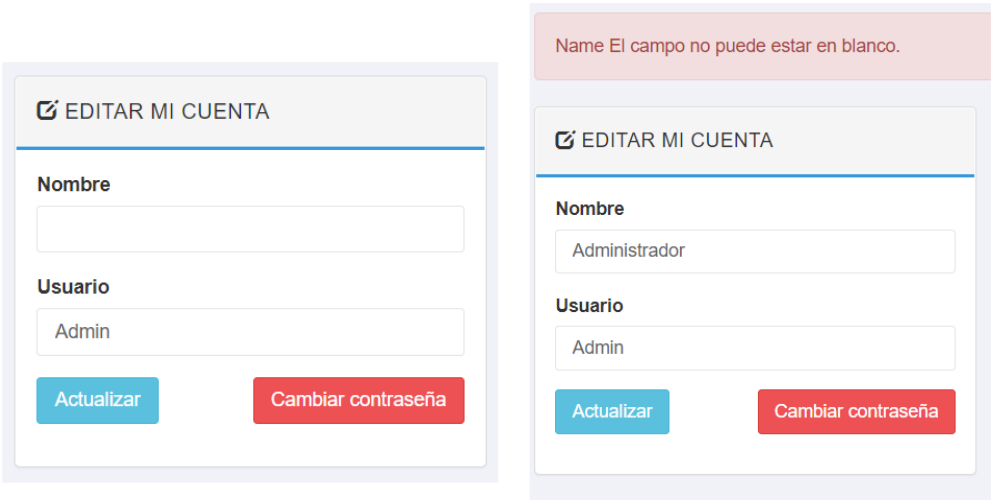


Figura 3.13. Editar cuenta. Fuente: Autor

Si los campos están completos los registros se actualizan correctamente como se indica en la Figura 3.14.

Figura 3.14. Cuenta actualizada. Fuente: Autor

También se puede actualizar la contraseña, como se mira en la Figura 3.15, si en el campo Anterior contraseña no coincide, no se actualiza. Si la contraseña es la correcta, se cambia la contraseña y la aplicación le redirige a la página de inicio para iniciar sesión con la nueva contraseña.

Figura 3.15. Cambiar contraseña. Fuente: Autor

3.3.2 PRUEBAS DE APLICACIÓN MÓVIL

En la Figura 3.16 se mira la aplicación instalada en el terminal móvil.

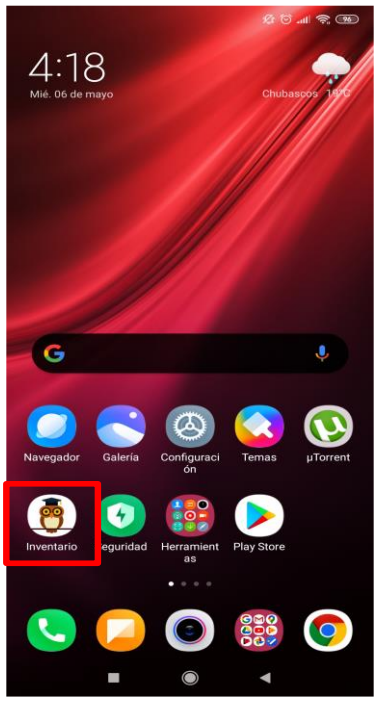


Figura 3.16. Aplicación instalada en Xiaomi Mi 9T

3.3.2.1 Login

En la Figura 3.17 se muestra la primera interfaz al acceder a la aplicación.

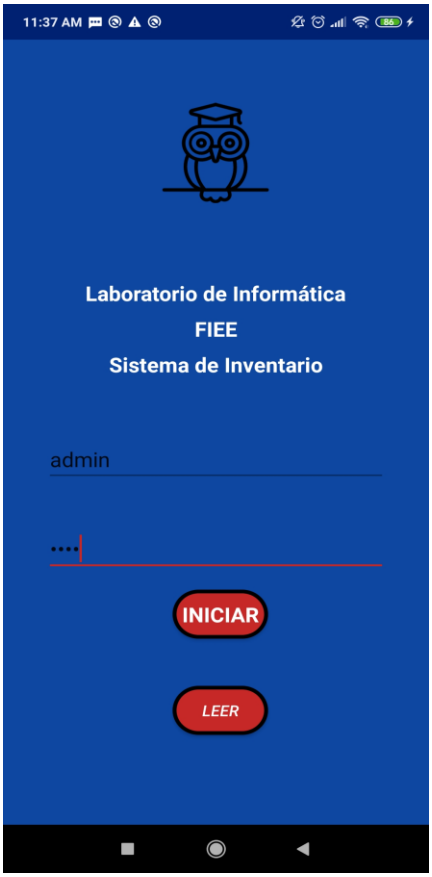


Figura 3.17. Interfaz de inicio

Para el acceso se debe utilizar el botón INICIAR, se usan las mismas credenciales que en la aplicación web ya que se trabaja con la misma base de datos. En el caso que se especifiquen credenciales incorrectas la aplicación devolverá un mensaje de usuario y contraseña incorrectas (Figura 3.18) y no se podrá acceder al menú.

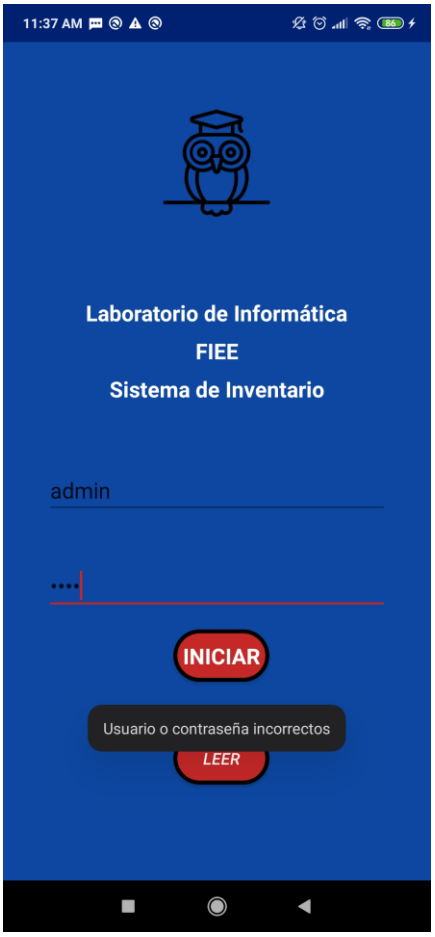


Figura 3.18. Credenciales incorrectas. Fuente: Autor

Como se mira en la Figura 3.17, el botón LEER, sirve para mirar la información que está escrita en la etiqueta si no hay acceso a Internet. Se hace la prueba con un bien ya ingresado, en la Figura 3.19 se mira el resultado. Si no existe un mensaje en la etiqueta no se muestra nada.

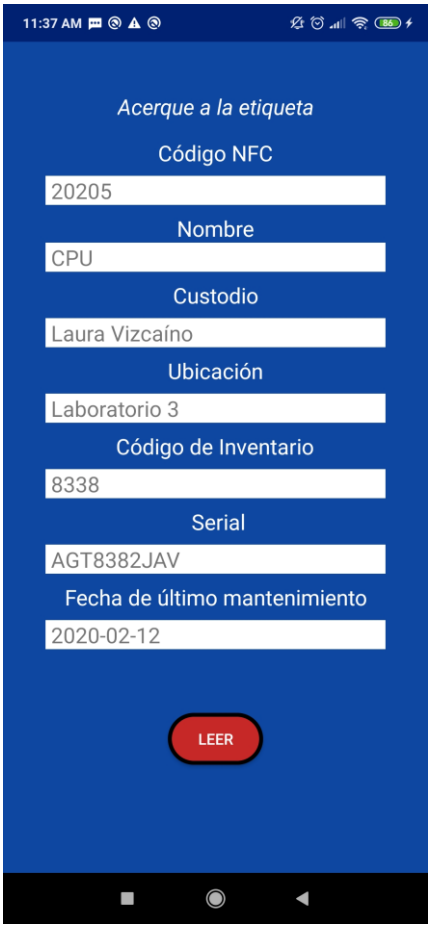


Figura 3.19. Información escrita en el tag Fuente: Autor

3.3.2.2 Escribir Etiqueta

Para el caso de añadir un nuevo bien mediante la aplicación móvil se debe escoger en el menú la opción ESCRIBIR ETIQUETA, como se indica en la interfaz (Figura 2.40).

En la Figura 3.20 se muestra el proceso de la actividad, primero se completan los campos donde en cada caso se indica su nombre, al acercar a la etiqueta se muestra un mensaje de etiqueta encontrada, como se mira en la primera pantalla de la Figura. Al presionar GUARDAR la aplicación devuelve *Mensaje guardado*, lo que quiere decir que la etiqueta está escrita, y otro mensaje *Cargando,* mientras se comunica con el servidor; como se visualiza en la segunda pantalla de la figura. Finalmente, cuando se realizó el registro del nuevo bien se muestra otro mensaje de que se registró correctamente, como se muestra en la tercera pantalla de la figura.

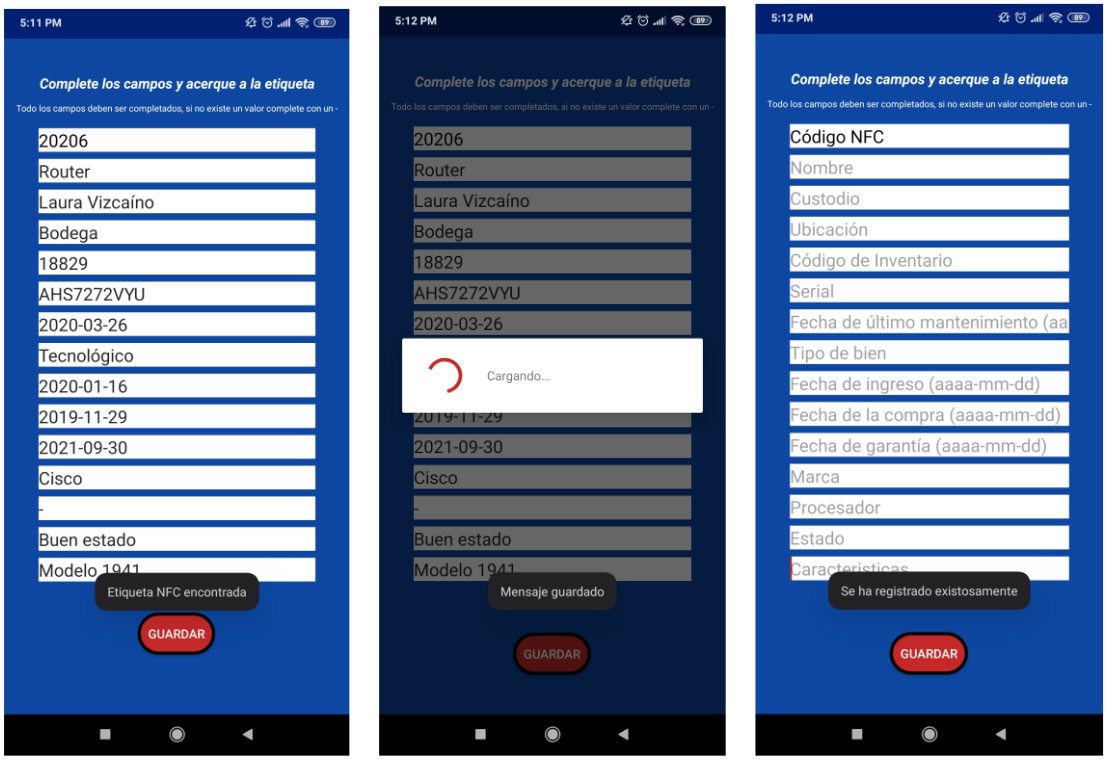


Figura 3.20. Se añade un nuevo bien desde la aplicación móvil. Fuente: Autor

Si en el menú se selecciona INVENTARIO, se podrá mirar el nuevo bien añadido, como se muestra en la Figura 3.21. Igualmente, el nuevo registro también se muestra en la aplicación web en la página Bienes - Administrar Bienes, como se mira en la Figura 3.22.

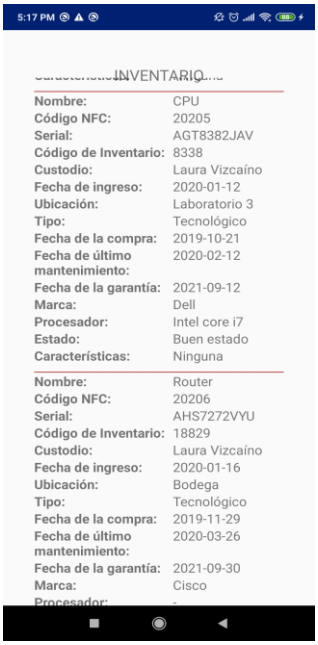


Figura 3.21. Inventario en la aplicación móvil Fuente: Autor

#	Nombre	Tipo de bien	Código NFC	Serial	Código Inventario	Custodio	Ubicación	Fecha de ingreso	Fecha de la compra	Fecha de último mantenimiento	Fecha de garantía	Marca	Procesador	Estado	Características	Acciones
1	Resma de hojas	Fungible	1234	0	12345	Laura Vizcaíno	Laboratorio 1	04-05-2020	01-04-2020	01-04-2020	01-04-2020	-	-	-		
2	Laptop	Tecnológico	20202	JHU7373HYD	12345	Laura Vizcaíno	Laboratorio 1	15-01-2020	12-05-2019	12-03-2020	13-05-2021	HP	Intel core i7	Buen estado	Ninguno	
3	CPU	Tecnológico	20203	HYT1635KIO	1627	Laura Vizcaíno	Bodega	25-10-2019	13-08-2019	12-01-2020	12-11-2021	ASUS	Intel core i7	Buen estado	Ninguna	
4	Switch	Tecnológico	20204	KSK828KJ	1882	Laura Vizcaíno	Laboratorio 2	03-02-2020	12-05-2019	03-02-2020	06-05-2021	Cisco	-	Buen estado	Ninguna	
5	CPU	Tecnológico	20205	AGT8382JAV	8338	Laura Vizcaíno	Laboratorio 3	12-01-2020	21-10-2019	12-02-2020	12-09-2021	Dell	Intel core i7	Buen estado	Ninguna	
6	Router	Tecnológico	20206	AHS7272VYU	18829	Laura Vizcaíno	Bodega	16-01-2020	29-11-2019	26-03-2020	30-09-2021	Cisco	-	Buen estado	Modelo 1941	

Figura 3.22. Inventario en la aplicación web

En el caso de que el usuario no complete todos los campos la aplicación devuelve un mensaje de que el bien no se ha registrado como se muestra en la Figura 3.23. La aplicación no permite agregar un bien si se ingresa un código NFC ya existente.

