ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

DESARROLLO DE UN KEYLOGGER PARA WINDOWS 11 CON UN DASHBOARD WEB

DESARROLLO DE UN KEYLOGGER

TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO SUPERIOR EN DESARROLLO DE SOFTWARE

CARLOS JOEL DIAZ ROSERO

DIRECTOR: DR. RICHARD PAUL RIVERA GUEVARA

DMQ, enero 2022

CERTIFICACIONES

Yo, Carlos Joel Díaz Rosero declaro que el trabajo de integración curricular 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.

Carlos Joel Díaz Rosero

carlos.diaz03@epn.edu.ec

cdiaz_jl@hotmail.com.ar

Certifico que el presente trabajo de integración curricular fue desarrollado por Carlos Joel Díaz Rosero, bajo mi supervisión.

ING. Richard Paul Rivera Guevara, MSC.

DIRECTOR

richard.rivera01@epn.edu.ec

DECLARACIÓN DE AUTORÍA

A través de la presente declaración, afirmamos que el trabajo de integración curricular aquí descrito, así como el (los) producto(s) resultante(s) del mismo, son públicos y estarán a disposición de la comunidad a través del repositorio institucional de la Escuela Politécnica Nacional; sin embargo, la titularidad de los derechos patrimoniales nos corresponde a los autores que hemos contribuido en el desarrollo del presente trabajo; observando para el efecto las disposiciones establecidas por el órgano competente en propiedad intelectual, la normativa interna y demás normas.

Carlos Joel Díaz Rosero

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

El presente documento detalla y explica a fondo el funcionamiento y uso del *Keylogger* que se ha desarrollado y tiene como por objetivo demostrar lo vulnerable que puede ser el sistema operativo Windows 11 ante un spyware. Cuenta con un instalador el cual realiza el proceso de transferencia de los archivos del *Keylogger* en el computador para ser infectado, después de su instalación se ejecutará cada vez que se inicie Windows. Se diseño para que funcione en segundo plano y no muestre mensajes en pantalla de tal manera que pase desapercibido por el usuario infectado. Después del proceso de interceptar pulsaciones se genera la información, esta se puede ver de manera local donde los registros se encuentren dentro de archivos de texto y las capturas en imágenes con formato JPEG. Se estableció para sea una aplicación de consola con el lenguaje de programación C++ empleando bibliotecas nativas de Windows para que sea compatible con las computadoras que se desea infectar.

Este documento se explican todos los aspectos que conllevaron el proceso de desarrollo del Keylogger, por otro lado, se implementa la metodología Scrum, la misma que aporta con un marco de trabajo para el desarrollo ágil. Se detalla el patrón arquitectónico, la justificación de las herramientas usadas y se muestran los resultados obtenidos durante cada iteración de la codificación con su respectiva prueba de aceptación. También se encuentran las conclusiones y recomendaciones basadas en los resultados obtenidos durante el proceso que soporto la elaboración de este proyecto.

PALABRAS CLAVE: keylogger, spyware, Windows, software, seguridad, aplicación de consola.

ABSTRACT

This document details and explains in depth the operation and use of the Keylogger that has been developed and aims to demonstrate how vulnerable the Windows 11 operating system can be to spyware. It has an installer which performs the process of transferring the Keylogger files on the computer to be infected, after installation it will run every time Windows starts. It is designed to run in the background and not display messages on the screen so that it goes unnoticed by the infected user. After the process of intercepting keystrokes, information is generated and can be viewed locally where the logs are in text files and the screenshots in JPEG format images. It was established to be a console application with the C++ programming language using native Windows libraries to be compatible with the computers to be infected.

This document explains all the aspects involved in the development process of the Keylogger, on the other hand, the Scrum methodology is implemented, which provides a framework for agile development. The architectural pattern is detailed, the justification of the tools used, and the results obtained during each iteration of the coding with its respective acceptance test are shown. Conclusions and recommendations based on the results obtained during the process that supported the development of this project are also included.

KEYWORDS: keylogger, spyware, software, console application, security, Windows.

1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO

El desarrollo que se ha llevado a cabo de este componente permite que una persona pueda capturar las interacciones que se generan en una computadora mediante los eventos que suceden entre el teclado y mouse [1]. Este tipo de software es denominado como un *keylogger* [2], un spyware malicioso [3] que está diseñada para ser invisible lo cual lo convierte en un atacante sigiloso en el dispositivo que haya infectado [4] que puede ser utilizado como una herramienta de espionaje [5], en distintos dispositivos [6], tanto computadoras como dispositivos móviles [7], todo esto se realiza con el objetivo de recopilar la información personal de la víctima [8].

El keylogger es una herramienta maliciosa que tiene la capacidad de interceptar y almacenar las pulsaciones realizadas en el teclado de un equipo infectado [9]. Para ello, este malware se sitúa entre el teclado y el sistema operativo para que su funcionamiento no sea notado por el usuario [3], luego almacena los datos recolectados de forma local o permite que el atacante pueda almacenar los datos en otro equipo [10]. Mediante el uso de Hook dentro del sistema de Windows es posible detectar e identificar cada vez que se genera una pulsación en cualquiera de los dos periféricos. Esta aplicación maliciosa ha sido desarrollada en el lenguaje de programación de C++ [11], con la cual se implementaron algunas librerías de Windows que permiten hacer subrutinas que permitan capturar la información que el usuario vaya generando mediante el uso del computador.

Los Hook son la plataforma donde se procesa los mensajes de Windows, lo cual ayuda a obtener e interpretar el mensaje o evento que se haya enviado antes que este llegue a la ventana de destino [12]. Este mecanismo permite que se procese toda acción interceptada por otras aplicaciones y se ejecuten un conjunto de eventos. De tal manera que en el Keylogger se emplea este tipo de mecanismo para capturar, analizar, almacenar e incluso enviar esta información hacia un servidor externo. Según el usuario vaya interactuando con la computadora esta aplicación comenzara a ejecutarse de tal manera que logre su acometido.

Los registros generados a partir de la interceptación de los eventos generados nos permiten crear funciones adicionales que permitan tomar capturas en donde se pueda tener una referencia visual sobre el uso que da el usuario infectado en su computador. De esta manera se puede tener una mayor claridad al interpretar los registros generados a lo largo de la ejecución de esta herramienta maliciosa.

En cuanto al proyecto este es desarrollado en una aplicación de consola con el lenguaje de programación de C++. En el cual se ha implementado funciones que permiten crear un

directorio dentro de la carpeta personal del usuario, en donde servirá para almacenar los registros de pulsaciones e incluso la captura de pantalla. De tal manera que se pueda llevar un registro ordenado de la información generada por el usuario. Adicional a esto se han implementado métodos que pueden enviar los registros hacia un servidor fuera de la red del usuario infectado. Esto se realiza por medio de peticiones HTTP hacia una API la cual recibirá los métodos POST que vayamos generando a partir de los eventos que se van suscitando durante el uso de la computadora. A continuación, se explica a detalle cada una de estas funciones y su implementación.

1.1 Objetivo general

Desarrollo de un Keylogger para Windows 11.

1.2 Objetivos específicos

- OBJ 1: Determinar los requerimientos funcionales y no funcionales del keylogger.
- OBJ 2: Diseñar el modelo de la base de datos y prototipo de interfaces.
- OBJ 3: Implementar los módulos del *keylogger*.
- OBJ 4: Probar el funcionamiento del *keylogger*.

1.3 Alcance

El componente debe cumplir ciertos requisitos para satisfacer las expectativas que se tiene sobre el *keylogger*, por tanto, se ha realizado un análisis sobre las herramientas que se requiere para el desarrollo de esta herramienta maliciosa. De acuerdo con esto se ha establecido usar el lenguaje de programación C++ [11] de tal forma que se emplearan bibliotecas y librerías que permitirán obtener funciones necesarias para el proyecto y se ejecute de manera eficiente. Logrando interceptar las pulsaciones que realice un usuario y las capturas de pantalla en Windows 11 [13]. Su almacenamiento se lo realizara de manera local sin emplear alguna base de datos, ara ello se ha establecido manejar archivos de texto donde se registrarán todas las pulsaciones realizadas por el teclado y por el mouse y para la captura de imágenes se ha decidido utilizar el formato JPEG [14], donde las capturas de pantalla se almacenarán dentro de un directorio.

Las bibliotecas de ha usarse para el desarrollo se ha decidido emplear bibliotecas nativas de Windows para evitar tener que usar bibliotecas externas lo cual puede dificultar el uso en otras computadoras. La biblioteca más importante de este proyecto es Windows.h la cual nos permite acceder y gestionar el uso de los *Hooks*. Para la captura de pantalla y la

gestión de formatos de la imagen se ha empleado la librería gdisplus.h la cual estará en constante ejecución para realizar la conversión en memoria del archivo de imagen. Para las peticiones HTTP se ha implementado el uso de la biblioteca Winhttp.h, con la cual se gestionará los métodos POST lo que permite que el *keylogger* se comunique con la API.

La aplicación se la realizará con la metodología Scrum [15], ya que nos permite tener flexibilidad en la adopción de cambios y nuevos requisitos. Adicionalmente se podrá asegurar buenos resultados en cada iteración realizada ya que es posible ajustar el proceso del proyecto para mitigar los riesgos que pueden surgir.

1.4 Marco Teórico

La metodología, según el diccionario de la RAE lo define como “Ciencia del método” o “Conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal” [16]. Por ello podemos inferir que es la ciencia que efectúa de manera eficaz para obtener los resultados deseados, obteniendo una planificación a seguir durante el proceso de investigación [17].

Dentro del desarrollo de software se requiere una gestión flexible, eficiente y que satisfaga las necesidades del cliente [18], por ello se ha denominado a todo aquello que permita la adecuación de la forma de trabajo a las circunstancias requeridas para la elaboración del proyecto, permitiendo el desarrollo inmediato de acuerdo con las exigencias del proyecto como metodologías agiles [19].

La metodología ágil nació en 2001, como solución a los enfoques de cascada que se solía implementar en la gestión de proyectos. El documento que se redactó por algunos desarrolladores de software se lo llamo el Manifiesto para el Desarrollo Ágil de Software. Dentro de él se detalla cuatro características fundamentales que siempre deben estar presentes: Las personas y las interacciones antes que los procesos y las herramientas, El software en funcionamiento antes que la documentación exhaustiva, La colaboración con el cliente antes que la negociación contractual, La respuesta antes el cambio antes que el apego a un plan [20].

Las aplicaciones de consola se consideran todas aquellas que toman la entrada y muestran una salida mediante una consola de línea de comandos, es decir estas no usan una interfaz gráfica de usuario. En el caso de Windows se invoca desde el símbolo de sistema de Windows, esta aplicación se ejecuta con una interfaz gráfica mínima o nula [21]. Entre los lenguajes de programación que se utiliza para el desarrollo de aplicaciones de consola se encuentra C++ el cual apareció en 1983, es una extensión de C y ofrece características

orientadas a objetos con el tiempo este se ha convertido en un lenguaje de programación multiparadigma. Adicional a esto ofrece acceso a memoria de bajo nivel y compila directamente a las instrucciones de máquina [22].

2 METODOLOGÍA

La metodología Scrum es el marco de gestión de proyectos ágil, por lo tanto, con el uso de esta metodología se puede dimensionar de mejor manera el proyecto con lo cual se pueda satisfacer las necesidades y problemáticas brindando soluciones eficientes al cliente y beneficiando al equipo de desarrollo por cada retroalimentación en respuesta de cada iteración generada [23].