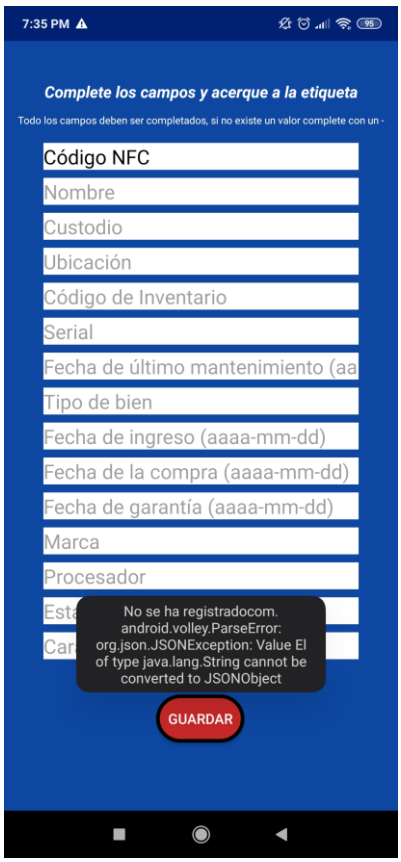


Figura 3.23. Error en el registro

3.3.2.3 Leer Etiqueta

Cuando se requiera consultar la información de cada bien, se debe acceder a la opción LEER ETIQUETA del menú. La aplicación lee el código NFC de la etiqueta, lo asocia con el registro del bien al que corresponde y muestra la información de este. En la Figura 3.24 se muestra el resultado leyendo la información del ejemplo anterior.

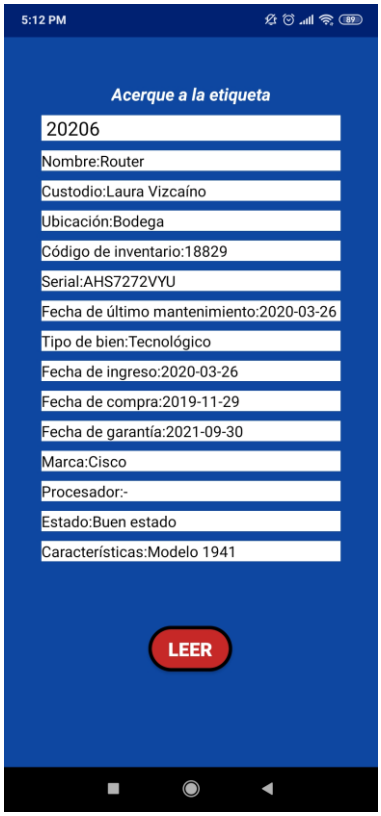


Figura 3.24. Se muestra la información asociada al código NFC

En el caso de que el valor del código NFC que lee la etiqueta no existe, no se muestra ningún valor en los campos como se mira en la Figura 3.25.

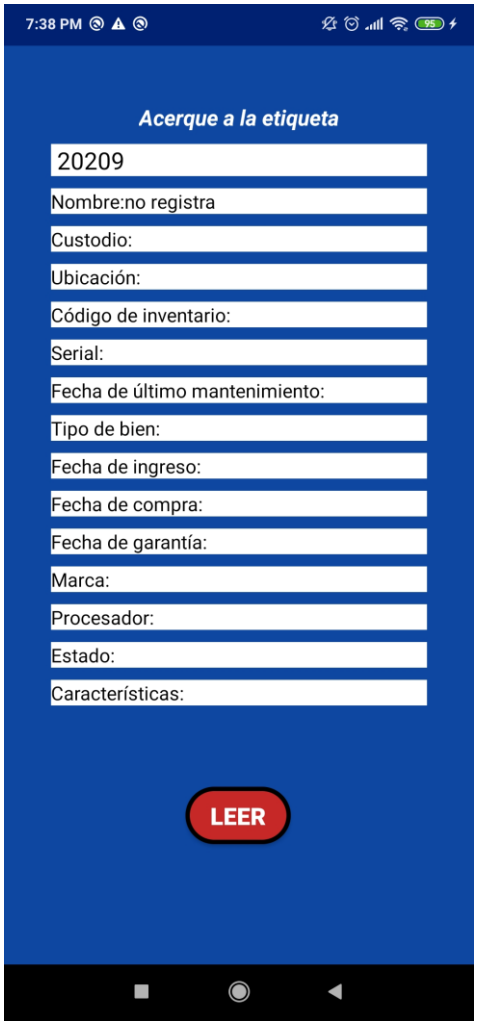


Figura 3.25. Código NFC inválido

3.3.2.4 Error de Conexión

Los principales errores, además de los inconvenientes antes descritos como no completar los campos o completarlos incorrectamente, son errores de conexión. En un primer caso iniciar la sesión no será posible si no existe conexión a Internet. En este escenario la aplicación devuelve un mensaje de error de conexión como se mira en la Figura 3.26.

Figura 3.26. Error de conexión 1. Fuente: Autor

En un segundo caso cuando existen problemas de conexión, es decir, cuando la velocidad de Internet es baja, existe intermitencia en la conexión, el servidor tarda en responder; la aplicación devuelve el mensaje mostrado en la Figura 3.27. En este escenario se debe volver a cargar la actividad y devolverá lo esperado.

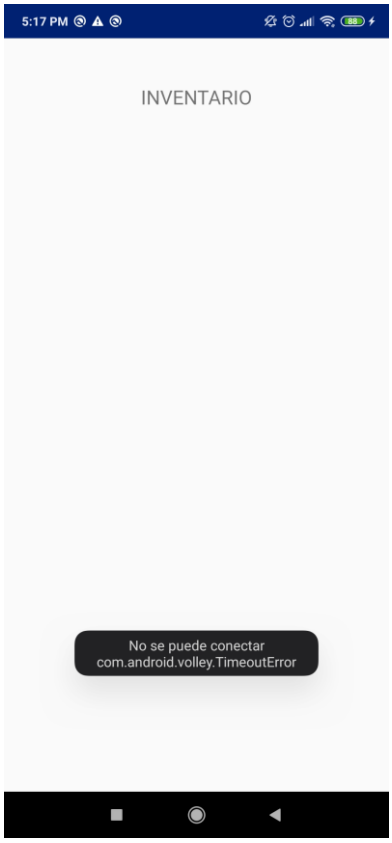


Figura 3.27. Error de conexión 2. Fuente: Autor

3.4 TIEMPOS DE OCURRENCIAS DE EVENTOS

A continuación, se realiza un resumen de los tiempos de las acciones de la aplicación móvil. Los datos son obtenidos de Android Studio utilizando la ventana de *Logcat* que muestra los mensajes del sistema.

Tabla 3.2. Tiempos de lanzamiento de las actividades. Fuente: Android Studio

Evento	**Tiempo**
Login	324ms
Menú	96ms
Escribir etiqueta	195ms
Leer etiqueta	165ms
Visualizar inventario	137ms

La respuesta de las aplicaciones depende de la conectividad a la red y de la velocidad de carga y descarga que se disponga. A continuación, en la Figura 3.28 y 3.29 se muestra una

prueba de ping hacia la IP del servidor de diferentes computadores con diferentes proveedores de internet.

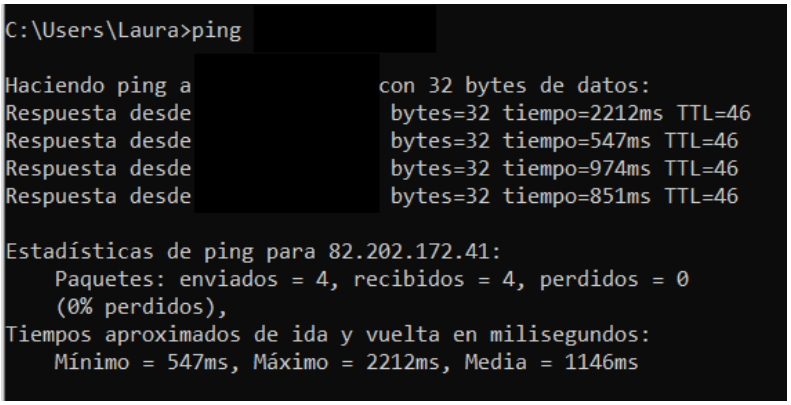


Figura 3.28. Ping hacia el servidor 1

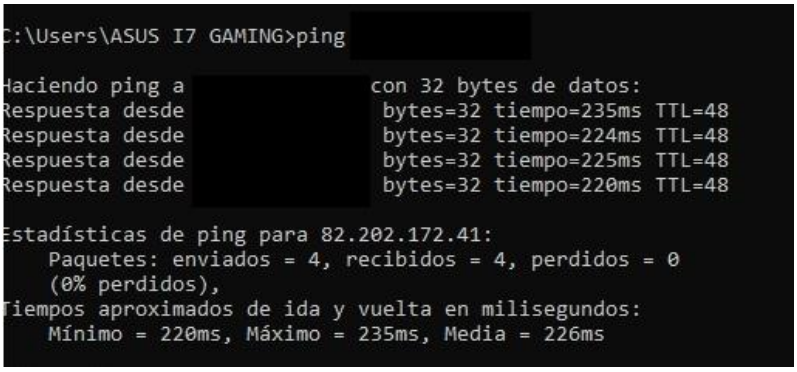


Figura 3.29. Ping hacia el servidor 2

3.5 ANÁLISIS DE RESULTADOS

3.5.1 LOGIN

La aplicación web está diseñada principalmente para la administración de usuarios, se tiene dos grupos de usuarios, uno de administradores y otro de usuarios especiales. Solamente el usuario que pertenezca al grupo administradores puede añadir nuevos usuarios y crear sus credenciales y solo con estas se puede ingresar a las aplicaciones. El administrador puede editar las credenciales o la contraseña en caso de olvido.

3.5.2 REGISTRO EN BASE DE DATOS

En la aplicación móvil al momento de registrar un nuevo bien si todos los campos no son completados, la información del bien no será almacenada en la base de datos. En la

aplicación web se pueden dejar campos en blanco al momento de registra un nuevo bien ya que esta funcionalidad se dispuso para bienes que no necesiten una etiqueta NFC.

3.5.3 VISUALIZACIÓN DE INFORMACIÓN

Se comprueba el correcto registro de cada bien en la página Bienes- Administrar Bienes de la aplicación web y en la actividad Inventario de la aplicación móvil. Además, de siempre tener el respaldo de que todos estos registros se encuentran en la base de datos, la misma cuenta con un registro no visible para el usuario final que es la fecha y hora de cuando se registra un nuevo bien, esta información puede ser útil para el administrador.

3.5.4 LECTURA Y ESCRITURA DE LAS ETIQUETAS

Con respecto a la lectura y escritura de las etiquetas se comprueba que se producen errores cuando el terminal móvil no se encuentra lo suficientemente cerca de la misma, entonces, para que haya una correcta lectura y escritura el móvil debe estar por lo menos a un centímetro de distancia de la etiqueta y no se debe separarlo de la misma hasta que la aplicación devuelva el mensaje de registro exitoso. Así también para que una mejor detección de la etiqueta es mejor que el móvil no se encuentre con un estuche o carcasa.

La aplicación móvil ofrece la opción de leer la etiqueta sin necesidad de iniciar sesión, aquí solo se miran los seis campos escritos en la etiqueta y que son los más importantes.

La aplicación móvil dispone del Sistema de Despacho en Primer Plano lo que permite que cuando el celular detecte una etiqueta NFC, solo entre en funcionamiento la aplicación en cuestión, y no otras que disponga el dispositivo para lectura y escritura de etiquetas NFC.

En cuanto a sintaxis el código NFC está declarado como un entero por consiguiente un bien no se registra si no se completa como se indica. Además de que este código no se puede repetir ya que debe ser único, en este caso tampoco se registrará el nuevo bien.

3.5.5 ERRORES DE CONEXIÓN

La aplicación web y la aplicación móvil envían y reciben las consultas hacia y desde el servidor, en el presente proyecto para la comunicación de frontend con backend se utiliza un servicio de hosting, el mismo brinda una dirección IP que se utiliza para los web services de la aplicación móvil, por consiguiente, la rapidez y el tiempo con que ambas aplicaciones respondan a solicitudes del usuario dependerá de su conexión a Internet.

4. CONCLUSIONES Y RECOMENDACIONES

Una vez concluido el proyecto y gracias a la experiencia y resultados obtenidos a través de este se concluye lo siguiente:

4.1 CONCLUSIONES

- Al examinar los resultados presentados se concluye que el prototipo implementado cumple con el objetivo de ofrecer a la Escuela Politécnica Nacional un sistema de administración de inventario de bienes mediante el uso de una aplicación móvil que brinda las facilidades de utilizar la tecnología NFC.

- El uso de la tecnología NFC permite que el sistema sea más eficiente, al ser una tecnología inalámbrica de corto alcance para intercambio de datos, no es necesario que se haga previas configuraciones como otras tecnologías, basta con acercar el móvil para que la etiqueta sea identificada. Así se potencia su uso con la integración con teléfonos inteligentes con sistema operativo Android.

- Se implementó la base de datos que permite el almacenamiento de la información de los usuarios y de los bienes utilizando el gestor de base de datos MySQL Workbench con sus respectivas tablas y registros. Así la base de datos es un componente fundamental ya que en esta se encuentran todos los datos que se ingresan en la aplicación web y la aplicación móvil. El uso de MySQL hace que no se tenga un limitado número de registros y de columnas, es decir que si el usuario lo solicita se puede seguir añadiendo nuevos campos para las características de los bienes y actualmente está disponible para almacenar hasta mil registros, mismos que se pueden aumentar según sea necesario.

- La aplicación web y la aplicación móvil hacen uso de credenciales que son encriptadas con una función hash. De esta manera el inicio de sesión es seguro.

- La aplicación web cumple con las funcionalidades planteadas para el completo funcionamiento del sistema, esto es, la gestión de los datos de los usuarios y de los bienes. Se desarrolló mediante HTML y CSS para su estructura, lo que permite al usuario tener una interfaz amigable y se utilizó el lenguaje de programación PHP para la interacción con la base de datos. La aplicación permite la creación, visualización, modificación y borrado de los bienes y de usuarios, datos que se encuentran en la base de datos.

- La aplicación móvil desarrollada con Android Studio cumple con el objetivo planeado; escribe la etiqueta, lee la misma y muestra las características de cada bien. La aplicación permite añadir al inventario un nuevo bien ingresando los campos requeridos como: nombre, tipo de bien, código NFC, serial, código de inventario, custodio, ubicación, fecha de ingreso, fecha de la compra, fecha de último mantenimiento, fecha de garantía, marca, procesador, estado y características. Para una identificación rápida del bien como de sus características la aplicación también tiene la función de lectura donde acercando el móvil a la etiqueta la aplicación muestra los campos antes definidos de un bien ya registrado, mismos campos que ya se encuentran en la base de datos.

- La comunicación de la aplicación web con la base de datos se realiza mediante PHP dentro de HTML. La comunicación entre la aplicación móvil y la base de datos se realiza mediante Volley (biblioteca HTTP de Android Studio) y web services escritos en PHP. Esto garantiza que el usuario accede a la base de datos interactuando con las aplicaciones.

- El sistema utiliza un servicio de hosting por lo que las aplicaciones son accesibles desde cualquier parte con acceso a Internet, de esta forma el usuario puede mirar fácilmente el inventario independientemente de que la información se encuentre en un archivo almacenado en un computador de la universidad.

- Se comprueba el funcionamiento del sistema observando que el inventario se visualiza en la aplicación web, en la aplicación móvil y en la base de datos. Así cuando un nuevo bien es añadido o un bien es modificado se ve reflejado en las interfaces de las aplicaciones correspondientes para una mejor gestión para el administrador de bienes.

- La aplicación móvil permite además la lectura de la información escrita en la etiqueta cuando no se tiene acceso a Internet, funcionalidad creada para cuando el bien se encuentre en mantenimiento y se requiera un rápido conocimiento de sus principales características resolviendo problemas de identificación del bien.

4.2 RECOMENDACIONES

A continuación, se presentan algunas recomendaciones con el fin de un correcto manejo del prototipo y su mejoramiento:

- Para un correcto funcionamiento de la aplicación el smartphone deberá estar lo suficientemente cerca de la etiqueta, además se debe tenerlo cerca de esta hasta que la aplicación muestre el mensaje de éxito en la escritura o lectura.
- No se debe utilizar el mismo identificador para todos los bienes ya que esta acción desencadena en problemas de registro de estos. Para mejorar el sistema se recomienda que los identificadores que se escriben en cada etiqueta sean generados de manera automática así el usuario no debe estar atento a que código ya está siendo utilizado.
- Se recomienda el uso de etiquetas de la serie NTAG21x ya que son compatibles con Android, debido a que existen otras marcas de etiquetas que no lo son.
- Es posible mejorar el sistema haciendo uso de los receptores de emisiones de Android generando una notificación cuando un bien es añadido o modificado para el administrador del sistema.
- Se podría mejorar el sistema también añadiendo un buscador en la aplicación web y en la aplicación móvil para encontrar un bien a través de ciertos filtros.
- Aprovechando las funcionalidades de la tecnología NFC se puede realizar un sistema similar para el control de asistencia para que el estudiante acercando su teléfono a un lector registre su asistencia a cada clase.
- También se puede utilizar esta tecnología para un acceso seguro por ejemplo a la casa o para abrir un automóvil utilizando el móvil como una llave de acceso.

5. REFERENCIAS BIBLIOGRÁFICAS

[1] W. Laveriano, “Importancia del control de inventarios en la empresa”, p. 1 [Online]. Disponible en: http://biblioteca.iplacex.cl/RCA/Importancia%20del%20control%20de%20inventarios%20en%20la%20empresa.pdf

[2] A. A. López, D. J. M. González, C. M. Morales, y I. P. Fajardo, “DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO PARA EL REGISTRO Y VERIFICACIÓN DE ACTIVOS FIJOS UTILIZANDO PLATAFORMA ANDROID Y TECNOLOGÍA NFC”, *Redes Ing.*, vol. 5, núm. 1, pp. 26–33, may 2014, doi: 10.14483/2248762X.7117. [Online]. Disponible en: https://revistas.udistrital.edu.co/index.php/REDES/article/view/7117. [Consultado: 20-mar-2020]

[3] M. F. Carignano, “NFC (Near Field Communication)”, p. 22 [Online]. Disponible en: https://rdu.iua.edu.ar/bitstream/123456789/462/1/TFI_ESE_Maria_Fernanda_Carignano.pdf

[4] “Standard ECMA-340”. [Online]. Disponible en: https://www.ecma-international.org/publications/standards/Ecma-340.htm. [Consultado: 20-mar-2020]

[5] 14:00-17:00, “ISO/IEC 18092:2013”, *ISO*. [Online]. Disponible en: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/66/56692.html. [Consultado: 20-mar-2020]

[6] Evander Pierre, “About Us”, *NFC Forum*. [Online]. Disponible en: https://nfc-forum.org/about-us/. [Consultado: 22-mar-2020]

[7] tommysven, “Our Mission & Goals”, *NFC Forum*. [Online]. Disponible en: https://nfc-forum.org/about-us/the-nfc-forum/. [Consultado: 22-mar-2020]

[8] “ISO/IEC 14443-3:2011”, *ISO*. [Online]. Disponible en: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/09/50942.html. [Consultado: 22-mar-2020]

[9] “ISO/IEC 18092:2013”, *ISO*. [Online]. Disponible en: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/66/56692.html. [Consultado: 22-mar-2020]

[10] “Sony Global - FeliCa - Overview of FeliCa - What is FeliCa ?” [Online]. Disponible en: https://www.sony.net/Products/felica/about/. [Consultado: 22-mar-2020]

[11] N. F. C. Forum, “What are the operating modes of NFC devices?”, *NFC Forum*. [Online]. Disponible en: https://nfc-forum.org/resources/what-are-the-operating-modes-of-nfc-devices/. [Consultado: 22-mar-2020]

[12] E. L. A. Gómez, “ESTUDIO DE LA TECNOLOGÍA INALÁMBRICA NFC (NEAR FIELD COMMUNICATION) Y SUS APLICACIONES EN EL ÁMBITO DE LAS TELECOMUNICACIONES”, p. 141 [Online]. Disponible en: https://bibdigital.epn.edu.ec/handle/15000/6440

[13] Evander Pierre, “Specification Overview”, *NFC Forum*. [Online]. Disponible en: https://nfc-forum.org/our-work/specification-releases/specifications/. [Consultado: 22-mar-2020]

[14] A. Anaya y I. López, “La tecnología NFC en teléfonos celulares, sus retos y aplicaciones”, p. 11, 2014 [Online]. Disponible en: https://pdfs.semanticscholar.org/b6be/5db6aae1d12947499271840dadfb3705a2b9.pdf

[15] A. J. B. Meneses y C. G. V. Garcia, “DISEÑO E IMPLEMENTACIÓN DE UN PROTOTIPO PARA EL CONTROL DE ACCESO EN LA SEDE DE INGENIERÍA DE

LA UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS MEDIANTE EL USO DE TORNIQUETES CONTROLADOS POR CARNET CON TECNOLOGIA NFC Y LECTOR BIOMÉTRICO DE HUELLA DACTILAR.", p. 100, 2016 [Online]. Disponible en: http://repository.udistrital.edu.co/bitstream/11349/3430/1/VargasGarciaCristianGerman2016.pdf

[16] V. Subtil, *Near Field Communication with Android Cookbook*. Packt Publishing Ltd, 2014.

[17] D. J. R. Proaño, "PROTOTIPO DE COBRO DE TARIFA PARA EL ACCESO A ANDEN DE PARADAS DE BUSES CON DINERO ELECTRÓNICO", p. 133 [Online]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/17276

[18] J. L. O. Tobar, "DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA PROTOTIPO DE ETIQUETAS INFORMATIVAS DE USO ACADÉMICO UTILIZANDO UNA APLICACIÓN ANDROID MEDIANTE COMUNICACIÓN NFC", p. 125 [Online]. Disponible en: http://bibdigital.epn.edu.ec/handle/15000/19293

[19] "Android Overview | Open Handset Alliance". [Online]. Disponible en: https://www.openhandsetalliance.com/android_overview.html. [Consultado: 25-mar-2020]

[20] "1. Introducción - Software de Comunicaciones". [Online]. Disponible en: https://sites.google.com/site/swcuc3m/home/android/portada. [Consultado: 25-mar-2020]

[21] "Arquitectura de la plataforma | Desarrolladores de Android", *Android Developers*. [Online]. Disponible en: https://developer.android.com/guide/platform?hl=es-419. [Consultado: 25-mar-2020]

[22] "Introducción a Android Studio | Desarrolladores de Android", *Android Developers*. [Online]. Disponible en: https://developer.android.com/studio/intro?hl=es-419. [Consultado: 26-mar-2020]

[23] "Descripción general de proyectos | Desarrolladores de Android", *Android Developers*. [Online]. Disponible en: https://developer.android.com/studio/projects?hl=es-419. [Consultado: 26-mar-2020]

[24] V. Coskun, K. Ok, y B. Ozdenizci, "Professional: NFC Application Development for Android™", p. 316.

[25] "Cómo interpretar el ciclo de vida de una actividad". [Online]. Disponible en: https://developer.android.com/guide/components/activities/activity-lifecycle?hl=es. [Consultado: 30-abr-2020]

[26] G. M. M. Guadalupe, "USOS Y TIPOS DE APLICACIONES MÓVILES", p. 24 [Online]. Disponible en: https://www.academia.edu/13777638/USOS_Y_TIPOS_DE_APLICACIONES_M%C3%93VILES

[27] C. Coronel, S. Morris, y P. Rob, *Base de Datos: Diseno, Implementacion y Administracion*. Cengage Learning Editores, 2011.

[28] V. Olaya, "Sistemas de Información Geográfica", p. 854 [Online]. Disponible en: https://volaya.github.io/libro-sig/index.html

[29] J. M. A. Hernández, P. C. Cantero, T. C. Delgado, y R. M. S. Montero, "Desarrollo de una aplicación Web para la gestión de Entornos Virtuales", p. 83 [Online]. Disponible en: https://eprints.ucm.es/13083/1/Memoria_SI_Final.pdf

[30] C. Nedelcu, *Nginx HTTP Server*. Packt Publishing Ltd, 2015.

[31] Arsys, "Nginx, una de las mejores alternativas entre los servidores web", *Blog de arsys.es*, 05-abr-2017. [Online]. Disponible en: https://www.arsys.es/blog/programacion/nginx-servidor-web/. [Consultado: 15-may-2020]

[32] C. Sergio Luján Mora, “Programación de aplicaciones web: historia, principios básicos y clientes web”, *Editor. Club Univ.*, p. 354 [Online]. Disponible en: https://rua.ua.es/dspace/bitstream/10045/16995/1/sergio_lujan-programacion_de_aplicaciones_web.pdf
[33] “Introducción a los Servicios Web. Invocación de servicios web SOAP.” [Online]. Disponible en: http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.html. [Consultado: 26-mar-2020]
[34] J. López y F. Mayorga, “Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services”, p. 6 [Online]. Disponible en: http://192.188.46.193/bitstream/123456789/60091/1/L%c3%93PEZ%20ROMERO%20JAIME%20ALFREDO%20-%202018.pdf
[35] “Métodos de petición HTTP”, *Documentación web de MDN*. [Online]. Disponible en: https://developer.mozilla.org/es/docs/Web/HTTP/Methods. [Consultado: 24-abr-2020]
[36] G. E. C. G, W. Y. Campo, J. P. Amaya, y J. L. Arciniegas, “Esquema de servicios para Televisión Digital Interactiva, basados en el protocolo REST-JSON”, *Cad. Informática*, vol. 6, núm. 1, pp. 233–240, 2011 [Online]. Disponible en: https://www.seer.ufrgs.br/cadernosdeinformatica/article/view/v6n1p233-240. [Consultado: 24-abr-2020]
[37] L. E. Tello, C. M. S. R, y D. A. T. Leal, “Revisión de los Sistemas de Control de Versiones utilizados en el desarrollo de software”, *Rev. Ing. USBMed*, vol. 3, núm. 1, pp. 74–81, 2012 [Online]. Disponible en: https://dialnet.unirioja.es/servlet/articulo?codigo=4694154. [Consultado: 30-mar-2020]
[38] “Git”. [Online]. Disponible en: https://git-scm.com/. [Consultado: 30-mar-2020]
[39] “Cómo elegir la etiqueta NFC - Shop NFC”. [Online]. Disponible en: https://www.shopnfc.com/es/content/11-como-elegir-la-etiqueta-nfc. [Consultado: 29-mar-2020]
[40] “Características técnicas de las Etiquetas NFC - Shop NFC”. [Online]. Disponible en: https://www.shopnfc.com/es/content/6-caracteristicas-tecnicas-de-etiquetas-nfc. [Consultado: 25-abr-2020]
[41] “NTAG213/215/216 | NXP”. [Online]. Disponible en: https://www.nxp.com/products/rfid-nfc/nfc-hf/ntag/ntag-for-tags-labels/ntag-213-215-216-nfc-forum-type-2-tag-compliant-ic-with-144-504-888-bytes-user-memory:NTAG213_215_216. [Consultado: 29-mar-2020]
[42] “IDC - Smartphone Market Share - OS”, *IDC: The premier global market intelligence company*. [Online]. Disponible en: https://www.idc.com/promo/smartphone-market-share. [Consultado: 30-mar-2020]
[43] F. J. Lopez-Pellicer, R. Béjar, M. A. Latre, J. Nogueras-Iso, y F. J. Zarazaga-Soria, “GitHub como herramienta docente”, presentado en Actas de las XXI Jornadas de la Enseñanza Universitaria de la Informática, 2015, pp. 66–73 [Online]. Disponible en: https://upcommons.upc.edu/handle/2117/76761. [Consultado: 30-mar-2020]
[44] “Descripción general de Volley | Desarrolladores de Android”, *Android Developers*. [Online]. Disponible en: https://developer.android.com/training/volley?hl=es-419. [Consultado: 26-abr-2020]
[45] “Cómo crear una lista con RecyclerView | Desarrolladores de Android”, *Android Developers*. [Online]. Disponible en: https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=es. [Consultado: 30-abr-2020]

ANEXOS

ANEXO A. Código – Aplicación Web

Archivos de configuración

load.php

```
define("DS", DIRECTORY_SEPARATOR);


defined('SITE_ROOT')? null: define('SITE_ROOT', realpath(dirname(__FILE__)))
;
define("LIB_PATH_INC", SITE_ROOT.DS);


require_once(LIB_PATH_INC.'config.php');
require_once(LIB_PATH_INC.'functions.php');
require_once(LIB_PATH_INC.'session.php');
require_once(LIB_PATH_INC.'database.php');
require_once(LIB_PATH_INC.'sql.php');

?>
```

config.php

```
<?php

  define( 'DB_HOST', '' );          // Host Base de datos
  define( 'DB_USER', '' );             // Usuario Base de Datos
  define( 'DB_PASS', '' );             // Password
  define( 'DB_NAME', '' );        // Nombre Base de Datos

?>
```