2.1 Metodología de Desarrollo

2.1.1 Roles

En Scrum los roles son sumamente importantes e indispensables para el desarrollo eficiente del proyecto, ya que cada uno tiene responsabilidades y tareas asignadas que se deben cumplir para brindar un desempeño adecuado durante cada iteración. Debe existir una buena comunicación y compromiso de parte de cada miembro para tener éxito en el desarrollo del proyecto [24].

Product Owner

Es la persona que conoce el negocio y es el encargado del producto en sí. Es el propietario del *Product Backlog*, es decir, de toda la lista de requisitos, y es la persona que prioriza esas tareas [24]. El Dr. Richard Rivera al ser la persona que en primera instancia planteo la idea con el equipo de desarrollo y al tener conocimiento sobre el funcionamiento y los resultados esperados del *keylogger* cumplirá con este rol.

Scrum Master

Es el responsable de asegurar que se adopte y se entienda la metodología Scrum; además de ser un líder dentro del Equipo Scrum, ayudando a entender que interacciones pueden ser útiles y cuales no [24]. El Dr. Richard Rivera cumple este rol, donde su experiencia le permite proveer ideas y soluciones que permiten resolver inconvenientes presentados durante el desarrollo del proyecto.

Development Team

Conforman todos los profesionales que participan de manera directa en realizar el trabajo de entregar un producto terminado de acuerdo con cada incremento de producto [24]. En este caso está conformado por una persona, el estudiante y programador Carlos Diaz, el cual se encuentra comprometido en ejecutar cada una de las tareas designadas en cada

Sprint. Teniendo en cuenta las características y requerimientos que debe tener el producto para satisfacer al cliente.

Asignación de Roles

La **Tabla I** presenta los roles establecidos dentro del desarrollo del proyecto siguiendo la ya mencionada metodología Scrum.

Tabla I. Roles del Proyecto

Roles	**Miembros**
Product Owner	Dr. Richard Paul Rivera Guevara
Scrum Master	Dr. Richard Paul Rivera Guevara
Development Team	Sr. Carlos Joel Díaz Rosero

2.1.2 Artefactos

Dentro de Scrum los artefactos son todos aquellos elementos físicos que se realizan y son el resultado de la aplicación de esta metodología, con los cuales se asegura el registro de las actividades y la información importante de la gestión empleada para el proyecto.

Recopilación de Requerimientos

Es el proceso donde se define y sirve para documentar las necesidades del interesado para con ello cumplir los objetivos del proyecto. Estos requisitos pueden incluir las necesidades, expectativas y deseos del cliente. Donde todas estas se deben analizar a detalle con lo cual se pueda medir los avances del proyecto mientras este se desarrolle [25]. Los requerimientos del Keylogger se realizaron después de recolectar la información necesaria con el *Product Owner* y posteriormente el equipo de desarrollo la analizara. Estos resultados se pueden ver a detalle en el *ANEXO II.* Manual Técnico

Historias de Usuario

Es la descripción corta y precisa que se le da a una función de software desde la perspectiva del usuario final, esto quiere decir que no se utiliza un lenguaje técnico. El fin es el de proporcionar las características detalladas del valor que tiene esta función para el cliente [12].

Posteriormente, la **Tabla II** es una representación sobre una Historia de Usuario que se define una de las funcionalidades del *keylogger,* las demás historias se encuentran en el ANEXO II sección *Historias de usuario*.

Tabla II. Historia de usuario Nro. 1 – Interceptar pulsaciones del teclado

HISTORIAS DE USUARIO			
Identificador (ID):	HU001	**Usuario:**	Atacante
Nombre de la historia:	Interceptar pulsaciones del teclado.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:	1		
Responsable:	Carlos Díaz		
Descripción: El usuario del k*eylogger* podrá interceptar las pulsaciones realizadas por el teclado de manera que se registre las teclas alfanuméricas y teclas especiales.			
Observación: El k*eylogger* tendrá la capacidad de identificar las teclas minúsculas y mayúsculas, números, caracteres especiales y teclas de mando.			

Product Backlog

Es el inventario en el cual se registra cualquier tipo de trabajo a realizarse. Contiene toda la información sobre los requerimientos que se van a implementar, esta lista es el resultado del trabajo del *Product Owner* con el Cliente. Este solo puede ser gestionado por el *Product Owner* de tal manera que se lleve un control acerca del estado en el que se encuentra la implementación del producto y también se priorice las principales funciones que tienen más importancia y valor para el Cliente [26]. El *Product Backlog* del k*eylogger* se encuentra en el ANEXO II sección

Product Backlog.

Sprint Backlog

Es la lista de elementos que se van a desarrollar en cada *Sprint*, cada uno está compuesto de tareas técnicas que logran obtener incrementos en cada iteración. Permite visualizar el trabajo que debe realizarse durante cada *Sprint* y está gestionado por el equipo de desarrollo con el fin de mantener una transparencia para el desarrollo del proyecto. Otra función de esta lista es proporcionar el estado que tiene cada elemento e identificar la evolución del trabajo durante cada iteración, además de ver si se encuentra en prueba, despliegue o completamente terminado, también se puede obtener información sobre los miembros del equipo de desarrollo que están trabajando en el elemento [27]. El *Sprint Backlog* del *keylogger* se encuentra en el ANEXO II sección

Sprint Backlog.

2.2 Diseño de interfaces

El *keylogger* es una aplicación de consola por tanto no requiere que se diseñe ni se implemente una interface, ya que esta herramienta nunca va a ser vista por el usuario.

2.3 Diseño de la arquitectura

Una vez finalizado con el levantamiento de requerimientos del *keylogger*, debemos proceder con la arquitectura que llevará este, a continuación, se describe el patrón de la arquitectura que tendrá este software. El cual estará constituido por una lógica de seguimiento y con su respectiva estructura de registro de datos.

2.3.1 Arquitectura de Datos

El *keylogger* no requiere de una base de datos, pero para el almacenamiento local de los archivos se ha establecido utilizar archivos de texto tanto para las capturas del teclado como para las del *mouse* estos son generados al cambiar la ventana activa en cada caso. Para el caso de las imágenes se ha establecido guardar en el formato JPEG. Todo esto dentro del directorio (C:\Users\"Nombre_del_usuario"\.temp_data) dentro de este directorio se crearán 3 carpetas como se lo ve en la **Fig. 1**, las cuales servirán para el almacenamiento de los datos capturados.

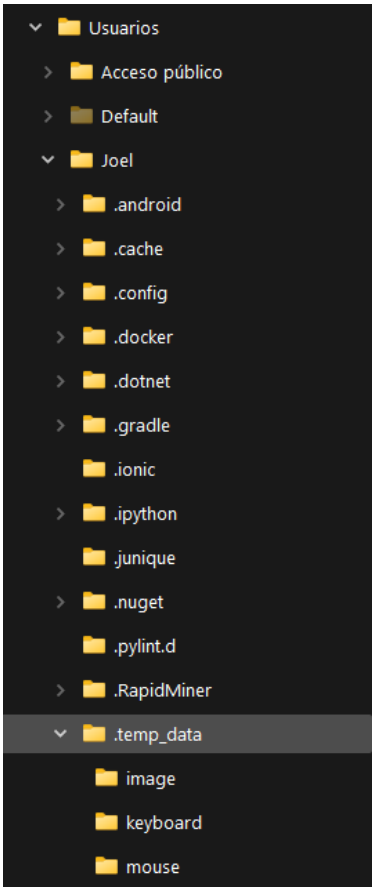

Fig. 1 Directorio del almacenamiento de datos del Keylogger

Por otra parte, para realizar él envió de la información a un servidor externo esto se lo va a realizar mediante los *endpoint,* donde uno de ellos recibirá la información del usuario y computadora infectada y también de cada uno de los registros creados durante su funcionamiento. Para el caso del registro del cliente se lo realizara enviando tres valores: el *id*, *nickname* y *desktop_name* tal como lo vemos en la **Fig. 2**.

```
1  {
2      "id":"Joel-Desktop_Prueba",
3      "nickname":"Joel",
4      "desktop_name":"Desktop-prueba"
5  }
```

Fig. 2 Estructura JSON – *Endpoint* Cliente

En cambio, para él envió de los registros los cuales pueden ser *keystroke*, *website* o *screenshot*, va a ser necesario enviar siete valores: *app_name, window_name, date, time, type, content* y *client_id* tal como se puede apreciar en la **Fig. 3**.

```
{
    "app_name":"prueba",
    "window_name": "postman",
    "date": "2022-01-31",
    "time":"17:00:00",
    "type":"keystroke",
    "content":"[asad]-Esto es una prueba desde postman",
    "client_id": "Joel-DESKTOP-541R5S3"
}
```


Fig. 3 Estructura JSON – *Endpoint* Registros

2.3.2 Patrón arquitectónico Bus de Eventos

El patrón Bus de Eventos es un modelo que sirve para diseñar aplicaciones basadas en eventos. Los sistemas que utilizan este modelo aprovechan la detección y consumo de eventos de esa forma logra reaccionar a ellos realizando acciones o tareas que se relacionan con dicho evento [28].

Evento: Son todos los sucesos o cambios del estado de hardware o software dentro de un sistema. Estos se originan por algún estimulo interno o externo, por ejemplo, pueden ser causados por algún periférico como un teclado o mouse [28].

A continuación, se muestra la **Fig. 4**, la cual representa el patrón descrito en función del proyecto del *Keylogger*.

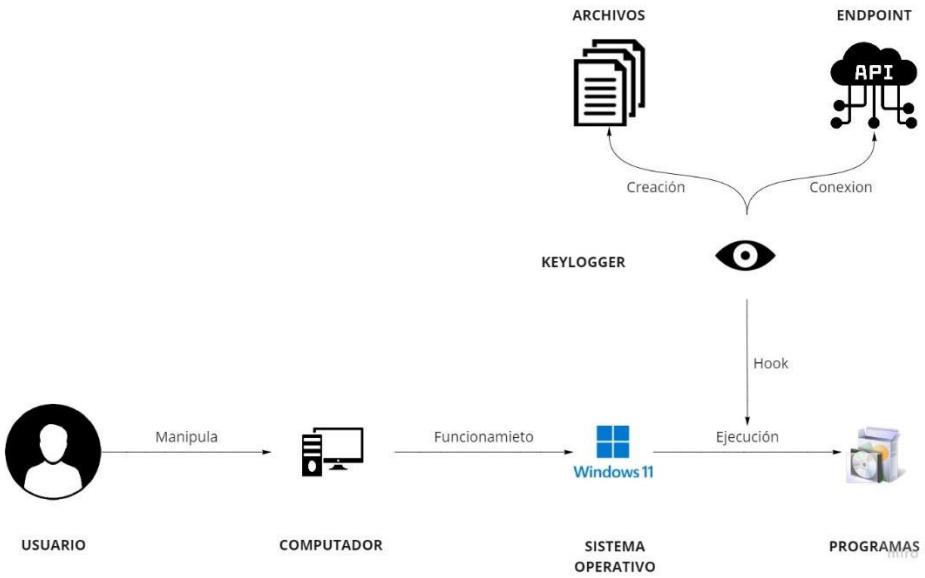

Fig. 4 Modelo Bus de Eventos

2.4 Herramientas de desarrollo

Para el desarrollo del proyecto se ha establecido herramientas que tienen como objetivo principal facilitar, optimizar y aumentar el desempeño del flujo de trabajo, por tanto, el resultado de este proyecto será más productivo [29]. La **Tabla III** detalla las herramientas de desarrollo que se ha utilizado para el desarrollo del *keylogger*.