functions.php

```
<?php

 date_default_timezone_set("America/Lima");
 $errors = array();

 /*-------------------------------------------------------------*/
 /* Función para eliminar caracteres especiales en un string
```

```php
 /* para usar en una sentencia SQL
 /*--------------------------------------------------------------*/
function real_escape($str){
  global $con;
  $escape = mysqli_real_escape_string($con,$str);
  return $escape;
}
/*--------------------------------------------------------------*/
/* Función para eliminar caracteres html
/*--------------------------------------------------------------*/
function remove_junk($str){
  $str = nl2br($str);
  $str = htmlspecialchars(strip_tags($str, ENT_QUOTES));
  return $str;
}
/*--------------------------------------------------------------*/
/* Función para verificar que los campos de entrada no estén vacíos
/*--------------------------------------------------------------*/
function validate_fields($var){
  global $errors;
  foreach ($var as $field) {
    $val = remove_junk($_POST[$field]);
    if(isset($val) && $val==''){
      $errors = $field ." El campo no puede estar en blanco.";
      return $errors;
    }
  }
}
/*--------------------------------------------------------------*/
/* Función para mostrar mensaje de sesión
   Ex echo displayt_msg($message);
/*--------------------------------------------------------------*/
function display_msg($msg =''){
   $output = array();
   if(!empty($msg)) {
      foreach ($msg as $key => $value) {
         $output  = "<div class=\"alert alert-{$key}\">";
         $output .= "<a href=\"#\" class=\"close\" data-
dismiss=\"alert\">&times;</a>";
         $output .= remove_junk(first_character($value));
         $output .= "</div>";
      }
      return $output;
   } else {
     return "" ;
   }
```

```
}
/*-------------------------------------------------------------*/
/* Función para redirigir
/*-------------------------------------------------------------*/
function redirect($url, $permanent = false)
{
    if (headers_sent() === false)
    {
      header('Location: ' . $url, true, ($permanent === true) ? 301 : 302);
    }

    exit();
}

/*-------------------------------------------------------------*/
/* Función para lectura legible fecha hora
/*-------------------------------------------------------------*/
function make_date(){

  return strftime("%Y-%m-%d %H:%M:%S", time());
}
/*-------------------------------------------------------------*/
/* Función contador
/*-------------------------------------------------------------*/
function count_id(){
  static $count = 1;
  return $count++;
}

/*-------------------------------------------------------------*/
/* Función para cambiar la fecha a español
/*-------------------------------------------------------------*/
function fechaCastellano ($fecha) {
  $fecha = substr($fecha, 0, 10);
  $numeroDia = date('d', strtotime($fecha));
  $dia = date('l', strtotime($fecha));
  $mes = date('F', strtotime($fecha));
  $anio = date('Y', strtotime($fecha));
  $dias_ES = array("Lunes", "Martes", "Miércoles", "Jueves", "Viernes", "Sáb
ado", "Domingo");
  $dias_EN = array("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "
Saturday", "Sunday");
  $nombredia = str_replace($dias_EN, $dias_ES, $dia);
$meses_ES = array("Enero", "Febrero", "Marzo", "Abril", "Mayo", "Junio", "Ju
lio", "Agosto", "Septiembre", "Octubre", "Noviembre", "Diciembre");
```

```
  $meses_EN = array("January", "February", "March", "April", "May", "June",
"July", "August", "September", "October", "November", "December");
  $nombreMes = str_replace($meses_EN, $meses_ES, $mes);
  return $nombredia." ".$numeroDia." de ".$nombreMes." de ".$anio;
}

?>
```

sql.php

```
<?php
  require_once('includes/load.php');

/*--------------------------------------------------------------*/
/* Función para buscar todas las filas de las tabla de la base de datos por
el nombre de la tabla
/*--------------------------------------------------------------*/
function find_all($table) {
   global $db;
   if(tableExists($table))
   {
     return find_by_sql("SELECT * FROM ".$db->escape($table));
   }
}
/*--------------------------------------------------------------*/
/* Función para realizar consultas
/*--------------------------------------------------------------*/
function find_by_sql($sql)
{
  global $db;
  $result = $db->query($sql);
  $result_set = $db->while_loop($result);
 return $result_set;
}
/*--------------------------------------------------------------*/
/* Función para buscar información de una tabla por su id
/*--------------------------------------------------------------*/
function find_by_id($table,$id)
{
  global $db;
  $id = (int)$id;
    if(tableExists($table)){
          $sql = $db->query("SELECT * FROM {$db-
>escape($table)} WHERE id='{$db->escape($id)}' LIMIT 1");
          if($result = $db->fetch_assoc($sql))
            return $result;
```

```
          else
            return null;
     }
}
/*--------------------------------------------------------------*/
/* Función para borrar información de una tabla por id
/*--------------------------------------------------------------*/
function delete_by_id($table,$id)
{
  global $db;
  if(tableExists($table))
   {
    $sql = "DELETE FROM ".$db->escape($table);
    $sql .= " WHERE id=". $db->escape($id);
    $sql .= " LIMIT 1";
    $db->query($sql);
    return ($db->affected_rows() === 1) ? true : false;
   }
}
/*--------------------------------------------------------------*/
/* Función para contar id por el nombre de la tabla
/*--------------------------------------------------------------*/

function count_by_id($table){
  global $db;
  if(tableExists($table))
  {
    $sql    = "SELECT COUNT(id) AS total FROM ".$db->escape($table);
    $result = $db->query($sql);
     return($db->fetch_assoc($result));
  }
}
/*--------------------------------------------------------------*/
/* Determinar si la tabla existe
/*--------------------------------------------------------------*/
function tableExists($table){
  global $db;
  $table_exit = $db->query('SHOW TABLES FROM '.DB_NAME.' LIKE "'.$db-
>escape($table).'"');
      if($table_exit) {
        if($db->num_rows($table_exit) > 0)
              return true;
         else
              return false;
      }
  }
```

```
 /*--------------------------------------------------------------*/
 /* Iniciar sesión con los datos proporcionados en $ _POST,
 /* provenientes del formulario de inicio de sesión.
/*--------------------------------------------------------------*/
  function authenticate($username='', $password='') {
    global $db;
    $username = $db->escape($username);
    $password = $db->escape($password);
    $sql  = sprintf("SELECT id,username,password,user_level FROM users WHERE
 username ='%s' LIMIT 1", $username);
    $result = $db->query($sql);
    if($db->num_rows($result)){
      $user = $db->fetch_assoc($result);
      $password_request = sha1($password);
      if($password_request === $user['password'] ){
        return $user['id'];
      }
    }
   return false;
  }

  /*--------------------------------------------------------------*/
  /* Buscar todos los usuarios
  /* uniendo la tabla de usuarios y la tabla de grupos de usuarios
  /*--------------------------------------------------------------*/
  function find_all_user(){
      global $db;
      $results = array();
      $sql = "SELECT u.id,u.name,u.username,u.user_level,u.status,";
      $sql .="g.group_name ";
      $sql .="FROM users u ";
      $sql .="LEFT JOIN user_groups g ";
      $sql .="ON g.group_level=u.user_level ORDER BY u.name ASC";
      $result = find_by_sql($sql);
      return $result;
  }
   /*--------------------------------------------------------------*/
   /* Función para encontrar todos los nombres de la tabla products
   /*--------------------------------------------------------------*/
  function join_product_table(){
     global $db;
     $sql  =" SELECT p.`id`,
     p.`name`,
     p.`date`,
     p.`codigo_nfc`,
     p.`serial`,
```

```
     p.`codigo_inventario`,
     p.`custodio`,
     p.`fecha_ingreso`,
     p.`fecha_compra`,
     p.`ubicacion`,
     p.`fecha_ultimo_mantenimiento`,
     p.`fecha_garantia`,
     p.`marca`,
     p.`procesador`,
     p.`estado`,
     p.`caracteristica`,
     p.`tipo`,
     p.`estado`,
     p.`caracteristica`";

    $sql  .=" FROM products p";
    $sql  .=" ORDER BY p.id ASC";
    return find_by_sql($sql);

   }
  /*--------------------------------------------------------------*/
  /* Función para encontrar toda la información de los productos por su nomb
re
  /*
  /*--------------------------------------------------------------*/
  function find_all_product_info_by_title($title){
    global $db;
    $sql  = "SELECT * FROM products ";
    $sql .= " WHERE name ='{$title}'";
    $sql .=" LIMIT 1";
    return find_by_sql($sql);
  }

  /*--------------------------------------------------------------*/
  /* Función para mostrar añadidos recientemente
  /*--------------------------------------------------------------*/
 function find_recent_product_added($limit){
   global $db;
   $sql   = " SELECT p.`id`,p.`name`";
   $sql  .= "FROM products p";
   $sql  .= " ORDER BY p.id DESC LIMIT ".$db->escape((int)$limit);
   return find_by_sql($sql);
 }

?>
```

Layouts

Header.php

```
<?php $user = current_user();
  include_once('includes/load.php')
?>
<!DOCTYPE html>
  <html lang="en">
    <head>
    <meta charset="UTF-8">
    <title><?php if (!empty($page_title))
            echo remove_junk($page_title);
             elseif(!empty($user))
            echo ucfirst($user['name']);
             else echo "Sistema de Inventario";?>
    </title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3
.3.4/css/bootstrap.min.css"/>
    <link rel="stylesheet" href="//cdnjs.cloudflare.com/ajax/libs/bootstrap-
datepicker/1.3.0/css/datepicker3.min.css" />
    <link rel="stylesheet" href="libs/css/main.css" />
  </head>
  <body>
  <?php  if ($session->isUserLoggedIn(true)): ?>
    <header id="header">
      <div class="logo pull-left"> Opciones
        <a href="home.php"></a>
        </div>

        <div class="header-content ">
            <div class="header-date pull-left">
              <strong><?php $actual_date = date("d-m-Y");
              echo fechaCastellano($actual_date); ?></strong>
            </div>
          <div class="pull-right clearfix">
            <ul class="info-menu list-inline list-unstyled">
              <li class="profile">
                <a href="#" data-toggle="dropdown" class="toggle" aria-
expanded="false">
                  <span><?php echo remove_junk(ucfirst($user['name'])); ?> <
i class="caret"></i></span>
                </a>
                <ul class="dropdown-menu">

                  <li>
```

```
                    <a href="edit_account.php" title="edit account">
                        <i class="glyphicon glyphicon-cog"></i>
                        Ajustes
                    </a>
                </li>
                <li class="last">
                    <a href="logout.php">
                        <i class="glyphicon glyphicon-off"></i>
                        Salir
                    </a>
                </li>
              </ul>
              </li>
            </ul>
        </div>
     </div>
    </header>
    <div class="sidebar">
      <?php if($user['user_level'] === '1'): ?>
        <!-- admin menu -->
      <?php include_once('admin_menu.php');?>

      <?php elseif($user['user_level'] === '2'): ?>
        <!-- Special user menu -->
      <?php include_once('special_menu.php');?>

      <?php endif;?>

   </div>
<?php endif;?>

<div class="page">
  <div class="container-fluid">
```

admin_menu.php

```
<ul>
  <li>
    <a href="admin.php">
      <i class="glyphicon glyphicon-home"></i>
      <span>Resumen</span>
    </a>
  </li>
  <li>
    <a href="#" class="submenu-toggle">
      <i class="glyphicon glyphicon-user"></i>
```

```
      <span>Usuarios</span>
    </a>
    <ul class="nav submenu">
      <li><a href="group.php">Administrar grupos</a> </li>
      <li><a href="users.php">Administrar usuarios</a> </li>
   </ul>
  </li>

  <li>
    <a href="#" class="submenu-toggle">
      <i class="glyphicon glyphicon-th-large"></i>
      <span>Bienes</span>
    </a>
    <ul class="nav submenu">
       <li><a href="product.php">Administrar bienes</a> </li>
       <li><a href="add_product.php">Añadir bien</a> </li>
   </ul>
  </li>

</ul>
```

special_menu.php

```
<ul>

  <li>
    <a href="#" class="submenu-toggle">
      <i class="glyphicon glyphicon-th-large"></i>
      <span>Bienes</span>
    </a>
    <ul class="nav submenu">
       <li><a href="product.php">Administrar bienes</a> </li>
       <li><a href="add_product.php">Añadir bien</a> </li>
   </ul>
  </li>
</ul>
```

Funciones

add_group.php

```
<?php
  $page_title = 'Añadir grupo';
  require_once('includes/load.php');
  // Comprobar qué nivel de usuario tiene permiso para ver esta página
   page_require_level(1);
```

```
?>
<?php
  if(isset($_POST['add'])){

   $req_fields = array('group-name','group-level');
   validate_fields($req_fields);  // Comprobar qué nivel de usuario tiene pe
rmiso para ver esta página


   if(find_by_groupName($_POST['group-name']) === false ){
     $session-
>msg('d','El nombre del grupo ingresado ya está en la base de datos!');
     redirect('add_group.php', false);
   }elseif(find_by_groupLevel($_POST['group-level']) === false) {
     $session-
>msg('d','El nivel del grupo ingresado ya está en la base de datos!');
     redirect('add_group.php', false);
   }
   if(empty($errors)){
           $name = remove_junk($db->escape($_POST['group-name']));
          $level = remove_junk($db->escape($_POST['group-level']));
         $status = remove_junk($db->escape($_POST['status']));

        $query  = "INSERT INTO user_groups (";
        $query .="group_name,group_level,group_status";
        $query .=") VALUES (";
        $query .=" '{$name}', '{$level}','{$status}'";
        $query .=")";
        if($db->query($query)){
          $session->msg('s',"El grupo ha sido creado ");
          redirect('add_group.php', false);
        } else {
          $session->msg('d',' No se pudo crear el grupo.');
          redirect('add_group.php', false);
        }
   } else {
     $session->msg("d", $errors);
      redirect('add_group.php',false);
   }
 }
?>
<?php include_once('layouts/header.php'); ?>
<div class="login-page">
    <div class="text-center">
       <h3>Añadir nuevo grupo</h3>
     </div>
```

```
     <?php echo display_msg($msg); ?>
      <form method="post" action="add_group.php" class="clearfix">
        <div class="form-group">
              <label for="name" class="control-
label">Nombre de grupo</label>
              <input type="name" class="form-control" name="group-name">
        </div>
        <div class="form-group">
              <label for="level" class="control-label">Nivel</label>
              <input type="number" class="form-control" name="group-level">
        </div>
        <div class="form-group">
          <label for="status">Estado</label>
            <select class="form-control" name="status">
              <option value="1">Activo</option>
              <option value="0">Inactivo</option>
            </select>
        </div>
        <div class="form-group clearfix">
                <button type="submit" name="add" class="btn btn-
info">Actualizar</button>
        </div>
    </form>
</div>

<?php include_once('layouts/footer.php'); ?>
```