Tabla III. Herramientas y librerías para el desarrollo de la aplicación de consola

Herramienta	Descripción	Justificación
Visual Studio Professional 2022	Es el entorno de desarrollo integrado para Windows y MacOS, cuenta con compatibilidad para una gran variedad de lenguajes de programación y *frameworks*. Esta desarrollado por Microsoft y su última versión estable fue lanzada en 2022 [30].	Se decidió usar este IDE ya que soporta el desarrollo de aplicaciones de consola con facilidad con el lenguaje de programación C++, permite compilar, depurar y diagnosticar el código escrito, permitiendo escribir código sin errores y que sea lo más eficiente posible.
Windows API	Esta librería tiene funciones de las bibliotecas DLL, las cuales permiten que se ejecute una aplicación en Windows, las cuales permiten acceder a información necesaria para el objetivo del proyecto [31].	Se ha decidido utilizar esta librería ya que nos permite acceder a parámetros dentro del sistema de Windows con los cuales podremos realizar la implementación del *keylogger*.
Process.h	Es una biblioteca de C que permite trabajar con subprocesos y procesos [32].	El empleo de esta biblioteca permite crear hilos de procesos para diferentes tareas del *keylogger* permitiéndole ser más eficiente en su ejecución.
Filesystem	Es la biblioteca que permite realizar operaciones en el	Se ha decidido el uso de la biblioteca para implementar

	sistema de archivos, entre ellas crear directorios, archivos normales y establecer rutas de acceso [33].	las rutas para acceso y directorios necesarios para el almacenamiento de los registros.
fstream	Permite el trabajo con ficheros en el lenguaje de programación C++, permite leer, escribir o ambos al mismo tiempo [34].	El uso de esta librería permite trabajar con archivos necesarios para el funcionamiento y registro de información del *Keylogger*.
Chrono	Es la librería que permite trabajar con el tiempo [35].	Se decidió usar esta librería para implementar temporizadores necesarios para el cumplimiento de ciertas tareas para el *keylogger*.
Iomanip	Esta biblioteca se utiliza para definir el ancho de la biblioteca ios, donde mediante un parámetro es capaz de manipular el flujo [36].	Se decidió el empleo de esta biblioteca para manipular y dar formato algunos elementos de salida.
Iostream	Es la biblioteca que permite realizar operación en la entrada y salida de datos [37].	El uso de esta biblioteca permite trabajar con las entradas y salidas de datos lo cual es importante para el funcionamiento del *keylogger*.
Functional	Biblioteca que brinda un conjunto de plantillas definidas para objetos, clases, operaciones aritméticas y lógicas. Contiene muchas funciones	Se decidió usar esta biblioteca para poder realizar y dar formato a fechas y horas simplificando el código y haciéndolo más eficiente.

	polimórficas para uso general [38].	
unordered_map	Es una biblioteca que permite el trabajo de elementos de longitud variable, mediante funciones hash con las cuales divide el elemento principal en secuencias llamadas *buckets* [39].	El uso de esta librería permite almacenar las configuraciones del archivo conf de manera más eficiente.
sstream	Esta biblioteca permite que un string sea tratado como un stream facilitando la implantación y extracción como si estuviésemos usando cin y cout [40].	Se decidió el uso de esta librería por la facilidad que permite al trabajar con variables de tipo string.
Lmcons.h	La biblioteca es empleada para establecer tamaños de buffer [41].	El uso de esta biblioteca permitió trabajar de manera eficiente con la extracción de nombres y características del sistema de Windows.
Winhttp	Es una biblioteca de Microsoft que permite realizar peticiones HTTP en la web [42].	Se decidió usar esta librería para realizar las peticiones mediante métodos post hacia la API.
Shlwapi	Es una biblioteca de enlaces dinámicos desarrollada por Microsoft, tiene funciones que permiten gestionar procedimientos de control [43].	El empleo de esta biblioteca facilito el uso y manipulación de datos al realizar las peticiones HTTP.
Conio.h	La biblioteca contiene funciones para optimizar la	Se decidió el uso de esta biblioteca para manipular y

	entrada y salida de datos [44].	gestionar de manera eficiente la salida de datos.
Postman	Es la aplicación que ejecuta pruebas a un API. De tal manera que se pueda testear el funcionamiento mediante la interfaz gráfica que dispone [45].	Se empleo esta herramienta que permite el testeo de una API para comprobar el funcionamiento de esta y poder continuar con la implementación de procesos de peticiones HTTP.
GitHub	Es la herramienta de control de versiones, la cual ayuda a la gestión y exploración de cada cambio realizado [46].	El uso de esta herramienta fue necesario para la controlar los camios realizados y poder gestionarlos a futuro.

3 RESULTADOS

Esta sección detalla la implementación y resultados obtenidos en cada actividad de manera resumida. Además de la demostración del despliegue y pruebas realizadas.

3.1 Sprint 0. Configuración del ambiente de desarrollo

De acuerdo con la planificación del *Sprint Backlog*, Este *Sprint* contiene todas las tareas que se requieren para la implementación del entorno de desarrollo.

A continuación, se detalla las tareas que corresponden a este Sprint:

- Instalación y configuración del Visual Studio 2022.
- Creación del proyecto de aplicación de consola.
- Creación del repositorio en GitHub.
- Instalación de Postman.

3.1.1 Instalación y configuración del Visual Studio 2022

Para el desarrollo del proyecto se va a instalar el IDE de programación Visual Studio 2022, el cual permite tener un entorno de desarrollo para el lenguaje de C++. Para ello se necesita seleccionar el lenguaje de programación C++ en su módulo de instalación. Para ello nos descargamos el instalador desde la página oficial y al abrirlo nos mostrara una interfaz visual como en la **Fig. 5**.

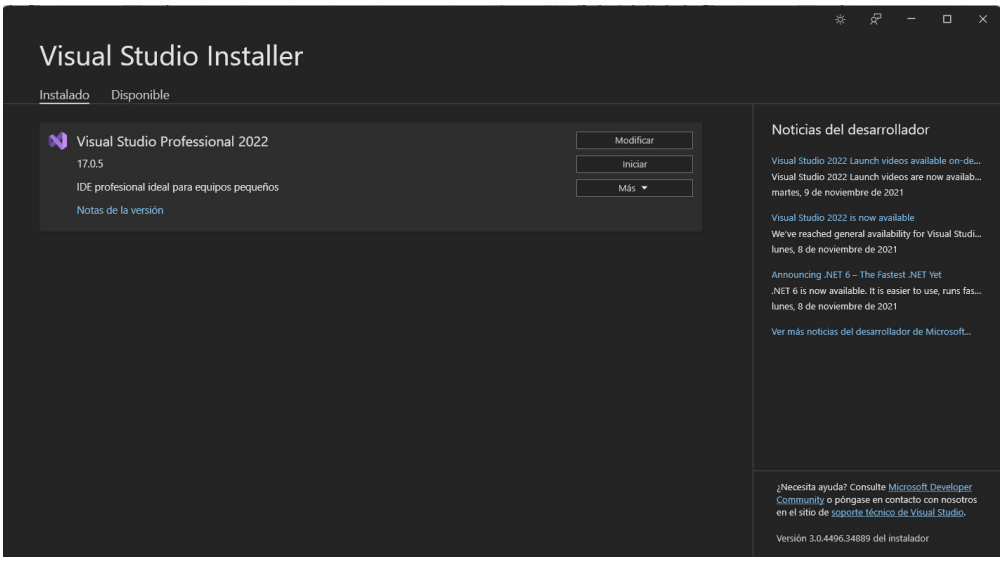

Fig. 5 Instalador de Visual Studio

Seleccionamos la opción que necesitamos para el desarrollo de la aplicación de consola en nuestro lenguaje de programación como se puede apreciar en la **Fig. 6**.

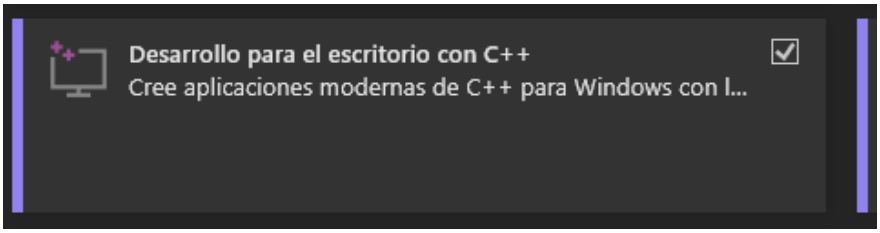

Fig. 6 Selección del lenguaje de programación con el tipo de proyecto a desarrollar

Una vez terminado de seleccionar las opciones a instalarse deberemos dar clic sobre el botón instalar, esto puede tardar unos minutos ya que descarga los archivos necesarios para la instalacion, como se puede ver en la **Fig. 7**.

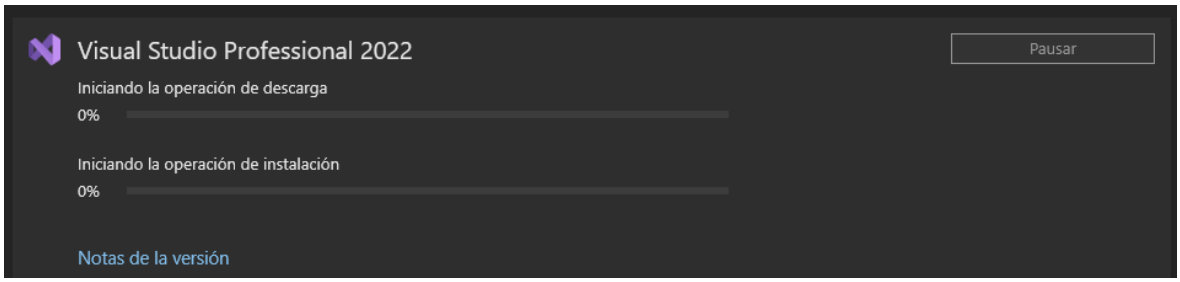

Fig. 7 Proceso de instalación de Visual Studio 2022

3.1.2 Creación del proyecto de aplicación de consola.

Creación del proyecto de aplicación de consola.

Para la creación de la aplicación de consola es necesario seleccionar “Crear un proyecto” y la opción de Aplicación de consola en el lenguaje de programación C++. Como se muestra en la **Fig. 8**.

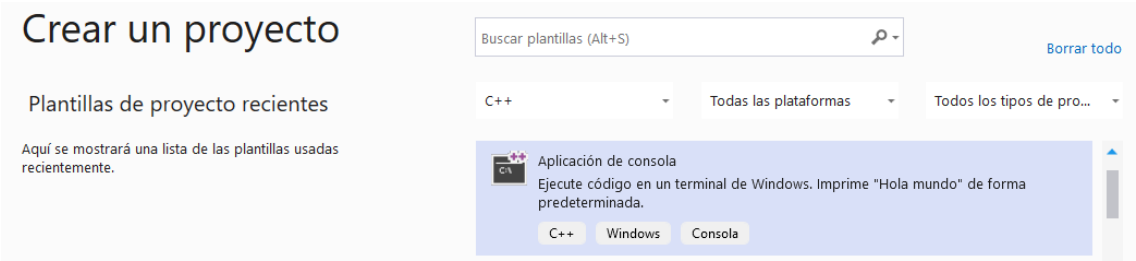

Fig. 8 Creación del proyecto – Aplicación de consola

Se configura el proyecto dándole el nombre y su ubicación, una vez finalizado se podrá observar el despliegue del IDE con nuestro proyecto. En la **Fig. 9**, se puede ver como finaliza la configuración del proyecto e inicia el IDE.

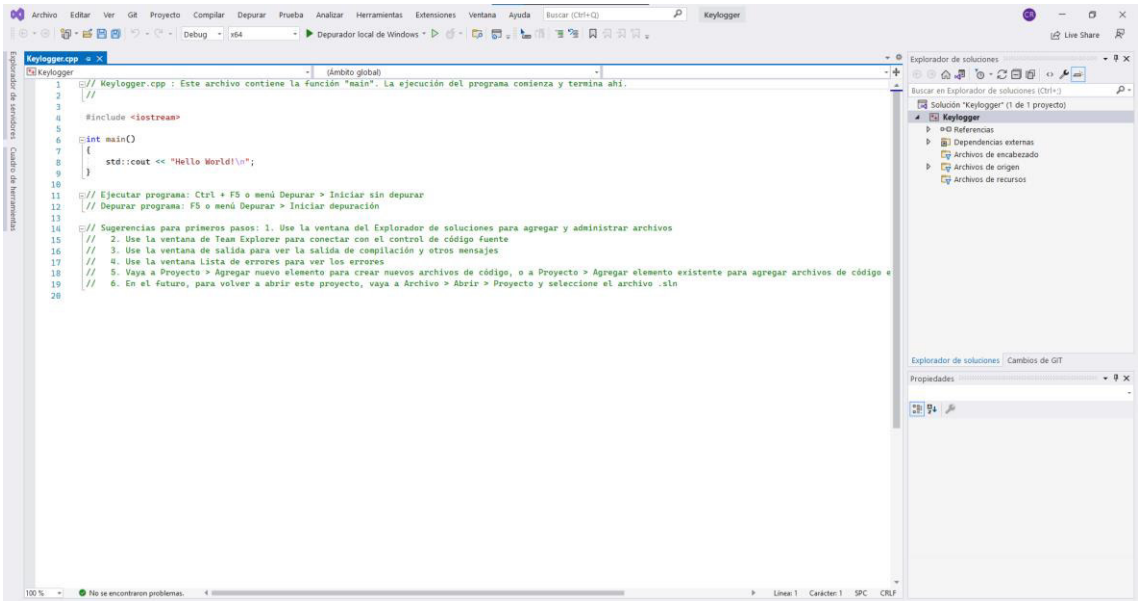

Fig. 9 Visual Studio 2022 – Aplicación de consola Spy

3.1.3 Creación del repositorio en GitHub.

Iniciamos sesión en GitHub y creamos un nuevo repositorio en nuestro caso le vamos a dar el nombre de *Cpp-Keylogger* tal como se lo puede ver en la **Fig. 10**.

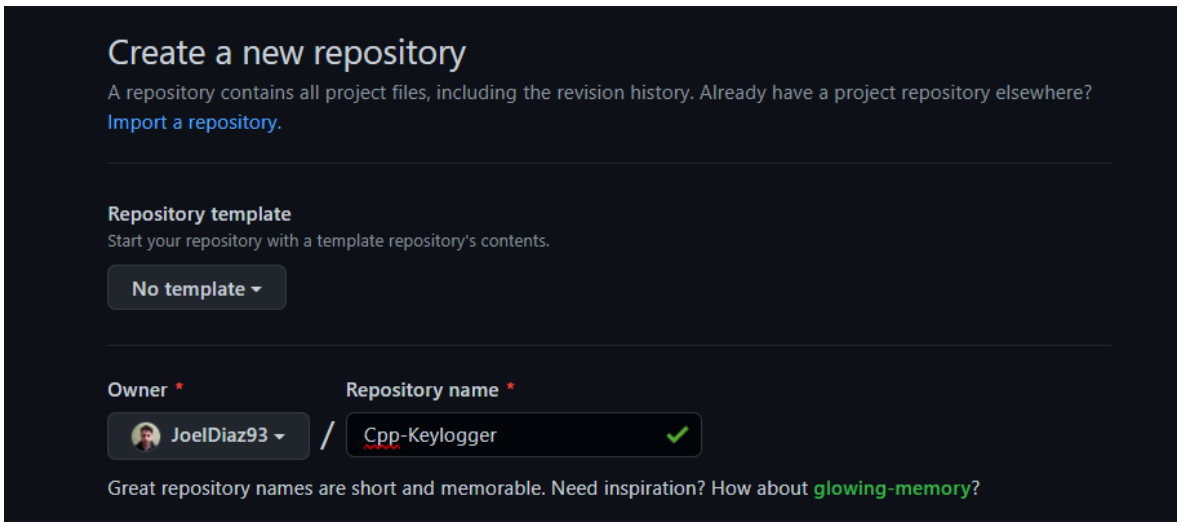

Fig. 10 Creación de un repositorio en GitHub – *Cpp-Keylogger*

Configuramos nuestro archivo (.gitignore) en el cual se va a excluir dos directorios como se lo ve en la **Fig. 11**.

```
# See https://help.github.com/articles/ignoring-files/
for more about ignoring files.

/.vs
/x64
```

Fig. 11 Archivo .gitignore

Para finalizar esta sección subiremos los archivos que se disponen hasta el momento mediante el comando git como en la **Fig. 12**.

```
 8 git remote add origin https://github.com/JoelDiaz93/Cpp-Keylogger.git
 9 git .add
10 git add .
11 git commit -m "Keylogger"
12 git push -u origin main
```

Fig. 12 Comandos git para subir el repositorio

3.1.4 Instalación de Postman.

Para la instalación de esta aplicación es necesario descargar el instalador desde la página del desarrollador en la versión requerida para nuestro sistema operativo. Seguir las instrucciones de instalación y al culminar se puede ejecutar esta herramienta para el testeo de API.

3.2 Sprint 1. Interceptar acciones del usuario

En la planificación del Sprint Backlog, el Sprint 2, se tiene definido las actividades para la interceptación de acciones del usuario en donde se utilizarán algunas de las librerías definidas anteriormente.

A continuación, se listan los resultados obtenidos después de haber realizado cada una de las tareas destinadas para este Sprint.

- Interceptar las pulsaciones del teclado.
- Interceptar las pulsaciones del mouse.
- Realizar capturas de pantalla por evento del mouse.
- Realizar capturas de pantalla por evento de la tecla enter.
- Realizar capturas de pantalla por evento de cambio de aplicación.
- Realizar capturas de pantalla cada cierto tiempo.

3.2.1 Interceptar las pulsaciones del teclado.

Para esta tarea es necesario crear clases para cada funcionalidad requerida en el *Keylogger*, por ello vamos a crear la clase *KeyboardHook*. Esta clase contara con un archivo de cabecera y otro de código fuente con el mismo nombre, pero con extensiones diferentes como se lo puede ver en la **Fig. 13.**

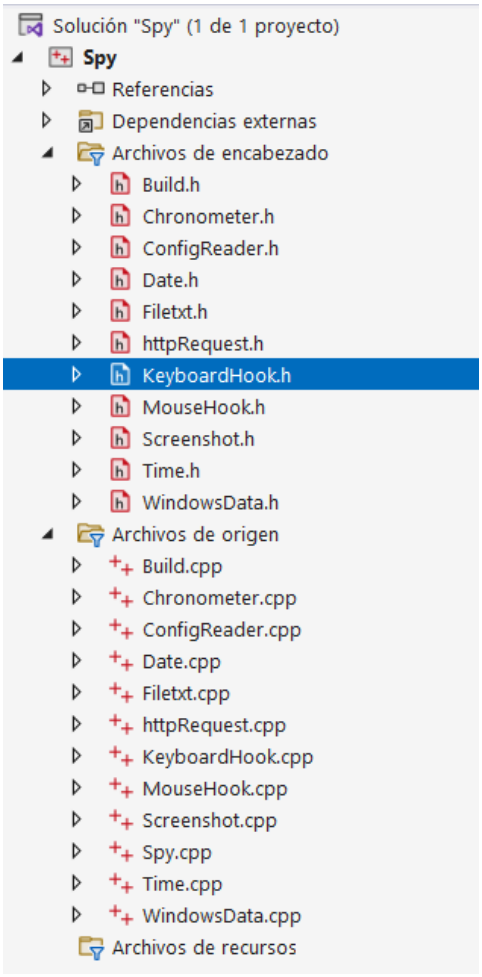

Fig. 13 Solución de la aplicación de consola - Spy

Esta clase se encargará en detectar las pulsaciones realizadas en el teclado mediante un *Hook*, esto quiere decir que mediante la librería *Windows.h* vamos a detectar cada vez que se realice una pulsación con el teclado y esta va a ser interceptada para su posterior implementación de llamadas a funciones como registrar la tecla pulsada dentro de una variable. Es necesario crear una instancia para este *Hook* de manera que va a estar corriendo en todo tiempo a espera de la detección de una pulsación, cuando el programa se cierre es necesario que la instancia creada para el funcionamiento de este *Hook* sea

eliminada por ello se implementó una función para ello. Como resultado de esta tarea podemos ver las funciones que componen a esta clase en la **Fig. 14**.

```
#pragma once
#include <iostream>
#include <Windows.h>
#include <Lmcons.h>
#include "string"
#include <sstream>

class KeyboardHook
{
public:
    static KeyboardHook& Instance() {
        static KeyboardHook keyboardHook;
        return keyboardHook;
    }
    void InstallHook();
    void UninstallHook();
    MSG msg;
    int Messsages();
    HHOOK MyKeyboardHook;

private:

};
```

Fig. 14 Clase *KeyboardHook* – Spy

3.2.2 Interceptar las pulsaciones del mouse.

Para la realización de esta tarea también se elaboró una clase llamada *MouseHook* con sus respectivos archivos como se lo puede ver en la Fig. 13. Al igual como se elaboró las funciones necesarias para la clase *KeyboardHook*, en este caso también necesitamos establecer unas funciones donde se pueda crear una instancia para el funcionamiento del Hook que se encargara de la intercepción de la actividad de los botones del mouse. En este caso solo registraremos las pulsaciones del botón derecho e izquierdo y de igual manera cuando se cierre el programa este debe eliminarse por ello es necesario establecer una función para que suceda esto y eliminar estos procesos de la memoria, esto se puede apreciar en la **Fig. 15**.

```
#pragma once
#include <iostream>
#include <Windows.h>
#include <Lmcons.h>
#include "string"
#include <sstream>

class MouseHook {
public:
    static MouseHook& Instance() {
        static MouseHook myHook;
        return myHook;
    }
    HHOOK hook;
    void InstallHook();
    void UninstallHook();

    MSG msg;
    int Messsages();
};
```