add_product.php

```
<?php
  $page_title = 'Añadir bien';
  require_once('includes/load.php');
  // Comprobar qué nivel de usuario tiene permiso para ver esta página
  page_require_level(2);

  $all_users=find_all('users');
?>
<?php
 if(isset($_POST['add_product'])){

   $req_fields = array('name','tipo', 'codigo_inventario', 'custodio', 'ubic
acion', 'fecha_ingreso', 'fecha_compra' );
   validate_fields($req_fields);
   if(empty($errors)){
     $p_name  = remove_junk($db->escape($_POST['name']));
     $p_tipo   = remove_junk($db->escape($_POST['tipo']));
```

```
      //$p_nfc   = remove_junk($db->escape($_POST['codigo_nfc']));
      //$p_ser   = remove_junk($db->escape($_POST['serial']));
      if (is_null($_POST['serial']) || $_POST['serial'] === "") {
        $p_ser = '0';
      } else {
        $p_ser= remove_junk($db->escape($_POST['serial']));
      }
      $p_cod   = remove_junk($db->escape($_POST['codigo_inventario']));
      $p_cus   = remove_junk($db->escape($_POST['custodio']));
      $p_ubi   = remove_junk($db->escape($_POST['ubicacion']));
      $p_ing   = remove_junk($db->escape($_POST['fecha_ingreso']));
      $p_com   = remove_junk($db->escape($_POST['fecha_compra']));
      //$p_man   = remove_junk($db-
>escape($_POST['fecha_ultimo_mantenimiento']));
      if (is_null($_POST['fecha_ultimo_mantenimiento']) || $_POST['fecha_ulti
mo_mantenimiento'] === "") {
        $$p_man = '-';
      } else {
        $$p_man= remove_junk($db-
>escape($_POST['fecha_ultimo_mantenimiento']));
      }
      //$p_gar   = remove_junk($db->escape($_POST['fecha_garantia']));
      if (is_null($_POST['fecha_garantia']) || $_POST['fecha_garantia'] === "
") {
        $p_gar = '-';
      } else {
        $p_gar= remove_junk($db->escape($_POST['fecha_garantia']));
      }
      //$p_mar   = remove_junk($db->escape($_POST['marca']));
      if (is_null($_POST['marca']) || $_POST['marca'] === "") {
        $p_mar = '-';
      } else {
        $p_mar= remove_junk($db->escape($_POST['marca']));
      }
      //$p_pro   = remove_junk($db->escape($_POST['procesador']));
      if (is_null($_POST['procesador']) || $_POST['procesador'] === "") {
        $p_pro  = '-';
      } else {
        $p_pro = remove_junk($db->escape($_POST['procesador']));
      }
      //$p_est   = remove_junk($db->escape($_POST['estado']));
      if (is_null($_POST['estado']) || $_POST['estado'] === "") {
        $p_est = '-';
      } else {
        $p_est = remove_junk($db->escape($_POST['estado']));
      }
```

```
     //$p_car  = remove_junk($db->escape($_POST['caracteristica']));
      if (is_null($_POST['caracteristica']) || $_POST['caracteristica'] ===
"") {
       $p_car = '-';
     } else {
       $p_car = remove_junk($db->escape($_POST['caracteristica']));
     }
     //$date    = make_date();
     $query  = "INSERT INTO products (";
     $query .=" name,tipo,serial,codigo_inventario,custodio,ubicacion,fecha_
ingreso,fecha_compra,fecha_ultimo_mantenimiento,fecha_garantia,marca,procesa
dor,estado,caracteristica";
     $query .=") VALUES (";
     $query .=" '{$p_name}', '{$p_tipo}','{$p_ser}', '{$p_cod}', '{$p_cus}',
 '{$p_ubi}','{$p_ing}','{$p_com}', '{$p_man}', '{$p_gar}', '{$p_mar}','{$p_p
ro}','{$p_est}','{$p_car}'";
     $query .=")";
     $query .=" ON DUPLICATE KEY UPDATE name='{$p_name}'";
     if($db->query($query)){
       $session->msg('s',"Añadido");
       redirect('add_product.php', false);
     } else {
       $session->msg('d',' No se pudo agregar!');
       redirect('product.php', false);
     }

   } else{
     $session->msg("d", $errors);
     redirect('add_product.php',false);
   }

 }

?>
<?php include_once('layouts/header.php'); ?>
<div class="row">
  <div class="col-md-12">
    <?php echo display_msg($msg); ?>
  </div>
</div>
  <div class="row">
  <div class="col-md-8">
      <div class="panel panel-default">
        <div class="panel-heading">
          <strong>
            <span class="glyphicon glyphicon-th"></span>
```

```
            <span>Añadir nuevo bien</span>
          </strong>
         </div>
         <div class="panel-body">
          <div class="col-md-12">
             <form method="post" action="add_product.php" class="clearfix">
               <div class="form-group">
                 <div class="input-group">
                   <span class="input-group-addon">
                    <i class="glyphicon glyphicon-th-large"></i>
                   </span>
                   <input type="text" class="form-
control" name="name" placeholder="Nombre">
                 </div>
               </div>

               <div class="form-group">
                <div class="row row-cols-2">
                   <!--<div class="col-md-6">
                     <div class="form-group">
                       <label for="qty">Código NFC</label>
                       <div class="input-group ">
                        <input type="number" class="form-
control" name="codigo_nfc" placeholder="Código NFC" id="qty">

                       </div>
                     </div>
                   </div>-->

                   <div class="col-md-6">
                     <div class="form-group">
                       <label for="qty">Tipo de bien</label>
                       <div class="input-group ">
                        <input type="text" class="form-
control" name="tipo" placeholder="Tipo de bien" id="qty">

                       </div>
                     </div>
                   </div>

                   <div class="col-md-6">
                     <div class="form-group">
                       <label for="qty">Serial</label>
                       <div class="input-group">
                         <input type="text" class="form-
control" name="serial" placeholder="Número de serial">
```

```
                    </div>
                  </div>
                </div>

                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Código de Inventario</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="codigo_inventario" placeholder="Codigo de inventario">

                    </div>
                  </div>
                </div>

                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Nombre del Custodio</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="custodio" placeholder="Nombre del custodio">

                    </div>
                  </div>
                </div>

                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Ubicación</label>
                    <div class="input-group">
                        <input type="text" class="form-
control" name="ubicacion" placeholder="Ubicación del bien">

                    </div>
                  </div>
                </div>

                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Fecha de ingreso</label>
                    <div class="input-group">
                      <input type="date" class="form-
control" name="fecha_ingreso" placeholder="aaaa-mm-
dd">
                    </div>
                  </div>
```

```
                    </div>


                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Fecha de la compra</label>
                        <div class="input-group">
                          <input type="date" class="form-
control" name="fecha_compra" placeholder="aaaa-mm-dd">
                        </div>
                      </div>
                    </div>

                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Fecha del último mantenimiento</label
>
                        <div class="input-group">
                          <input type="date" class="form-
control" name="fecha_ultimo_mantenimiento" placeholder="aaaa-mm-
dd">
                        </div>
                      </div>
                    </div>

                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Fecha de la garantía</label>
                        <div class="input-group">
                          <input type="date" class="form-
control" name="fecha_garantia" placeholder="aaaa-mm-
dd">
                        </div>
                      </div>
                    </div>

                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Marca</label>
                        <div class="input-group">
                          <input type="text" class="form-
control" name="marca" placeholder="Marca">
                        </div>
                      </div>
                    </div>
```

```
                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Procesador</label>
                        <div class="input-group">
                          <input type="text" class="form-
control" name="procesador" placeholder="Procesador">
                        </div>
                      </div>
                    </div>

                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Estado</label>
                        <div class="input-group">
                          <input type="text" class="form-
control" name="estado" placeholder="Estado">
                        </div>
                      </div>
                    </div>

                    <div class="col-md-6">
                      <div class="form-group">
                        <label for="qty">Características</label>
                        <div class="input-group">
                          <input type="text" class="form-
control" name="caracteristica" placeholder="Características">

                        </div>
                      </div>
                    </div>

                  </div>

                </div>
                <button type="submit" name="add_product" class="btn btn-
danger">Añadir</button>
              </form>
            </div>
          </div>
        </div>
      </div>

<?php include_once('layouts/footer.php'); ?>
```

add_user.php

```php
<?php
  $page_title = 'Añadir usuario';
  require_once('includes/load.php');
  // Comprobar qué nivel de usuario tiene permiso para ver esta página
  page_require_level(1);
  $groups = find_all('user_groups');
?>
<?php
  if(isset($_POST['add_user'])){

   $req_fields = array('full-name','username','password','level' );
   validate_fields($req_fields);

   if(empty($errors)){
           $name   = remove_junk($db->escape($_POST['full-name']));
       $username   = remove_junk($db->escape($_POST['username']));
       $password   = remove_junk($db->escape($_POST['password']));
       $user_level = (int)$db->escape($_POST['level']);
       $password = sha1($password);
        $query = "INSERT INTO users (";
        $query .="name,username,password,user_level,status";
        $query .=") VALUES (";
        $query .=" '{$name}', '{$username}', '{$password}', '{$user_level}',
'1'";
        $query .=")";
        if($db->query($query)){
          //sucess
          $session->msg('s',"¡Se ha creado la cuenta de usuario! ");
          redirect('add_user.php', false);
        } else {
          //failed
          $session->msg('d','No se pudo crear la cuenta.');
          redirect('add_user.php', false);
        }
   } else {
     $session->msg("d", $errors);
      redirect('add_user.php',false);
   }
 }
?>
<?php include_once('layouts/header.php'); ?>
  <?php echo display_msg($msg); ?>
  <div class="row">
    <div class="panel panel-default">
      <div class="panel-heading">
        <strong>
```

```
          <span class="glyphicon glyphicon-th"></span>
          <span>Añadir nuevo usuario</span>
       </strong>
      </div>
      <div class="panel-body">
        <div class="col-md-6">
          <form method="post" action="add_user.php">
            <div class="form-group">
                <label for="name">Nombre</label>
                <input type="text" class="form-control" name="full-
name" placeholder="Nombre">
            </div>
            <div class="form-group">
                <label for="username">Usuario</label>
                <input type="text" class="form-
control" name="username" placeholder="Usuario">
            </div>
            <div class="form-group">
                <label for="password">Contraseña</label>
                <input type="password" class="form-
control" name ="password"  placeholder="Contraseña">
            </div>
            <div class="form-group">
              <label for="level">Rol de usuario</label>
                <select class="form-control" name="level">
                  <?php foreach ($groups as $group ):?>
                   <option value="<?php echo $group['group_level'];?>"><?php
 echo ucwords($group['group_name']);?></option>
                <?php endforeach;?>
                </select>
            </div>
            <div class="form-group clearfix">
              <button type="submit" name="add_user" class="btn btn-
primary">Añadir Usuario</button>
            </div>
        </form>
        </div>

      </div>

    </div>
  </div>

<?php include_once('layouts/footer.php'); ?>
```

auth.php

```php
<?php include_once('includes/load.php'); ?>
<?php
$req_fields = array('username','password' );
validate_fields($req_fields);
$username = remove_junk($_POST['username']);
$password = remove_junk($_POST['password']);

if(empty($errors)){
  $user_id = authenticate($username, $password);
  if($user_id){
     $session->login($user_id);
     $session->msg("s", "Bienvenido");
     redirect('home.php',false);

  } else {
    $session->msg("d", "Nombre de usuario/Contraseña incorrecta.");
    redirect('index.php',false);
  }

} else {
   $session->msg("d", $errors);
   redirect('index.php',false);
}

?>
```

delete_group.php

```php
<?php
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
   page_require_level(1);
?>
<?php
  $delete_id = delete_by_id('user_groups',(int)$_GET['id']);
  if($delete_id){
      $session->msg("s","El grupo ha sido eliminado");
      redirect('group.php');
  } else {
      $session->msg("d","No se pudoeliminar el grupo");
      redirect('group.php');
  }
?>
```

delete_product.php

```php
<?php
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
  page_require_level(2);
?>
<?php
  $product = find_by_id('products',(int)$_GET['id']);
  if(!$product){
    $session->msg("d","Falta id del producto.");
    redirect('product.php');
  }
?>
<?php
  $delete_id = delete_by_id('products',(int)$product['id']);
  if($delete_id){
      $session->msg("s","Borrado");
      redirect('product.php');
  } else {
      $session->msg("d","No se pudo borrar.");
      redirect('product.php');
  }
?>
```

delete_user.php

```php
<?php
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
   page_require_level(1);
?>
<?php
  $delete_id = delete_by_id('users',(int)$_GET['id']);
  if($delete_id){
      $session->msg("s","Usuario borrado.");
      redirect('users.php');
  } else {
      $session->msg("d","No se pudo borrar el usuario");
      redirect('users.php');
  }
?>
```

edit_group.php

```php
<?php
  $page_title = 'Editar Grupo';
  require_once('includes/load.php');
```

```php
  // Checkin What level user has permission to view this page
   page_require_level(1);
?>
<?php
  $e_group = find_by_id('user_groups',(int)$_GET['id']);
  if(!$e_group){
    $session->msg("d","Falta la identificación del grupo.");
    redirect('group.php');
  }
?>
<?php
  if(isset($_POST['update'])){

   $req_fields = array('group-name','group-level');
   validate_fields($req_fields);
   if(empty($errors)){
           $name = remove_junk($db->escape($_POST['group-name']));
          $level = remove_junk($db->escape($_POST['group-level']));
         $status = remove_junk($db->escape($_POST['status']));

        $query  = "UPDATE user_groups SET ";
        $query .= "group_name='{$name}',group_level='{$level}',group_status=
'{$status}'";
        $query .= "WHERE ID='{$db->escape($e_group['id'])}'";
        $result = $db->query($query);
         if($result && $db->affected_rows() === 1){
          //sucess
          $session->msg('s',"¡El grupo ha sido actualizado!");
          redirect('edit_group.php?id='.(int)$e_group['id'], false);
        } else {
          //failed
          $session->msg('d','No se pudo actualizar el grupo.');
          redirect('edit_group.php?id='.(int)$e_group['id'], false);
        }
   } else {
     $session->msg("d", $errors);
    redirect('edit_group.php?id='.(int)$e_group['id'], false);
   }
 }
?>
<?php include_once('layouts/header.php'); ?>
<div class="login-page">
    <div class="text-center">
       <h3>Editar Grupo</h3>
     </div>
     <?php echo display_msg($msg); ?>
```

```
      <form method="post" action="edit_group.php?id=<?php echo (int)$e_group
['id'];?>" class="clearfix">
        <div class="form-group">
              <label for="name" class="control-
label">Nombre de grupo</label>
              <input type="name" class="form-control" name="group-
name" value="<?php echo remove_junk(ucwords($e_group['group_name'])); ?>">
        </div>
        <div class="form-group">
              <label for="level" class="control-label">Nivel</label>
              <input type="number" class="form-control" name="group-
level" value="<?php echo (int)$e_group['group_level']; ?>">
        </div>
        <div class="form-group">
          <label for="status">Estado</label>
              <select class="form-control" name="status">
                <option <?php if($e_group['group_status'] === '1') echo 'sel
ected="selected"';?> value="1"> Activo </option>
                <option <?php if($e_group['group_status'] === '0') echo 'sel
ected="selected"';?> value="0">Inactivo</option>
              </select>
        </div>
        <div class="form-group clearfix">
                <button type="submit" name="update" class="btn btn-
info">Actualizar</button>
        </div>
    </form>
</div>

<?php include_once('layouts/footer.php'); ?>
```