Fig. 15 Clase *MouseHook* – Spy

3.2.3 Realizar capturas de pantalla por evento del mouse.

Para realizar esta tarea se decidió crear una clase que permita realizar las capturas de pantalla y estas sean almacenadas en formato JPEG. Esta clase se llama *Screenshot* y tiene sus respectivos archivos para su funcionamiento, como en la **Fig.13**. En esta clase se cuenta con una variable en privado la cual se asigna cuando inicia el constructor de manera que se la llene con la dirección del directorio donde debe guardarse las imágenes que se vayan capturando, adicional cuenta con varias funciones que servirán de apoyo durante todo el proyecto como se lo puede ver en la **Fig. 16**.

```
1   #pragma once
2
3   #define _WIN32_LEAN_AND_MEAN
4   #include <string>
5   #include <vector>
6   #include <iostream>
7   #include <stdio.h>
8   #include <windows.h>
9   #include <gdiplus.h>
10  #include <time.h>
11  #include <direct.h>
12  #include <fstream>
13  #pragma comment( lib, "gdiplus" )
14  using namespace Gdiplus;
15
16  class Screenshot
17  {
18  public:
19      Screenshot(std::string dir);
20      int GetEncoderClsid(const WCHAR* format, CLSID* pClsid);
21      void saveJpeg(std::string nameFile);
22      std::string getBase64(std::string nameFile);
23      static std::string encodeFromFile(const std::string& inFileName);
24      static std::string encode(const std::vector<char>& data);
25
26  private:
27      std::string dirPath;
28
29  };
30
31
```

Fig. 16 Clase *Screenshot – Spy*

3.2.4 Realizar capturas de pantalla por evento de la tecla enter.

Para la ejecución de esta tarea se usó la clase *Screenshot* la cual se describió anteriormente y se puede ver sus funciones en la **Fig. 16.** Para ello llamaremos esta librería desde el archivo de código fuente de la clase *KeyboardHook* y estableceremos dentro de la detección de las teclas un condicional que filtre la pulsación de la tecla *enter*. Desde donde se llamará a la clase *Screenshot* y se genere el archivo JPEG con la captura de pantalla deseada. Cuando finalice este proceso el resultado se lo podrá observar en el directorio de imágenes (*C:\Users\Nombre_de_usuario\.temp_data\image*), como se lo puede ver en la **Fig. 17**.

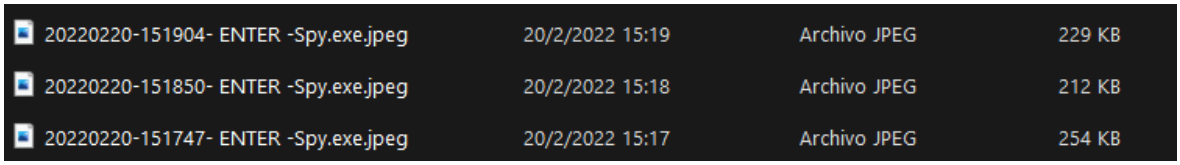

Fig. 17 Directorio *image* – Imágenes por evento pulsación de tecla *Enter*

3.2.5 Realizar capturas de pantalla por evento de cambio de aplicación.

En esta tarea reutilizaremos la clase *Screenshot* la cual se puede ver en la **Fig. 16.** Para lograr este objetivo es necesario capturar el nombre de la ventana activa donde en el momento que se realizaba el cambio de ventana, en este punto es donde utilizamos la función *saveJpeg* de tal manera que se genera una captura y se almacena en el directorio image (*C:\Users\Nombre_de_usuario\.temp_data\image*), se lo puede ver en la **Fig. 18**

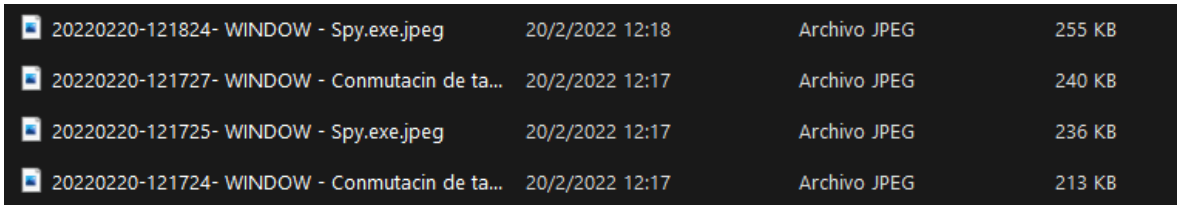

Fig. 18 Directorio *image* – Imágenes por evento de cambio de ventana activa

3.2.6 Realizar capturas de pantalla cada cierto tiempo.

Para cumplir con el objetivo de esta tarea es necesario elaborar un cronometro para ello se implementó una clase como se lo ve en la **Fig. 13** el cual permita activar las funciones de captura de pantalla como se lo ve en la **Fig. 16**, esto genera una imagen en formato JPEG y se almacena en el directorio asignado. Para ello la clase Chronometer emplea un constructor y tres funciones con las cuales se logra cumplir y satisfacer la necesidad de tomar la captura cada cierto tiempo, esto se muestra en la **Fig. 19**.

```
1   #pragma once
2   #define _CRT_SECURE_NO_WARNINGS
3   #include <chrono>
4   #include <memory>
5   #include <functional>
6   #include <iomanip>
7   #include <iostream>
8   #include <windows.h>
9
10  class Chronometer
11  {
12  public:
13      Chronometer(int minAlarm);
14      void setAlarm();
15      bool alarmChrono();
16      std::string formatTime(const std::chrono::system_clock::time_point& time, const std::string& format);
17
18  private:
19      int lapse;
20      std::chrono::system_clock::time_point alarm;
21      std::chrono::seconds time = std::chrono::seconds::zero();
22  };
23
```

Fig. 19 Clase Chronometer – Spy

Como resultado de este proceso y ejecución de funciones del keylogger, se obtiene una imagen en formato JPEG, como ejemplo se muestra el resultado en la **Fig. 20**.

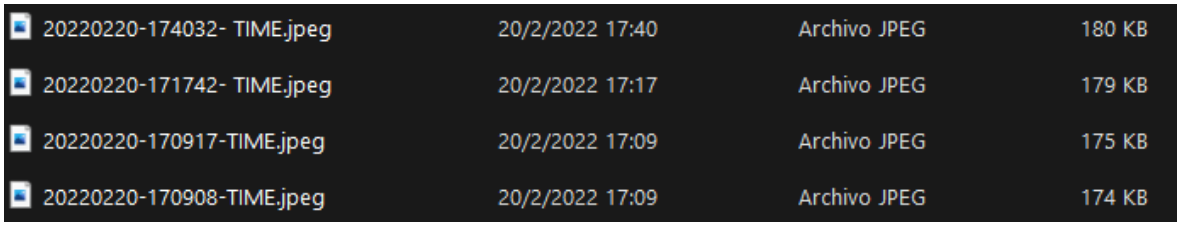

Fig. 20 Directorio *image* – Imágenes tomadas cada cierto tiempo por un cronometro

3.3 Sprint 2. Funciones de configuración y almacenamiento.

De acuerdo con lo establecido en el Sprint Backlog, en el Sprint 2, las actividades para desarrollarse en este módulo donde se realizarán funciones de vital importancia para el funcionamiento del Keylogger.

A continuación, se describen en una lista todos los resultados del Sprint:

- Configurar opciones de registro del Keylogger.
- Almacenar pulsaciones en un archivo de texto.
- Creación del directorio para almacenamiento.
- Funcionamiento en segundo plano.

3.3.1 Configurar opciones de registro del Keylogger.

Se decidió crear un archivo de configuración con el cual se pueda modificar el funcionamiento del Keylogger, por ello se decidió elaborar una clase WindowsData, Build y ConfigReader como se lo puede ver en **Fig. 13.** Donde la clase WindowsData se encarga de devolvernos el nombre del usuario, el nombre de la computadora y genera un id basado en los dos valores anteriores como en la **Fig. 21.** Para la obtención de estos valores lo realizamos mediante la librería Windows.h y se lo almacena en las variables privadas de la clase.

```cpp
#pragma once
#include <iostream>
#include <Windows.h>
#include <Lmcons.h>
#include "string"

using std::string;

class WindowsData
{
public:
    WindowsData();
    string getUserNamer();
    string getComputerName();
    string getID();

private:
    string userName;
    string computerName;
    string id;
};
```

Fig. 21 Clase WindowsData – Spy

En la clase *Build* se encarga de la creación del archivo .conf como en la **Fig. 22.** En este archivo de configuración se establece parámetros del usuario vistos en la **Fig. 21.** Este archivo se almacena en la raíz del directorio (.temp_data) desde donde se puede recuperar toda la información almacenada.

```cpp
#pragma once
#include <iostream>
#include <filesystem>
#include <fstream>

using std::system;
namespace fs = std::filesystem;

class Build
{
public:
    Build(std::string user);
    ~Build(){};
    void CreateDirectorySpy();
    void CreateFileConfig(std::string user, std::string computer, std::string id, std::string date, std::string time);
    bool Path();
    bool Keyboard();
    bool Image();
    bool Mouse();
    bool existConfig();
    std::string getPath();
    std::string getPathKeyboard();
    std::string getPathImage();
    std::string getPathMouse();

private:
    std::string directoryPath;
    std::string directoryPathKeyboard;
    std::string directoryPathImage;
    std::string directoryPathMouse;
};
```

Fig. 22 Clase Build – Spy

El archivo generado en esta clase se lo puede apreciar en la **Fig. 23** donde se ve todos los parámetros con los que cuenta y también una explicación breve de cada uno, además para

introducir comentarios dentro del archivo de configuración solo es necesario colocar un (#) antes del comentario a introducirlo.

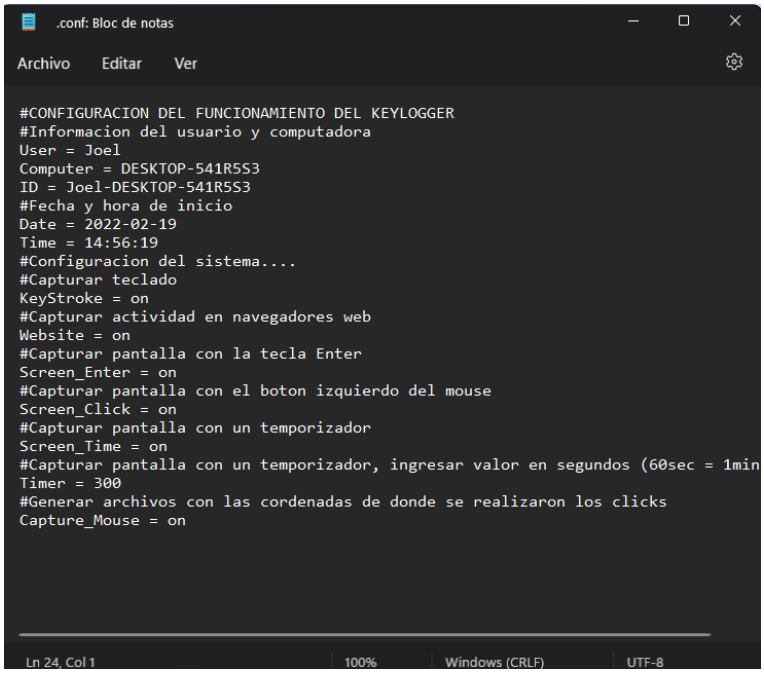

Fig. 23 Archivo de configuración .conf

La clase ConfigReader se encarga de recuperar los parámetros establecidos en el archivo (.conf) cuales ya se vio en la **Fig. 23,** mediante las funciones que se ven en la **Fig. 24** donde mediante sus funciones se extrae los valores del archivo y se lo almacena en una lista, a la cual se puede acceder y extraer el valor que se requiera obtener.

```
1   #pragma once
2   #include <fstream>
3   #include <iostream>
4   #include <sstream>
5   #include <string>
6   #include <unordered_map>
7   using std::istringstream;
8
9   class ConfigReader
10  {
11  public:
12      ConfigReader(std::string dir);
13      void readFile();
14      std::string getValue(std::string value);
15  private:
16      std::string dirConfig;
17      std::unordered_map<std::string, std::string> data;
18  };
19
20
```