edit_product.php

```
<?php
  $page_title = 'Editar';
  require_once('includes/load.php');
  // Checkin What level user has permission to view this page
   page_require_level(2);
?>
<?php
$product = find_by_id('products',(int)$_GET['id']);


if(!$product){
  $session->msg("d","Falta la identificación del produto");
  redirect('product.php');
```

```
}
?>
<?php
 if(isset($_POST['product'])){
    $req_fields = array('product-
title','tipo','serial', 'codigo_inventario', 'custodio', 'ubicacion', 'fecha
_ingreso', 'fecha_compra', 'fecha_ultimo_mantenimiento', 'fecha_garantia', '
marca','procesador','estado','caracteristica' );
    validate_fields($req_fields);

   if(empty($errors)){
       $p_name  = remove_junk($db->escape($_POST['product-title']));
       $p_tipo   = remove_junk($db->escape($_POST['tipo']));
       //$p_nfc   = remove_junk($db->escape($_POST['codigo_nfc']));
       $p_ser   = remove_junk($db->escape($_POST['serial']));
       $p_cod   = remove_junk($db->escape($_POST['codigo_inventario']));
       $p_cus   = remove_junk($db->escape($_POST['custodio']));
       $p_ubi   = remove_junk($db->escape($_POST['ubicacion']));
       $p_ing   = remove_junk($db->escape($_POST['fecha_ingreso']));
       $p_com   = remove_junk($db->escape($_POST['fecha_compra']));
       $p_man   = remove_junk($db-
>escape($_POST['fecha_ultimo_mantenimiento']));
       $p_gar   = remove_junk($db->escape($_POST['fecha_garantia']));
       $p_mar   = remove_junk($db->escape($_POST['marca']));
       $p_pro   = remove_junk($db->escape($_POST['procesador']));
       $p_est   = remove_junk($db->escape($_POST['estado']));
       $p_car   = remove_junk($db->escape($_POST['caracteristica']));

       $query   = "UPDATE products SET";
       $query   .= " name ='{$p_name}', tipo ='{$p_tipo}',serial='{$p_ser}',
 codigo_inventario='{$p_cod}', custodio='{$p_cus}', ubicacion='{$p_ubi}', fe
cha_ingreso='{$p_ing}', fecha_compra='{$p_com}', fecha_ultimo_mantenimiento=
'{$p_man}', fecha_garantia='{$p_gar}', marca='{$p_mar}', procesador='{$p_pro
}',estado='{$p_est}',caracteristica='{$p_car}'";
       $query  .=" WHERE id ='{$product['id']}'";
       $result = $db->query($query);
               if($result && $db->affected_rows() === 1){
                 $session->msg('s',"Producto actualizado");
                 redirect('product.php', false);
               } else {
                 $session->msg('d',' No se pudo actualizar!');
                 redirect('edit_product.php?id='.$product['id'], false);
               }

   } else{
       $session->msg("d", $errors);
```

```
       redirect('edit_product.php?id='.$product['id'], false);
   }

 }

?>
<?php include_once('layouts/header.php'); ?>
<div class="row">
  <div class="col-md-12">
    <?php echo display_msg($msg); ?>
  </div>
</div>
  <div class="row">
      <div class="panel panel-default">
        <div class="panel-heading">
          <strong>
            <span class="glyphicon glyphicon-th"></span>
            <span>Editar</span>
         </strong>
        </div>
        <div class="panel-body">
         <div class="col-md-7">
           <form method="post" action="edit_product.php?id=<?php echo (int)$
product['id'] ?>">
              <div class="form-group">
                <div class="input-group">
                  <span class="input-group-addon">
                   <i class="glyphicon glyphicon-th-large"></i>
                  </span>
                  <input type="text" class="form-control" name="product-
title" value="<?php echo remove_junk($product['name']);?>">
                </div>
              </div>

              <div class="form-group">
               <div class="row row-cols-2">
                 <!--<div class="col-md-6">
                    <div class="form-group">
                      <label for="qty">Código NFC</label>
                      <div class="input-group">
                        <input type="number" class="form-
control" name="codigo_nfc" value="<?/*php echo remove_junk($product['codigo_
nfc']); */?>">
                      </div>
                    </div>
                 </div>-->
```

```
                <div class="col-md-6">
                   <div class="form-group">
                     <label for="qty">Tipo de bien</label>
                     <div class="input-group">
                       <input type="text" class="form-
control" name="tipo" value="<?php echo remove_junk($product['tipo']); ?>">
                     </div>
                   </div>
                </div>
                <div class="col-md-6">
                   <div class="form-group">
                     <label for="qty">Serial</label>
                     <div class="input-group">
                       <input type="text" class="form-
control" name="serial" value="<?php echo remove_junk($product['serial']); ?>
">
                     </div>
                   </div>
                </div>
                <div class="col-md-6">
                   <div class="form-group">
                     <label for="qty">Código de Inventario</label>
                     <div class="input-group">
                       <input type="text" class="form-
control" name="codigo_inventario" value="<?php echo remove_junk($product['co
digo_inventario']); ?>">
                     </div>
                   </div>
                </div>
                <div class="col-md-6">
                   <div class="form-group">
                     <label for="qty">Nombre del Custodio</label>
                     <div class="input-group">
                       <input type="text" class="form-
control" name="custodio" value="<?php echo remove_junk($product['custodio'])
; ?>">
                     </div>
                   </div>
                </div>
                <div class="col-md-6">
                   <div class="form-group">
                     <label for="qty">Fecha de ingreso</label>
                     <div class="input-group">
                       <input type="date" class="form-
control" name="fecha_ingreso" value="<?php echo remove_junk($product['fecha_
ingreso']); ?>">
```

```
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Fecha de la compra</label>
                    <div class="input-group">
                      <input type="date" class="form-
control" name="fecha_compra" value="<?php echo remove_junk($product['fecha_c
ompra']); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Ubicación</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="ubicacion" value="<?php echo remove_junk($product['ubicacion'
]); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Fecha del último mantenimiento</label
>
                    <div class="input-group">
                      <input type="date" class="form-
control" name="fecha_ultimo_mantenimiento" value="<?php echo remove_junk($pr
oduct['fecha_ultimo_mantenimiento']); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Fecha de la garantía</label>
                    <div class="input-group">
                      <input type="date" class="form-
control" name="fecha_garantia" value="<?php echo remove_junk($product['fecha
_garantia']); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
```

```
                    <label for="qty">Marca</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="marca" value="<?php echo remove_junk($product['marca']); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Procesador</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="procesador" value="<?php echo remove_junk($product['procesado
r']); ?>">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Estado</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="estado" value="<?php echo remove_junk($product['estado']); ?>
">
                    </div>
                  </div>
                </div>
                <div class="col-md-6">
                  <div class="form-group">
                    <label for="qty">Características</label>
                    <div class="input-group">
                      <input type="text" class="form-
control" name="caracteristica" value="<?php echo remove_junk($product['carac
teristica']); ?>">
                    </div>
                  </div>
                </div>
              </div>
              <button type="submit" name="product" class="btn btn-
danger">Actualizar</button>
          </form>
        </div>
      </div>
    </div>
  </div>
```

```
<?php include_once('layouts/footer.php'); ?>
```

home.php

```
<?php
  $page_title = 'Home';
  require_once('includes/load.php');
  if (!$session->isUserLoggedIn(true)) { redirect('index.php', false);}
?>
<?php include_once('layouts/header.php'); ?>
<div class="row">
  <div class="col-md-12">
    <?php echo display_msg($msg); ?>
  </div>
 <div class="col-md-12">
    <div class="panel">
      <div class="jumbotron text-center">
         <h1>Sistema de Inventario</h1>
         <h2>Facultad de Ingeniería Eléctrica y Electrónica</h2>
         <h3>Laboratorio de Informática</h3>
         <img src="https://www.epn.edu.ec/wp-
content/uploads/2014/05/EPN_logo-250x152.jpg" alt="Escudo EPN">
          </div>
    </div>
 </div>
</div>
<?php include_once('layouts/footer.php'); ?>
```

index.php

```
<?php
  ob_start();
  require_once('includes/load.php');
  if($session->isUserLoggedIn(true)) { redirect('home.php', false);}
?>
<?php include_once('layouts/header.php'); ?>
<div class="login-page">
    <div class="text-center">
       <h1>Bienvenido</h1>
       <p>Inicia sesión para comenzar</p>
     </div>
     <?php echo display_msg($msg); ?>
      <form method="post" action="auth.php" class="clearfix">
        <div class="form-group">
```

```
              <label for="username" class="control-
label">Nombre de Usuario</label>
              <input type="name" class="form-
control" name="username" placeholder="Usuario">
        </div>
        <div class="form-group">
            <label for="Password" class="control-label">Contraseña</label>
            <input type="password" name= "password" class="form-
control" placeholder="Contraseña">
        </div>
        <div class="form-group">
                <button type="submit" class="btn btn-info  pull-
right">Iniciar sesión</button>
        </div>
    </form>
</div>
<?php include_once('layouts/footer.php'); ?>
```

logout.php

```
<?php
  require_once('includes/load.php');
  if(!$session->logout()) {redirect("index.php");}
?>
```

product.php

```
<?php
  $page_title = 'Inventario';
  require_once('includes/load.php');
  // Comprobar qué nivel de usuario tiene permiso para ver esta página
   page_require_level(2);
  $products = join_product_table();
?>
<?php include_once('layouts/header.php'); ?>
  <div class="row">
     <div class="col-md-12">
       <?php echo display_msg($msg); ?>
     </div>
    <div class="col-md-12">
      <div class="panel panel-default">
        <div class="panel-heading clearfix">
         <div class="pull-right">
           <a href="add_product.php" class="btn btn-
primary">Añadir nuevo</a>
         </div>
```

```
        </div>
        <div class="panel-body"  style="overflow: scroll;">
          <table class="table table-bordered">
            <thead>
              <tr>
                <th class="text-center" style="width: 50px;">#</th>
                <th class="text-center" style="width: 10%;"> Nombre</th>
                <th class="text-
center" style="width: 10%;"> Tipo de bien </th>
                <th class="text-
center" style="width: 10%;"> Código NFC </th>
                <th class="text-center" style="width: 10%;"> Serial </th>
                <th class="text-
center" style="width: 10%;"> Código Inventario </th>
                <th class="text-center" style="width: 10%;"> Custodio </th>
                <th class="text-center" style="width: 10%;"> Ubicación </th>
                <th class="text-
center" style="width: 10%;"> Fecha de ingreso </th>
                <th class="text-
center" style="width: 10%;"> Fecha de la compra </th>
                <th class="text-
center" style="width: 10%;"> Fecha de último mantenimiento </th>
                <th class="text-
center" style="width: 10%;"> Fecha de garantía </th>
                <th class="text-center" style="width: 10%;"> Marca </th>
                <th class="text-
center" style="width: 10%;"> Procesador </th>
                <th class="text-center" style="width: 10%;"> Estado </th>
                <th class="text-
center" style="width: 10%;"> Características </th>
                <th class="text-
center" style="width: 100px;"> Acciones </th>

              </tr>
            </thead>
            <tbody>
              <?php foreach ($products as $product):?>
              <tr>
                <td class="text-center"><?php echo count_id(); ?></td>

                <td class="text-
center"> <?php echo remove_junk($product['name']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['tipo']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['codigo_nfc']); ?></td>
```

```
                <td class="text-
center"> <?php echo remove_junk($product['serial']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['codigo_inventario']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['custodio']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['ubicacion']); ?></td>
                <td class="text-
center"> <?php echo read_date($product['fecha_ingreso']); ?></td>
                <td class="text-
center"> <?php echo read_date($product['fecha_compra']); ?></td>
                <td class="text-
center"> <?php echo read_date($product['fecha_ultimo_mantenimiento']); ?></t
d>
                <td class="text-
center"> <?php echo read_date($product['fecha_garantia']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['marca']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['procesador']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['estado']); ?></td>
                <td class="text-
center"> <?php echo remove_junk($product['caracteristica']); ?></td>

                <td class="text-center">
                  <div class="btn-group">

                    <a href="edit_product.php?id=<?php  echo $product['id'];
?>" class="btn btn-info btn-xs"  title="Editar" data-toggle="tooltip">
                      <span class="glyphicon glyphicon-edit"></span>
                    </a>
                    <a href="delete_product.php?id=<?php echo $product['id']
;?>" class="btn btn-danger btn-xs"  title="Borrar" data-toggle="tooltip">
                      <span class="glyphicon glyphicon-trash"></span>
                    </a>
                  </div>
                </td>
              </tr>
             <?php endforeach; ?>
            </tbody>
          </tabel>
        </div>
      </div>
    </div>
```

```
  </div>
  <?php include_once('layouts/footer.php'); ?>
```

ANEXO B. Código – Web services

wsJSONConsultarBien.php

```
<?PHP
$hostname_localhost ="";
$database_localhost ="";
$username_localhost ="";
$password_localhost ="";

$json=array();

    if(isset($_GET["codigo_nfc"])){
        $codigo_nfc=$_GET["codigo_nfc"];

        $conexion = mysqli_connect($hostname_localhost,$username_localhost,$
password_localhost,$database_localhost);

        $consulta="select name, codigo_nfc, serial, fecha_garantia, codigo_i
nventario, custodio, fecha_ingreso, ubicacion,fecha_ultimo_mantenimiento,fec
ha_compra, marca , procesador, estado, caracteristica, tipo from products wh
ere codigo_nfc= '{$codigo_nfc}'";
        $resultado=mysqli_query($conexion,$consulta);

        if($registro=mysqli_fetch_array($resultado)){
            $json['products'][]=$registro;
        }else{
            $resultar["codigo_nfc"]=0;
            $resultar["name"]='no registra';
            $json['products'][]=$resultar;
        }

        mysqli_close($conexion);
        echo json_encode($json);
    }
    else{
        $resultar["success"]=0;
        $resultar["message"]='Ws no Retorna';
        $json['products'][]=$resultar;
        echo json_encode($json);
    }
?>
```

wsJSONConsultarLista.php

```php
<?PHP

$hostname_localhost ="";
$database_localhost =" ";
$username_localhost ="";
$password_localhost =" ";

$json=array();

        $conexion = mysqli_connect($hostname_localhost,$username_localhost,$
password_localhost,$database_localhost);

        $consulta="select name, codigo_nfc, serial, fecha_garantia, codigo_i
nventario, custodio, fecha_ingreso, ubicacion,fecha_ultimo_mantenimiento,fec
ha_compra, marca , procesador, estado, caracteristica,tipo FROM products";
        $resultado=mysqli_query($conexion,$consulta);

        while($registro=mysqli_fetch_array($resultado)){
            $json['products'][]=$registro;
        }
        mysqli_close($conexion);
        echo json_encode($json);

?>
```

wsJSONlogin.php

```php
<?php
    $hostname_localhost="";
    $database_localhost=" ";
    $username_localhost=" ";
    $password_localhost=" ";

    $conexion=new mysqli($hostname_localhost,$username_localhost,$password_l
ocalhost,$database_localhost);

    if($conexion->connect_error){
        echo "La aplicacion está experimentando problemas";
    }

    $username=$_POST['username'];
```

```
    $password=sha1($_POST['password']);

    $sentencia=$conexion-
>prepare("SELECT * FROM users WHERE username=? AND password=?");
    $sentencia->bind_param('ss',$username,$password);
    $sentencia->execute();

    $resultado = $sentencia->get_result();
    if ($fila = $resultado->fetch_assoc()) {
            echo json_encode($fila,JSON_UNESCAPED_UNICODE);
    }
    $sentencia->close();
    $conexion->close();

?>
```

wsJSONRegistro.php

```
<?PHP

    $hostname_localhost="localhost";
    $database_localhost="inventario";
    $username_localhost="laura";
    $password_localhost="flor.123";

    $json=array();

        if(isset($_GET["name"]) && isset($_GET["codigo_nfc"]) && isset($_GET
["serial"]) && isset($_GET["fecha_garantia"]) && isset($_GET["codigo_inventa
rio"]) && isset($_GET["custodio"]) && isset($_GET["fecha_ingreso"]) && isset
($_GET["ubicacion"]) && isset($_GET["fecha_ultimo_mantenimiento"]) && isset(
$_GET["fecha_compra"]) && isset($_GET["marca"]) && isset($_GET["procesador"]
) && isset($_GET["estado"]) && isset($_GET["caracteristica"]) && isset($_GET
["tipo"])){

            $name=$_GET['name'];
            $codigo_nfc=$_GET['codigo_nfc'];
            $serial=$_GET['serial'];
            $fecha_garantia=$_GET['fecha_garantia'];
            $codigo_inventario=$_GET['codigo_inventario'];
            $custodio=$_GET['custodio'];
            $fecha_ingreso=$_GET['fecha_ingreso'];
            $ubicacion=$_GET['ubicacion'];
            $fecha_ultimo_mantenimiento=$_GET['fecha_ultimo_mantenimiento'];
            $fecha_compra=$_GET['fecha_compra'];
```

```
            $marca=$_GET['marca'];
            $procesador=$_GET['procesador'];
            $estado=$_GET['estado'];
            $caracteristica=$_GET['caracteristica'];
            $tipo=$_GET['tipo'];

            $conexion=new mysqli($hostname_localhost,$username_localhost,$pa
ssword_localhost,$database_localhost);

            $insert="INSERT INTO products(name,codigo_nfc,serial, fecha_gara
ntia, codigo_inventario, custodio, fecha_ingreso, ubicacion,fecha_ultimo_man
tenimiento,fecha_compra, marca , procesador, estado, caracteristica,tipo) VA
LUES ('{$name}',{$codigo_nfc},'{$serial}','{$fecha_garantia}','{$codigo_inve
ntario}','{$custodio}','{$fecha_ingreso}','{$ubicacion}','{$fecha_ultimo_man
tenimiento}','{$fecha_compra}','{$marca}','{$procesador}','{$estado}','{$car
acteristica}','{$tipo}') ON DUPLICATE KEY UPDATE codigo_nfc={$codigo_nfc}";

            if($conexion->query($insert)===TRUE){

                $resultado = $conexion-
>query("SELECT * FROM products WHERE codigo_nfc = {$codigo_nfc}");

                if($registro=mysqli_fetch_array($resultado)){
                    $json['products'][]=$registro;
                }
                mysqli_close($conexion);
                echo json_encode($json);
            }
            else{
                echo "El error es : ".mysqli_error($conexion);
                $resulta["codigo_nfc"]=0;
                $resulta["name"]='No Registra';
                $json['products'][]=$resulta;
                echo json_encode($json);
            }

        }
        else{
                $resulta["codigo_nfc"]=0;
                $resulta["name"]='Servidor no devuelve';
                $json['products'][]=$resulta;
                echo json_encode($json);
            }

?>
```