Fig. 24 Clase ConfigReader – Spy

3.3.2 Almacenar pulsaciones en un archivo de texto.

La creación de archivos de texto para el almacenamiento de las pulsaciones de teclado y *mouse* requirió implementar ciertas librerías entre las más importantes fstream. Se elaboro una clase llamada Filetxt como se ve en la **Fig. 13.** Para ello se estableció una función que permita almacenar los datos que se generan mediante los *Hooks* de Teclado y *Mouse* de tal manera que se almacene en el directorio establecido para estos archivos, esto se lo puede ver en la **Fig. 25.**

```
1   #pragma once
2   #include "string"
3   #include <fstream>
4   #include <iostream>
5
6   class Filetxt
7   {
8   public:
9       Filetxt(std::string dirKey, std::string dirMouse);
10      void saveFile(std::string type, std::string name, std::string date, std::string time, std::string window, std::string content);
11
12  private:
13      std::string dirPathKey;
14      std::string dirPathMouse;
15  };
16
17
```

Fig. 25 Clase Filetxt – Spy

Los archivos que se generan para el teclado llevan en el nombre del archivo de texto la fecha, la hora y el nombre de la ventana donde se estuvo presionando esas teclas, de esta manera se puede asegurar que todos los archivos tendrán un nombre único y no se pueda repetir durante el funcionamiento del *keylogger*. El resultado de todo este proceso se lo puede ver en la **Fig. 26**

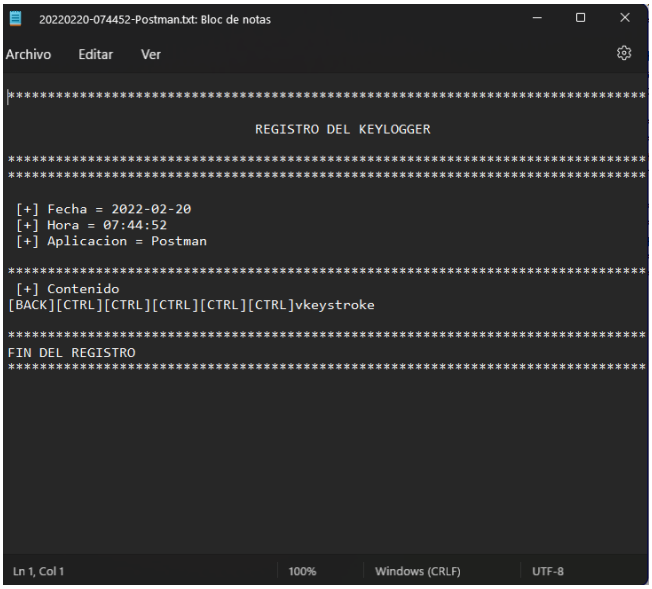

Fig. 26 Archivo de texto – Registro de pulsaciones de teclado

Mientras tanto para el registro de manera local de las pulsaciones de los botones del mouse su estructura es similar pero el archivo en el que se guarda toda la información va a generarse uno por día en donde se registrara las pulsaciones con las coordenadas donde se produjeron durante el uso del Keylogger, esto se lo puede en la **Fig. 27**.

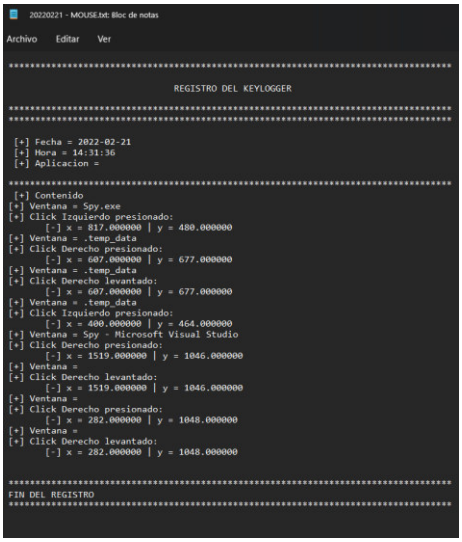

Fig. 27 Archivo de texto – Registro de pulsaciones del mouse

3.3.3 Creación del directorio para almacenamiento.

Para el almacenamiento de las capturas, recolección de datos por teclado y mouse es necesario establecer un lugar donde estos archivos se almacenen de manera local, para ello se necesita utilizar la clase Build como se lo ve en la **Fig. 13.** Donde sus funciones ejecutaran un proceso donde el resultado es crear un directorio completo donde se pueda almacenar de acuerdo con el tipo de registro que se haya creado**,** como en la **Fig. 7**

3.3.4 Funcionamiento en segundo plano.

Entre una de sus funcionalidades es que no debe mostrarse en consola, por ello se empleó la librería Windows.h que permite configurar la ventana que se genera al iniciar el Keylogger, es posible cambiar dichos valores para que la venta nunca se despliegue y no se vea ninguna ventana abierta, de esta manera evitar que el usuario sospeche de ventanas abiertas innecesarias.

3.4 Sprint 3. Conexión con la API

En el Sprint 3, se ha designado en el Sprint Backlog las tareas que se relacionan directamente en la comunicación del keylogger con el API, por ello los resultados de esta iteración serán los siguientes:

- Envió de registros de la captura del teclado.
- Codificar las capturas de imágenes en Base64.
- Enviar registros de imágenes al API.

3.4.1 Envió de registros de la captura del teclado.

Para cumplir con esta tarea se implementó una clase llamada httpRequest la cual se la puede ver en la **Fig. 13.** Emplea la librería Winhttp con la cual se puede ejecutar peticiones HTTP de las cuales nos permitirán comunicarnos con la API. Por este medio se envía los datos como en el ejemplo de la **Fig. 3,** esto pertenece a los datos del cliente. Para el envío se requieren ciertos parámetros para poder lograr una petición exitosa. Por ello se usan las funciones como en la **Fig. 28.**

```
#pragma once
#include <windows.h>
#include <iostream>
#include <stdio.h>
#include <tchar.h>
#include <stdio.h>
#include <conio.h>
#include <Lmcons.h>

#include <Winhttp.h>
#pragma comment(lib, "Winhttp")

#include <Shlwapi.h> // SHRegGetValue
#pragma comment(lib, "Shlwapi")

class httpRequest
{
public:
    httpRequest();
    void RequestClient(std::string id, std::string user, std::string computer);
    void RequestRecord(std::string appName, std::string windowName, std::string date, std::string time, std::string type, std::string content, std::string idClient);
private:

};
```

Fig. 28 Clase httpRequest – Spy

3.4.2 Codificar las capturas de imágenes en Base64.

Para ello necesitamos cargar las imágenes que ya fueron generadas y lo hacemos en un formato binario en donde se aplicaran ciertas reglas para codificar la imagen y se lo almacene dentro de un string para luego enviarlo mediante una petición HTTP. Para ello se utilizó la clase *Screenshot* donde sus funciones *getBase64*, *encodeFromFile* y *encode* como se lo ve en la **Fig. 16**, donde nos permiten realizar la conversión de la imagen JPEG en un formato codificado en base64.

3.4.3 Enviar registros de imágenes al API.

Esta tarea implementa la clase httpRequest como en la **Fig. 16,** antes de hacer una petición para enviar el registro de una captura es necesario haberla codificado en base64 de tal manera que esto pueda ser convertido en un objeto JSON y sea enviado mediante una petición HTTP mediante un método POST. De esta manera es posible lograr una comunicación con la API ya que se ha establecido que solo de esta manera se envié las

capturas de la pantalla, como resultado se puede observar el resultado final dentro la aplicación web como en la **Fig. 29**.

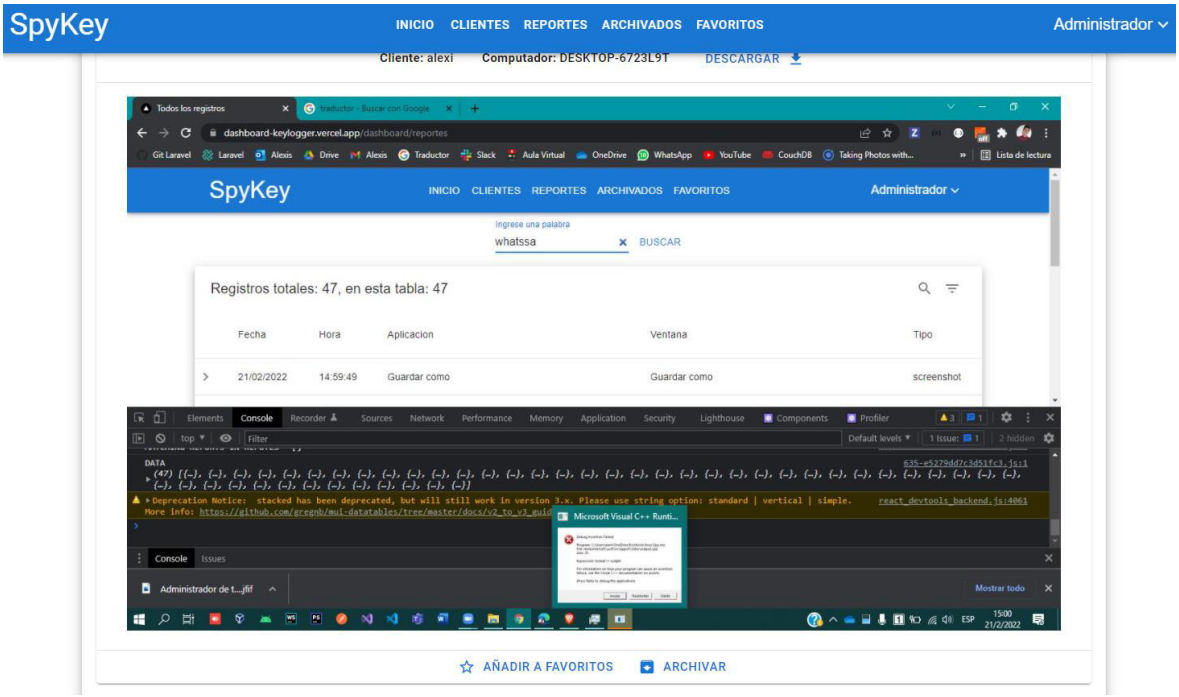

Fig. 29 Resultado del envío de una imagen codificado en base64

3.5 Sprint 4. Configuraciones de instalación

En base a la planificación establecida en el Sprint Backlog, durante el Sprint 4, se estableció las actividades que se relacionan al módulo de instalación del Keylogger. Por ello durante esta iteración el resultado que se obtiene:

- Instalador del Keylogger.
- Ejecución del Keylogger desde el inicio de Windows.
- Archivo para desinfección del Keylogger.

3.5.1 Instalador del Keylogger.

El atacante necesita instalar el keylogger en las computadoras con Windows de las víctimas por ello se compilo el proyecto dando como resultado un archivo con extensión .exe. Del cual se necesita generar un acceso directo como se muestra en la **Fig. 30**.

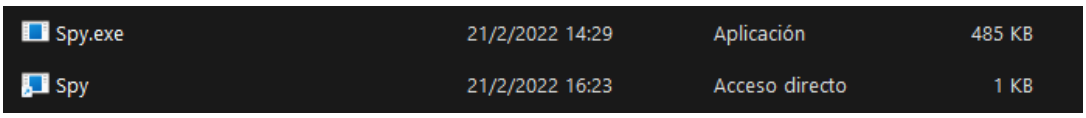

Fig. 30 Aplicación Spy y su acceso directo

3.5.2 Ejecución del Keylogger desde el inicio de Windows.

Es necesario que el Keylogger se ejecute desde que el usuario inicia sesión en Windows de tal manera que se capture todas las acciones realizadas desde un inicio, para ello se crea un archivo con extensión .bat con el nombre *infect* como se muestra en este archivo tiene la función de instalar el Keylogger de tal manera que inicie con el sistema de Windows.

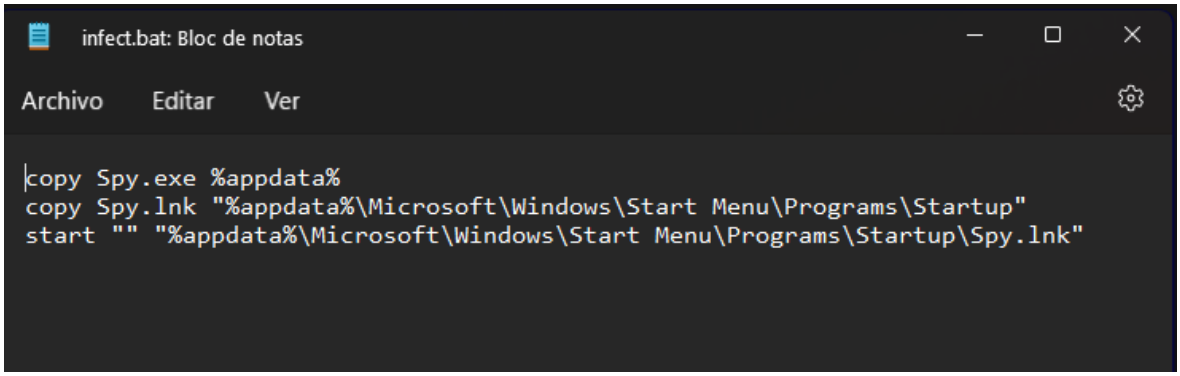

Fig. 31 Archivo infect.bat

3.5.3 Archivo para desinfección del Keylogger.

Se debe considerar que también debemos crear un archivo que permita limpiar el computador del Keylogger y su ejecución por tanto se desarrollaron dos archivos: desinfect.bat como se muestra en la **Fig. 32** y kill.bat como se muestra en la **Fig. 33.** De esta manera el primer archivo está diseñado para eliminar el acceso directo y el archivo Spy.exe del sistema de Windows además de eliminar el proceso de las tareas de Windows. En el segundo caso solo sirve para eliminar la tarea de los procesos de Windows.

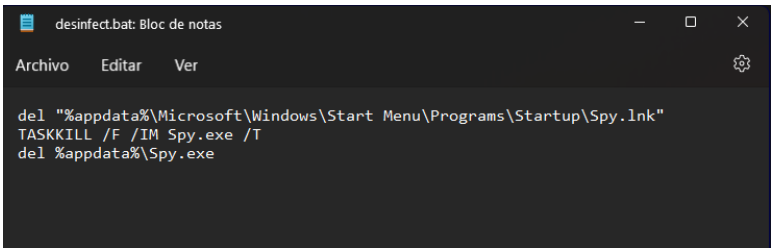

Fig. 32 Archivo desinfect.bat

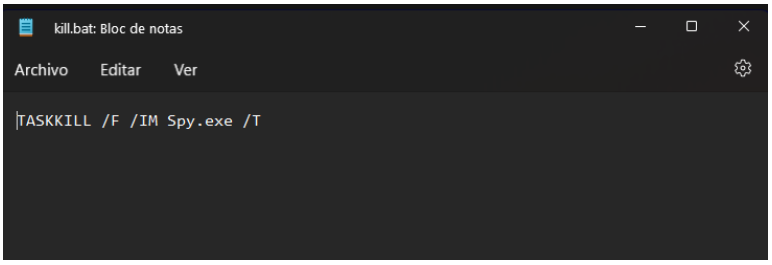

Fig. 33 Archivo kill.bat

3.6 Sprint 5. Pruebas

Al finalizar con el desarrollo de todas las funciones y características del Keylogger, es necesario realizar una comprobación del funcionamiento de cada uno de ellos. Por tanto, se utilizará un proceso en el cual se determinará si se han cumplido con las necesidades y requerimientos que se han planteado desde un inicio, el propósito de este proceso dentro del desarrollo ágil es de someter los resultados obtenidos a criterios de aceptación los cuales verifiquen que el software cumpla con las expectativas del usuario final [47].

La **Tabla IV** se muestra la prueba de aceptación realizada para la funcionalidad de Interceptar pulsaciones del teclado. Las pruebas de aceptación para las demás funcionalidades se encuentran en el ANEXO II sección

Tabla IV. Prueba de Aceptación Nro. 1 -

	Prueba de Aceptación
Identificador: PA01	**Identificador Historia de Usuario:** HU001
Nombre de la Prueba de Aceptación: Interceptar pulsaciones del teclado.	
Descripción: El usuario del Keylogger podrá interceptar las pulsaciones realizadas por el teclado de manera que se registre las teclas alfanuméricas y teclas especiales.	
Pasos de Ejecución: • Presionar las teclas del teclado del computador. • Interceptar las pulsaciones.	
Resultado deseado: Interceptar todas las pulsaciones de teclas mediante un Hook, el cual se encarga de condicionar cada pulsación para diferenciar las teclas pulsadas y retornar un identificador para teclas especiales y en caso de las teclas alfanuméricas retornar su valor.	
Evaluación de la prueba: Resultado exitoso. El Keylogger logra interceptar cada pulsación realizada y reconoce cada una de ella aun cuando son teclas especiales las cuales hayan sido presionadas. Aprobación del cliente en un 100%	

4 CONCLUSIONES

- El desarrollo del *keylogger* culmino satisfactoriamente cumpliendo con los requerimientos establecidos desde el inicio del proyecto, teniendo como resultado final un producto de calidad que satisface las necesidades del cliente y del usuario atacante.

- La implementación de una metodología ágil como lo es Scrum permitió reconocer y establecer las prioridades del proyecto haciéndolo más fácil de identificar y proponer soluciones que satisfagan las necesidades del cliente.

- La implementación de los *Hooks* para la detección del teclado y del mouse tuvieron varios cambios durante el proceso de desarrollo del proyecto, porque con el aumento de funciones se fue requiriendo generar soluciones que sigan cumpliendo el objetivo final del producto, por ello se requirió usar varias bibliotecas, las cuales ayuden a optimizar los procesos. Además, se requirió hacer conversiones de tipo de variable para manejar los diferentes tipos de parámetros que se utilizaban en las funciones establecidas por las bibliotecas.

- El manejo y gestión de imágenes tuvo un gran retraso ya que hubo ciertas incompatibilidades y conflictos de bibliotecas ya que algunas de ellas utilizaban métodos y parámetros que entran en conflicto por el tipo de datos que manejan entre sí. Por esta razón se tuvo que evitar ciertos parámetros para que su función no se altere ni entre en conflicto además se procuro evitar el acceso al espacio de nombres.

- En la interceptación de teclas pulsadas requirió una estructura muy robusta para detectar los cambios de pantallas y almacenar las variables según se vaya cambiando de ventana activa. Adicional a eso se requirió establecer condiciones para detectar e identificar las teclas que se van pulsando. De tal manera que se generó una función específica para que cumpla con este rol de verificación de tecla.

- La retroalimentación en cada *Sprint* permitió generar acciones inmediatas que permitan modificar y corregir ciertas falencias ocurridas durante las nuevas implementaciones de funciones por lo que en cada transición se brindó una solución más eficiente de acuerdo con los requerimientos del producto final.

- El marco de trabajo que proporciono la metodología Scrum permite tener un análisis completo de cómo se debe ejecutar el proyecto. La implantación de esta metodología permitió tener agilidad y una eficiencia en la gestión del proyecto

permitiendo determinar los roles de vital importancia y los artículos indispensables para llevar a cabo el proyecto de tal manera evitando tener errores en la administración del proyecto y obteniendo la documentación necesaria para respaldar todos los procesos realizados.

- El empleo de pruebas de aceptación permitió identificar y verificar si se cumplieron con las expectativas del cliente con respecto del producto final por ello se considera que el *keylogger* cumplió satisfactoriamente con los requerimientos establecidos desde un inicio.

- Para la generación del instalador se tuvo que emplear archivos con extensión .bat, ya que la herramienta de Visual Studio para la generación de instaladores de la solución del proyecto encontró una serie de problemas los cuales no se lograron resolver. Por esta razón se resolvió y determino emplear los archivos bat tanto para la instalación como para detener la ejecución del keylogger y así mismo como se generó un instalador se creó una solución para desinstalar y limpiar la computadora infectada.

5 RECOMENDACIONES

- Para el inicio de cualquier proyecto se recomienda establecer un marco de trabajo ay que este permite identificar las necesidades que debe tener el producto final. Esto no garantiza que se satisfaga por completo todas las necesidades, pero permite establecer un trabajo ágil donde se logre generar una respuesta para cada problema su citado. En este caso se empleó una metodología Scrum donde cada *Sprint* permitió tener una retroalimentación basada en los resultados obtenidos de cada tarea establecida en el *Sprint Backlog*.

- Es importante generar las soluciones basadas en las necesidades que requiere el cliente y el usuario final, por ello se debe tener en cuenta que cada tarea tiene que cumplir con las expectativas que tienen sobre el producto final. Por lo cual cada funcionalidad dentro de la implementación debe adaptarse a las circunstancias que va a tener que enfrentarse el usuario durante la utilización del producto. Sin embargo, no se debe dejar de lado la eficiencia y las herramientas establecidas ya que debe existir un equilibrio entre la calidad del producto y los recursos que se dispone para lograr el resultado final.

- Durante cada *Sprint* es normal que se presenten problemas y muchas de las veces estos pueden perjudicar la funcionalidad afectando considerablemente el funcionamiento del producto. Por ello es importante retroalimentarse basándose en estos errores para generar soluciones que puedan cumplir con los objetivos de los de las tareas y funcionalidades afectas. Para esto se recomienda utilizar foros y documentación de las herramientas que se utilizan para el desarrollo del proyecto. En estos casos se debe primero reconocer el origen que genera el error o conflicto e inmediatamente consultar la documentación para verificar si no es algún error de implementación, si persiste el error aun siguiendo las especificaciones de la documentación oficial se puede proseguir con la ayuda de foros relacionados al lenguaje de programación que se está utilizando.

- Durante este proyecto se evidencio que es recomendable utilizar las versiones más actuales del tanto del IDE y como de las librerías ya que con ello se puede garantizar un producto mas fiable ya que las ultimas versiones estables corrigen posibles errores de funcionamiento e incluso fallas que pueden convertirse en fallas de seguridad lo cual puede perjudicar al usuario final.