ANEXO C. Código – Aplicación Móvil

Inventario.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import android.app.ProgressDialog;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonArrayRequest;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;

public class Inventario extends AppCompatActivity implements
Response.Listener<JSONObject>, Response.ErrorListener {

    ArrayList<Campos> listDatos;
    RecyclerView recycler;

    ProgressDialog progress;
    RequestQueue request;
    JsonObjectRequest jsonObjectRequest;
    //JsonArrayRequest jsonArrayRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_inventario);
        getSupportActionBar().hide();

        listDatos=new ArrayList<>();
        recycler=findViewById(R.id.Recyclerid);
        recycler.setLayoutManager(new LinearLayoutManager(this));
        recycler.setHasFixedSize(true);

        request= Volley.newRequestQueue(this);

        cargarWebService();
    }
```

```
    private void cargarWebService() {

        progress=new ProgressDialog(this);
        progress.setMessage("Consultando...");
        progress.show();

        String url = "http:// /backend/wsJSONConsultarLista.php";
        //jsonArrayRequest= new
JsonArrayRequest(Request.Method.GET,url,null,this,this);
        jsonObjectRequest= new
JsonObjectRequest(Request.Method.GET,url,null,this,this);
        request.add(jsonObjectRequest);
        //request.add(jsonArrayRequest);
    }

    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(this,"No se puede conectar
"+error.toString(),Toast.LENGTH_LONG).show();
        Log.d("ERROR: ",error.toString());
        progress.hide();
    }

    @Override
    public void onResponse(JSONObject response) {
        //Toast.makeText(this,"Mensaje: "+response,Toast.LENGTH_SHORT).show();
        Campos campos= null;
        JSONArray json=response.optJSONArray("products");
        //JSONArray json=new JSONArray(response);

        try {

            for (int i = 0; i < json.length(); i++) {
                campos = new Campos();
                JSONObject jsonObject = null;
                jsonObject = json.getJSONObject(i);

                campos.setTxtname(jsonObject.optString("name"));
                campos.setTxtcodigo_nfc(jsonObject.optString("codigo_nfc"));
                campos.setTxtserial(jsonObject.optString("serial"));

campos.setTxtfecha_garantia(jsonObject.optString("fecha_garantia"));

campos.setTxtcodigo_inventario(jsonObject.optString("codigo_inventario"));
                campos.setTxtcustodio(jsonObject.optString("custodio"));
                campos.setTxttipo(jsonObject.optString("tipo"));

campos.setTxtfecha_ingreso(jsonObject.optString("fecha_ingreso"));
                campos.setTxtubicacion(jsonObject.optString("ubicacion"));

campos.setTxtfecha_ultimo_mantenimiento(jsonObject.optString("fecha_ultimo_mant
enimiento"));

campos.setTxtfecha_compra(jsonObject.optString("fecha_compra"));
                campos.setTxtmarca(jsonObject.optString("marca"));
```

```
                campos.setTxtprocesador(jsonObject.optString("procesador"));
                campos.setTxtestado(jsonObject.optString("estado"));

campos.setTxtcaracteristicas(jsonObject.optString("caracteristica"));

                listDatos.add(campos);

            }
            progress.hide();
            AdapterDatos adapter=new AdapterDatos(listDatos);
            recycler.setAdapter(adapter);
        }   catch (JSONException e) {
                e.printStackTrace();
                Toast.makeText(this,"No se ha podido establecer la conexión con
el servidor" +
                        " "+response,Toast.LENGTH_LONG).show();
                progress.hide();
            }
    }
}
```

Lectura.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;

import android.app.PendingIntent;
import android.content.Intent;
import android.content.IntentFilter;
import android.nfc.NdefMessage;
import android.nfc.NdefRecord;
import android.nfc.NfcAdapter;
import android.os.Bundle;
import android.os.Parcelable;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import java.io.UnsupportedEncodingException;

public class Lectura extends AppCompatActivity {

    NfcAdapter nfcAdapter;
    private Button leerOff;
    TextView
contenido,offCodNFC,offName,offCustodio,offUbicacion,offCodInv,offSerial,offMan
t;
    String mensaje1,mensaje2,mensaje3,mensaje4,mensaje5,mensaje6,mensaje7;
    Intent intent1;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lectura);

        getSupportActionBar().hide();

        nfcAdapter = NfcAdapter.getDefaultAdapter(this);
        leerOff=findViewById(R.id.lecturaoff);
        contenido=findViewById(R.id.contenido);
        offCodNFC=findViewById(R.id.offcodNFC);
        offName=findViewById(R.id.offName);
        offCustodio=findViewById(R.id.offCustodio);
        offUbicacion=findViewById(R.id.offUbicacion);
        offCodInv=findViewById(R.id.offCodInv);
        offSerial=findViewById(R.id.offSerial);
        offMant=findViewById(R.id.offMant);

    }

    @Override
    protected void onNewIntent(final Intent intent) {
        super.onNewIntent(intent);
        if (intent.hasExtra(NfcAdapter.EXTRA_TAG)){
            Toast.makeText(this, "Etiqueta NFC encontrada",
Toast.LENGTH_SHORT).show();
            leerOff.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Parcelable[] parcelables =
intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
                    if (parcelables !=null && parcelables.length > 0){
                        readTextFromMessage((NdefMessage)parcelables[0]);
                    }else {
                        Toast.makeText(Lectura.this, "Ningún mensaje NDEF
encontrado", Toast.LENGTH_SHORT).show();
                    }
                }
            });
       }
    }

    private void readTextFromMessage(NdefMessage ndefMessage) {

        NdefRecord[] ndefRecords = ndefMessage.getRecords();

        if(ndefRecords != null && ndefRecords.length>0){
            NdefRecord ndefRecord = ndefRecords[0];

            String tagContent = getTextFromNdefRecord(ndefRecord);

            contenido.setText(tagContent);

        }else{
            Toast.makeText(this, "Ningún registro NDEF encontrado",
Toast.LENGTH_SHORT).show();
        }
```

```
        String total =contenido.getText().toString();
        String[] parte = total.split(",");

        mensaje1=parte[0];
        offCodNFC.setText(" "+mensaje1);
        mensaje2=parte[1];
        offName.setText(" "+mensaje2);
        mensaje3=parte[2];
        offCustodio.setText(" "+mensaje3);
        mensaje4=parte[3];
        offUbicacion.setText(" "+mensaje4);
        mensaje5=parte[4];
        offCodInv.setText(" "+mensaje5);
        mensaje6=parte[5];
        offSerial.setText(" "+mensaje6);
        mensaje7=parte[6];
        offMant.setText(" "+mensaje7);

    }

    public String getTextFromNdefRecord (NdefRecord ndefRecord){
        String tagContent = null;
        try {
            byte[] payload = ndefRecord.getPayload();
            String textEncoding = ((payload[0] & 128) == 0) ? "UTF-8" : "UTF-
16";
            int languageSize = payload[0] & 0063;
            tagContent = new String(payload, languageSize + 1,
                    payload.length - languageSize - 1, textEncoding);
        }catch (UnsupportedEncodingException e ){
            Log.e("getTextFromNdefRecord", e.getMessage(),e);
        }
        return tagContent;
    }

    @Override
    protected void onResume() {
        super.onResume();
        enableForegroundDispatchSystem();
    }

    @Override
    protected void onPause() {
        super.onPause();
        disableForegroundDispatchSystem();
    }

    private void enableForegroundDispatchSystem(){
        Intent intent = new Intent(this,
Lectura.class).addFlags(Intent.FLAG_RECEIVER_REPLACE_PENDING);

        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
intent, 0);

        IntentFilter[] intentFilters = new IntentFilter[] {};
```

```
nfcAdapter.enableForegroundDispatch(this,pendingIntent,intentFilters,null);
    }

    private void disableForegroundDispatchSystem(){

        nfcAdapter.disableForegroundDispatch(this);

    }
}
```

Login.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.JsonRequest;
import com.android.volley.toolbox.StringRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class Login extends AppCompatActivity {

    EditText username,password;
    Button BtnIniciar, BtnOffline;
    String usuario,clave;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
        getSupportActionBar().hide();

        username= findViewById(R.id.editTextUsuario);
        password= findViewById(R.id.editTextPassword);
        BtnIniciar= findViewById(R.id.buttonLogin);
        BtnOffline=findViewById(R.id.buttonOffline);

        BtnOffline.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(Login.this, Lectura.class);
                startActivity(i);
            }
        });

        BtnIniciar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                usuario=username.getText().toString();
                clave=password.getText().toString();
                if(!usuario.isEmpty() && !clave.isEmpty()){
                    validarUsuario("http:// /backend/wsJSONlogin.php");
                }else {
                    Toast.makeText(Login.this, "No se permiten campos
vacíos",Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    private void validarUsuario(String URL){
        StringRequest stringRequest=new StringRequest(Request.Method.POST, URL,
new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                if(!response.isEmpty()){
                    Intent intent=new
Intent(getApplicationContext(),Menu.class);
                    startActivity(intent);
                }else{
                    Toast.makeText(Login.this,"Usuario o contraseña
incorrectos",Toast.LENGTH_SHORT).show();
                }
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {

Toast.makeText(Login.this,"Error"+error.toString(),Toast.LENGTH_SHORT).show();
            }
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                Map<String,String> parametros=new HashMap<String, String>();
                parametros.put("username",username.getText().toString());
                parametros.put("password",password.getText().toString());
```

```
                return parametros;
            }
        };

        RequestQueue requestQueue=Volley.newRequestQueue(this);
        requestQueue.add(stringRequest);
    }
}
```

Menu.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class Menu extends AppCompatActivity {

    Button buttonWrite,buttonInventario,buttonRead;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu);

        getSupportActionBar().hide();

        buttonWrite=findViewById(R.id.buttonWrite);
        buttonInventario=findViewById(R.id.buttonInv);
        buttonRead=findViewById(R.id.buttonRead);

        buttonWrite.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(Menu.this, Write.class);
                startActivity(i);
            }
        });

        buttonInventario.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(Menu.this, Inventario.class);
                startActivity(i);
            }
        });

        buttonRead.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent i = new Intent(Menu.this, Read.class);
```

```
                startActivity(i);
            }
        });
    }
}
```

Read.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;

import android.app.PendingIntent;
import android.app.ProgressDialog;
import android.content.Intent;
import android.content.IntentFilter;
import android.nfc.NdefMessage;
import android.nfc.NdefRecord;
import android.nfc.NfcAdapter;
import android.nfc.Tag;
import android.nfc.tech.Ndef;
import android.nfc.tech.NdefFormatable;
import android.os.Bundle;
import android.os.Parcelable;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.ByteArrayOutputStream;
import java.io.UnsupportedEncodingException;
import java.util.Locale;

public class Read extends AppCompatActivity implements
Response.Listener<JSONObject>,Response.ErrorListener{

    NfcAdapter nfcAdapter;
    Button buttonReadTag;
    String parte1;

    TextView
```

```
codigo,readName,readCustodio,readUbicacion,readCodInv,readSerial,readDateMant,r
eadTipo,

readDateIng,readDateComp,readDateGar,readMarca,readProcesador,readEstado,readCa
racteristicas, txtTagContent;

    ProgressDialog progreso;

    RequestQueue request;
    JsonObjectRequest jsonObjectRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_read);
        getSupportActionBar().hide();

        nfcAdapter = NfcAdapter.getDefaultAdapter(this);

        txtTagContent = findViewById(R.id.txtTagContent);
        buttonReadTag= findViewById(R.id.buttonReadTag);
        codigo=findViewById(R.id.codigo);

        readName=findViewById(R.id.readName);
        readCustodio=findViewById(R.id.readCustodio);
        readUbicacion=findViewById(R.id.readUbicacion);
        readCodInv=findViewById(R.id.readCodInv);
        readSerial=findViewById(R.id.readSerial);
        readDateMant=findViewById(R.id.readDateMant);
        readTipo=findViewById(R.id.readTipo);
        readDateIng=findViewById(R.id.readDateIng);
        readDateComp=findViewById(R.id.readDateComp);
        readDateGar=findViewById(R.id.readDateGar);
        readMarca=findViewById(R.id.readMarca);
        readProcesador=findViewById(R.id.readProcesador);
        readEstado=findViewById(R.id.readEstado);
        readCaracteristicas=findViewById(R.id.readCaracteristicas);

        request= Volley.newRequestQueue(this);
    }

    @Override
    protected void onResume() {
        super.onResume();
        enableForegroundDispatchSystem();
    }

    @Override
    protected void onPause() {
        super.onPause();
        disableForegroundDispatchSystem();
    }

    @Override
    protected void onNewIntent(final Intent intent) {
        super.onNewIntent(intent);
```

```
        if (intent.hasExtra(NfcAdapter.EXTRA_TAG)){
            Toast.makeText(this, "Etiqueta NFC encontrada",
Toast.LENGTH_SHORT).show();

            buttonReadTag.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Parcelable[] parcelables =
intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
                    if (parcelables !=null && parcelables.length > 0){
                        readTextFromMessage((NdefMessage)parcelables[0]);
                        cargarWebService();
                    }else {
                        Toast.makeText(Read.this, "Ningún mensaje NDEF
encontrado", Toast.LENGTH_SHORT).show();
                    }
                }
            });
        }
    }

    private void readTextFromMessage(NdefMessage ndefMessage) {

        NdefRecord [] ndefRecords = ndefMessage.getRecords();

        if(ndefRecords != null && ndefRecords.length>0){
            NdefRecord ndefRecord = ndefRecords[0];

            String tagContent = getTextFromNdefRecord(ndefRecord);

            txtTagContent.setText(tagContent);

        }else{
            Toast.makeText(this, "Ningún registro NDEF encontrado",
Toast.LENGTH_SHORT).show();
        }

        String total =txtTagContent.getText().toString();
        String[] parte = total.split(",");

        parte1=parte[0];
        codigo.setText(" "+parte1);
    }



    private void enableForegroundDispatchSystem(){
        Intent intent = new Intent(this,
Read.class).addFlags(Intent.FLAG_RECEIVER_REPLACE_PENDING);

        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
intent, 0);

        IntentFilter[] intentFilters = new IntentFilter[] {};
```

```
nfcAdapter.enableForegroundDispatch(this,pendingIntent,intentFilters,null);
    }

    private void disableForegroundDispatchSystem(){

        nfcAdapter.disableForegroundDispatch(this);

    }

    private void formatTag (Tag tag , NdefMessage ndefMessage){
        try {
            NdefFormatable ndefFormatable = NdefFormatable.get(tag);

            if (ndefFormatable == null){
                Toast.makeText(this, "Tag is not ndef
formatable",Toast.LENGTH_SHORT).show();
                return;
            }

            ndefFormatable.connect();
            ndefFormatable.format(ndefMessage);
            ndefFormatable.close();

            Toast.makeText(this, "Tag written",Toast.LENGTH_SHORT).show();


        }catch (Exception e){
            Log.e("formatTag",e.getMessage());
        }
    }



    public String getTextFromNdefRecord (NdefRecord ndefRecord){
        String tagContent = null;
        try {
            byte[] payload = ndefRecord.getPayload();
            String textEncoding = ((payload[0] & 128) == 0) ? "UTF-8" : "UTF-
16";
            int languageSize = payload[0] & 0063;
            tagContent = new String(payload, languageSize + 1,
                    payload.length - languageSize - 1, textEncoding);
        }catch (UnsupportedEncodingException e ){
            Log.e("getTextFromNdefRecord", e.getMessage(),e);
        }
        return tagContent;
    }

    private void cargarWebService(){
        progreso=new ProgressDialog(this);
        progreso.setMessage("Consultando...");
        progreso.show();
        String url = "http://
/backend/wsJSONConsultarBien.php?codigo_nfc="+codigo.getText().toString();
```

```
        jsonObjectRequest=new
JsonObjectRequest(Request.Method.GET,url,null,this,this);
        request.add(jsonObjectRequest);
    }

    @Override
    public void onErrorResponse(VolleyError error) {
        progreso.hide();
        Toast.makeText(this,"No se
consultó"+error.toString(),Toast.LENGTH_SHORT).show();
        Log.i("ERROR",error.toString());
    }

    @Override
    public void onResponse(JSONObject response) {
        progreso.hide();
        //Toast.makeText(this,"Mensaje: "+response,Toast.LENGTH_SHORT).show();

        Campos campo =new Campos();

        JSONArray json=response.optJSONArray("products");
        JSONObject jsonObject=null;

        try {
            jsonObject=json.getJSONObject(0);
            campo.setTxtname(jsonObject.optString("name"));
            campo.setTxtserial(jsonObject.optString("serial"));
            campo.setTxtfecha_garantia(jsonObject.optString("fecha_garantia"));

campo.setTxtcodigo_inventario(jsonObject.optString("codigo_inventario"));
            campo.setTxtcustodio(jsonObject.optString("custodio"));
            campo.setTxtfecha_ingreso(jsonObject.optString("fecha_ingreso"));
            campo.setTxtubicacion(jsonObject.optString("ubicacion"));

campo.setTxtfecha_ultimo_mantenimiento(jsonObject.optString("fecha_ultimo_mante
nimiento"));
            campo.setTxttipo(jsonObject.optString("tipo"));
            campo.setTxtfecha_compra(jsonObject.optString("fecha_compra"));
            campo.setTxtmarca(jsonObject.optString("marca"));
            campo.setTxtprocesador(jsonObject.optString("procesador"));
            campo.setTxtestado(jsonObject.optString("estado"));

campo.setTxtcaracteristicas(jsonObject.optString("caracteristica"));
        } catch (JSONException e) {
            e.printStackTrace();
        }

        readName.setText("Nombre:"+campo.getTxtname());
        readCustodio.setText("Custodio:"+campo.getTxtcustodio());
        readUbicacion.setText("Ubicación:"+campo.getTxtubicacion());
        readCodInv.setText("Código de
inventario:"+campo.getTxtcodigo_inventario());
        readSerial.setText("Serial:"+campo.getTxtserial());
        readTipo.setText("Tipo de bien:"+campo.getTxttipo());
        readDateMant.setText("Fecha de último
mantenimiento:"+campo.getTxtfecha_ultimo_mantenimiento());
```

```
        readDateIng.setText("Fecha de
ingreso:"+campo.getTxtfecha_ultimo_mantenimiento());
        readDateComp.setText("Fecha de compra:"+campo.getTxtfecha_compra());
        readDateGar.setText("Fecha de garantía:"+campo.getTxtfecha_garantia());
        readMarca.setText("Marca:"+campo.getTxtmarca());
        readProcesador.setText("Procesador:"+campo.getTxtprocesador());
        readEstado.setText("Estado:"+campo.getTxtestado());

readCaracteristicas.setText("Características:"+campo.getTxtcaracteristicas());
    }
}
```

Write.java

```
package com.example.inventario;

import androidx.appcompat.app.AppCompatActivity;

import android.app.PendingIntent;
import android.app.ProgressDialog;
import android.content.Intent;
import android.content.IntentFilter;
import android.nfc.NdefMessage;
import android.nfc.NdefRecord;
import android.nfc.NfcAdapter;
import android.nfc.Tag;
import android.nfc.tech.Ndef;
import android.nfc.tech.NdefFormatable;
import android.os.Bundle;
import android.os.Parcelable;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;

import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;

import java.io.ByteArrayOutputStream;
import java.io.UnsupportedEncodingException;
import java.util.Locale;

public class Read extends AppCompatActivity implements
Response.Listener<JSONObject>,Response.ErrorListener{
```

```
    NfcAdapter nfcAdapter;
    Button buttonReadTag;
    String parte1;

    TextView
codigo,readName,readCustodio,readUbicacion,readCodInv,readSerial,readDateMant,r
eadTipo,

readDateIng,readDateComp,readDateGar,readMarca,readProcesador,readEstado,readCa
racteristicas, txtTagContent;

    ProgressDialog progreso;

    RequestQueue request;
    JsonObjectRequest jsonObjectRequest;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_read);
        getSupportActionBar().hide();

        nfcAdapter = NfcAdapter.getDefaultAdapter(this);

        txtTagContent = findViewById(R.id.txtTagContent);
        buttonReadTag= findViewById(R.id.buttonReadTag);
        codigo=findViewById(R.id.codigo);

        readName=findViewById(R.id.readName);
        readCustodio=findViewById(R.id.readCustodio);
        readUbicacion=findViewById(R.id.readUbicacion);
        readCodInv=findViewById(R.id.readCodInv);
        readSerial=findViewById(R.id.readSerial);
        readDateMant=findViewById(R.id.readDateMant);
        readTipo=findViewById(R.id.readTipo);
        readDateIng=findViewById(R.id.readDateIng);
        readDateComp=findViewById(R.id.readDateComp);
        readDateGar=findViewById(R.id.readDateGar);
        readMarca=findViewById(R.id.readMarca);
        readProcesador=findViewById(R.id.readProcesador);
        readEstado=findViewById(R.id.readEstado);
        readCaracteristicas=findViewById(R.id.readCaracteristicas);

        request= Volley.newRequestQueue(this);
    }

    @Override
    protected void onResume() {
        super.onResume();
        enableForegroundDispatchSystem();
    }

    @Override
    protected void onPause() {
        super.onPause();
        disableForegroundDispatchSystem();
```

```
    }

    @Override
    protected void onNewIntent(final Intent intent) {
        super.onNewIntent(intent);

        if (intent.hasExtra(NfcAdapter.EXTRA_TAG)){
            Toast.makeText(this, "Etiqueta NFC encontrada",
Toast.LENGTH_SHORT).show();

            buttonReadTag.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    Parcelable[] parcelables =
intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
                    if (parcelables !=null && parcelables.length > 0){
                        readTextFromMessage((NdefMessage)parcelables[0]);
                        cargarWebService();
                    }else {
                        Toast.makeText(Read.this, "Ningún mensaje NDEF
encontrado", Toast.LENGTH_SHORT).show();
                    }
                }
            });
        }
    }


    private void enableForegroundDispatchSystem(){
        Intent intent = new Intent(this,
Read.class).addFlags(Intent.FLAG_RECEIVER_REPLACE_PENDING);

        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0,
intent, 0);

        IntentFilter[] intentFilters = new IntentFilter[] {};


nfcAdapter.enableForegroundDispatch(this,pendingIntent,intentFilters,null);
    }

    private void disableForegroundDispatchSystem(){

        nfcAdapter.disableForegroundDispatch(this);

    }

    private void formatTag (Tag tag , NdefMessage ndefMessage){
        try {
            NdefFormatable ndefFormatable = NdefFormatable.get(tag);

            if (ndefFormatable == null){
                Toast.makeText(this, "Tag is not ndef
formatable",Toast.LENGTH_SHORT).show();
                return;
            }
```

```
            ndefFormatable.connect();
            ndefFormatable.format(ndefMessage);
            ndefFormatable.close();

            Toast.makeText(this, "Tag written",Toast.LENGTH_SHORT).show();


        }catch (Exception e){
            Log.e("formatTag",e.getMessage());
        }
    }

    private void writeNdefMessage(Tag tag, NdefMessage ndefMessage){
        try {

            if (tag == null){
                Toast.makeText(this, "Tag object cannot be
null",Toast.LENGTH_SHORT).show();
                return;
            }

            Ndef ndef = Ndef.get(tag);

            if (ndef == null){
                //format tag with the ndef format and writes the message
                formatTag(tag, ndefMessage);
            }else {
                ndef.connect();

                if (!ndef.isWritable()){
                    Toast.makeText(this, "Tag is not
writable",Toast.LENGTH_SHORT).show();
                    ndef.close();
                    return;
                }

                ndef.writeNdefMessage(ndefMessage);
                ndef.close();

                Toast.makeText(this, "Tag writable",Toast.LENGTH_SHORT).show();
            }

        }catch (Exception e){
            Log.e("writeNdefMessage",e.getMessage());
        }
    }

    private NdefRecord createTextRecord(String content){
        try {
            byte[] language;
            language = Locale.getDefault().getLanguage().getBytes("UTF-8");

            final byte[] text = content.getBytes("UTF-8");
            final int languageSize = language.length;
            final int textLength = text.length;
```

```
            final ByteArrayOutputStream payload = new ByteArrayOutputStream(1 +
languageSize + textLength);

            payload.write((byte) (languageSize & 0x1F));
            payload.write(language, 0, languageSize);
            payload.write(text, 0, textLength);

            return new NdefRecord(NdefRecord.TNF_WELL_KNOWN,
NdefRecord.RTD_TEXT, new byte[0], payload.toByteArray());

        }catch (UnsupportedEncodingException e){
            Log.e("createTextRecord", e.getMessage());
        }
        return null;
    }

    private NdefMessage createNdefMessage(String content){

        NdefRecord ndefRecord = createTextRecord(content);

        NdefMessage ndefMessage = new NdefMessage(new NdefRecord[]{ ndefRecord
});

        return ndefMessage;
    }


    private void cargarWebService(){
        progreso=new ProgressDialog(this);
        progreso.setMessage("Consultando...");
        progreso.show();
        String url = "http://
/backend/wsJSONConsultarBien.php?codigo_nfc="+codigo.getText().toString();

        jsonObjectRequest=new
JsonObjectRequest(Request.Method.GET,url,null,this,this);
        request.add(jsonObjectRequest);
    }

    @Override
    public void onErrorResponse(VolleyError error) {
        progreso.hide();
        Toast.makeText(this,"No se
consultó"+error.toString(),Toast.LENGTH_SHORT).show();
        Log.i("ERROR",error.toString());
    }

    @Override
    public void onResponse(JSONObject response) {
        progreso.hide();
        //Toast.makeText(this,"Mensaje: "+response,Toast.LENGTH_SHORT).show();

        Campos campo =new Campos();

        JSONArray json=response.optJSONArray("products");
        JSONObject jsonObject=null;
```

```
        try {
            jsonObject=json.getJSONObject(0);
            campo.setTxtname(jsonObject.optString("name"));
            campo.setTxtserial(jsonObject.optString("serial"));
            campo.setTxtfecha_garantia(jsonObject.optString("fecha_garantia"));

campo.setTxtcodigo_inventario(jsonObject.optString("codigo_inventario"));
            campo.setTxtcustodio(jsonObject.optString("custodio"));
            campo.setTxtfecha_ingreso(jsonObject.optString("fecha_ingreso"));
            campo.setTxtubicacion(jsonObject.optString("ubicacion"));

campo.setTxtfecha_ultimo_mantenimiento(jsonObject.optString("fecha_ultimo_mante
nimiento"));
            campo.setTxttipo(jsonObject.optString("tipo"));
            campo.setTxtfecha_compra(jsonObject.optString("fecha_compra"));
            campo.setTxtmarca(jsonObject.optString("marca"));
            campo.setTxtprocesador(jsonObject.optString("procesador"));
            campo.setTxtestado(jsonObject.optString("estado"));

campo.setTxtcaracteristicas(jsonObject.optString("caracteristica"));
        } catch (JSONException e) {
            e.printStackTrace();
        }

        readName.setText("Nombre:"+campo.getTxtname());
        readCustodio.setText("Custodio:"+campo.getTxtcustodio());
        readUbicacion.setText("Ubicación:"+campo.getTxtubicacion());
        readCodInv.setText("Código de
inventario:"+campo.getTxtcodigo_inventario());
        readSerial.setText("Serial:"+campo.getTxtserial());
        readTipo.setText("Tipo de bien:"+campo.getTxttipo());
        readDateMant.setText("Fecha de último
mantenimiento:"+campo.getTxtfecha_ultimo_mantenimiento());
        readDateIng.setText("Fecha de
ingreso:"+campo.getTxtfecha_ultimo_mantenimiento());
        readDateComp.setText("Fecha de compra:"+campo.getTxtfecha_compra());
        readDateGar.setText("Fecha de garantía:"+campo.getTxtfecha_garantia());
        readMarca.setText("Marca:"+campo.getTxtmarca());
        readProcesador.setText("Procesador:"+campo.getTxtprocesador());
        readEstado.setText("Estado:"+campo.getTxtestado());

readCaracteristicas.setText("Características:"+campo.getTxtcaracteristicas());
    }
}
```

ORDEN DE EMPASTADO