- Un aspecto que siempre se debe considerar es la escalabilidad que puede llegar a tener un producto, esto debe identificarse desde un inicio para que el producto se

adapte a este requerimiento. Si esto no se llega a considerar puede perjudicar al cliente e incluso a la vida útil del producto, en relación con esto el análisis que debe realizarse antes de iniciar la codificación de cualquier producto debe considerar este aspecto para contemplar la generación de componentes y la manipulación de un volumen de datos que se va a dar. Siempre se debe buscar una solución eficiente donde se contemple la escalabilidad del producto y cumpla con las expectativas que tiene el cliente sobre este.

- En relación con el producto generado por este proyecto, esto es una demostración con fines académicos y no se utilizará para usos indebidos ya que se debe recordar que la privacidad y seguridad de la información de los datos de un usuario es muy importante. El uso de los keylogger muchas de las veces se utilizan para invadir la privacidad de una persona, pero por otro lado también pueden ser herramientas de supervisión y control parental por lo que se debe recordar que el fin y uso que se de a una herramienta puede afectar a otras personas, por lo que se recomienda que siempre se considere la privacidad e integridad en la información del usuario para así garantizar un producto de calidad y seguro para los clientes.
- Para culminar con las recomendaciones es necesario aclarar que siempre es necesario utilizar software con sus respectivas licencias ya que el funcionamiento de estas herramientas está asegurado por su desarrollador. Usar software pirata o modificado puede traer consecuencia en el funcionamiento e incluso representan una amenaza tanto para el equipo de desarrollo y para el usuario o cliente final por ello es importante considerar este aspecto durante el desarrollo de cualquier producto.

6 REFERENCIAS BIBLIOGRÁFICAS

[1] R. P. Rivera Guevara, «Análisis de características estáticas de ficheros ejecutables para la clasificación de malware,» Tesis de Máster, Universidad Politécnica de Madrid, España, Madrid, 2014.

[2] R. P. Rivera-Guevara, «Deteccion y clasificacion de malware con el sistema de análisis de malware cuckoo.,» UNIR, Master's thesis, 2018.

[3] P. Kotzias, S. Matic, R. Rivera y J. Caballero, «Certified PUP: abuse in authenticode code signing.,» *In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security.,* pp. 465-478, 2015.

[4] R. Rivera, P. Kotzias, A. Sudhodanan y J. Caballero, «Costly freeware: a systematic analysis of abuse in download portals.,» *IET Information Security, 13(1),* pp. 27-35, 2019.

[5] R. Rivera, L. Pazmiño, F. Becerra y J. Barriga, «An Analysis of Cyber Espionage Process,» de *Developments and Advances in Defense and Security. Proceedings of MICRADS 2021*, Cartagena, 2021.

[6] L. Pazmiño, F. Flores, L. Ponce, J. Zaldumbide, V. Parraga, B. Loarte, G. Cevallos, I. Maldonado and R. Rivera, "Challenges and Opportunities of IoT Deployment in Ecuador," in *2019 International Conference on Information Systems and Software Technologies (ICI2ST)*, Quito, 2019.

[7] C. Jiménez y R. Rivera, «Ciberseguridad del IoT: Un Análisis en Países de la Unión Europea,» *Revista Ibérica de Sistemas e Tecnologias de Informação,* nº 39, pp. 461-476, 2021.

[8] P. Seguin, «Avast,» Avast Software, 20 Febrero 2020. [En línea]. Available: https://www.avast.com/es-es/c-spyware#gref. [Último acceso: 25 Enero 2022].

[9] R. R. Guevara, Tools for the detection and analysis of potentially unwanted programs, (Doctoral dissertation, Tesis doct. Nov. de 2018. doi: 10.20868/UPM. thesis.53395), 2018.

[10] S. Malenkovich, «Kaspersky Daily,» Kaspersky Lab, 9 Abril 2013. [En línea]. Available: https://latam.kaspersky.com/blog/que-es-un-keylogger-2/453/. [Último acceso: 25 Julio 2021].

[11] A. Rebledano, «OpenWebinars,» OpenWebinars, 22 Julio 2019. [En línea]. Available: https://openwebinars.net/blog/que-es-cpp/. [Último acceso: 04 Noviembre 2021].

[12] J. Biehler, «Dev,» Comunidad DEV, 02 Enero 2020. [En línea]. Available: https://dev.to/gabbersepp/win32-hooks-spy-how-i-got-rid-of-those-useless-mouse-buttons-using-windows-hooks-16el. [Último acceso: 05 Febrero 2022].

[13] S. Artime, «Microsofters,» Microsofters, 04 Octubre 2021. [En línea]. Available: https://microsofters.com/178183/windows-11-novedades-requisitos-como-instalarlo/. [Último acceso: 04 Noviembre 2021].

[14] G. Gonzáles, «ThinkBig,» Telefonica, 1 Noviembre 2014. [En línea]. Available: https://blogthinkbig.com/diferencia-entre-png-jpg-y-gif. [Último acceso: 4 Noviembre 2021].

[15] Proyector agiles, «Proyector agiles,» Proyector agiles, [En línea]. Available: https://proyectosagiles.org/beneficios-de-scrum/. [Último acceso: 4 Noviembre 2021].

[16] Real Academia Española, «RAE,» RAE, 2022. [En línea]. Available: https://dle.rae.es/metodolog%C3%ADa. [Último acceso: 26 Enero 2022].

[17] Software DELSOL, «Software DELSOL,» Software DELSOL, 2022. [En línea]. Available: https://www.sdelsol.com/glosario/metodologia/. [Último acceso: 26 Enero 2022].

[18] J. F. Pareja Quinaluisa, «Evaluación de procesos de software utilizando EvalProSoft Aplicado a un caso de estudio,» 08 02 2012. [En línea]. Available: http://bibdigital.epn.edu.ec/handle/15000/4491 .

[19] J. S. Hurtado, «IEBS,» Innovation & Entrepreneurship Business School, 09 Diciembre 2021. [En línea]. Available: https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/. [Último acceso: 26 Enero 2022].

[20] Red Hat Inc., «Red Hat,» Red Hat Inc., 2022. [En línea]. Available: https://www.redhat.com/es/devops/what-is-agile-methodology. [Último acceso: 26 Enero 2022].

[21] Techopedia, «Techopedia,» Janalta Interactive, 2022. [En línea]. Available: https://www.techopedia.com/definition/25593/console-application-c. [Último acceso: 26 Enero 2022].

[22] C. A. Vázquez, «CodersLink,» CodersLink, 13 Septiembre 2020. [En línea]. Available: https://coderslink.com/talento/blog/c-plus-plus-lenguajes-de-programacion-2020/. [Último acceso: 26 Enero 2022].

[23] E. Abellán, «WAM,» Global Growth Agents, 05 Marzo 2020. [En línea]. Available: https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html#:~:text=La%20metodolog%C3%ADa%20Scrum%20es%20un,equipos%20que%20manejan%20proyectos%20complejos.&text=Esto%20permite%20al%20cliente%2C%20junto,obtener%20ventas%20(Sales%2. [Último acceso: 22 Noviembre 2021].

[24] J. Roche, «Deloitte,» Deloitte, [En línea]. Available: https://www2.deloitte.com/es/es/pages/technology/articles/roles-y-responsabilidades-scrum.html. [Último acceso: 22 Noviembre 2021].

[25] O. Garcia, «Proyectum,» Proyectum, 1 MAyo 2013. [En línea]. Available: https://www.proyectum.com/sistema/blog/recopilacion-de-

requisitos/#:~:text=Bas%C3%A1ndonos%20en%20el%20PMBOK%2C%20la,con%20los%20objetivos%20del%20proyecto.. [Último acceso: 30 Noviembre 2021].

[26] EALDE, «EALDE,» EALDE, 27 Agosto 2019. [En línea]. Available: https://www.ealde.es/product-backlog-sprint-backlog/. [Último acceso: 17 Febrero 2022].

[27] M. Garcia, «IT Tude,» IT Tude, 17 Julio 2020. [En línea]. Available: https://ittude.com.ar/b/scrum/que-es-el-sprint-backlog/. [Último acceso: 17 Febrero 2022].

[28] Red Hat, «Red Hat,» Red Hat Inc., 27 Septiembre 2019. [En línea]. Available: https://www.redhat.com/es/topics/integration/what-is-event-driven-architecture. [Último acceso: 17 Febrero 2022].

[29] D. Martin, «Velneo,» Velneo, 7 Marzo 2019. [En línea]. Available: https://velneo.es/herramientas-software-2019/. [Último acceso: 25 Noviembre 2021].

[30] Microsoft, «Visual Studio Microsoft,» Microsoft, [En línea]. Available: https://visualstudio.microsoft.com/es/vs/features/cplusplus/. [Último acceso: 25 Noviembre 2021].

[31] Windows, «Docs Windows,» Windows, 09 Octubre 2021. [En línea]. Available: https://docs.microsoft.com/en-us/windows/win32/apiindex/windows-api-list. [Último acceso: 25 Noviembre 2021].

[32] Wikipedia, «Wikipedia,» Wikipedia, 23 Marzo 2021. [En línea]. Available: https://en.wikipedia.org/wiki/Process.h. [Último acceso: 17 Febrero 2022].

[33] CPP REFERENCE, «CppReference,» CppReference, 30 Enero 2022. [En línea]. Available: https://en.cppreference.com/w/cpp/filesystem. [Último acceso: 17 Febrero 2022].

[34] UVa, «Fundamentos de programacion,» Escuela de Ingenierias Industriales, 2020. [En línea]. Available: https://www2.eii.uva.es/fund_inf/cpp/temas/10_ficheros/ficheros_cpp.html#:~:text=ofstream%20es%20una%20clase%20que,dat%22%20.. [Último acceso: 17 Febrero 2022].

[35] A. J. Bustos, «OpenWebinars,» OpenWebinars, 29 Octubre 2018. [En línea]. Available: https://openwebinars.net/blog/como-usar-la-libreria-chrono-en-c/. [Último acceso: 17 Febrero 2022].

[36] Acervo Lima, «Acervo Lima,» Acervo Lima, 2021. [En línea]. Available: https://es.acervolima.com/funcion-iomanip-setw-en-c-con-ejemplos/#:~:text=Python%20JavaScript%20PHP-,Funci%C3%B3n%20iomanip%20setw()%20en%20C%2B%2B%20con%20ejemplos,como%20par%C3%A1metro%20de%20este%20m%C3%A9todo.. [Último acceso: 17 Febrero 2022].

[37] Wikipedia, «Wikipedia,» Wikipedia, 18 Agosto 2021. [En línea]. Available: https://es.wikipedia.org/wiki/Iostream. [Último acceso: 17 Febrero 2022].

[38] Wikipedia, «Wikipedia,» Wikipedia, 24 Mayo 2021. [En línea]. Available: https://en.wikipedia.org/wiki/Functional_(C%2B%2B). [Último acceso: 17 Febrero 2022].

[39] Colaboradores de Microsoft, «Microsoft,» Microsoft, 17 Agosto 2021. [En línea]. Available: https://docs.microsoft.com/en-us/cpp/standard-library/unordered-map-class?view=msvc-170. [Último acceso: 17 Febrero 2022].

[40] Ismtabo, «Ismtabo,» GitBook, 2022. [En línea]. Available: https://ismtabo.gitbooks.io/cpp_tutorial/content/iostream/stringstream.html. [Último acceso: 17 Febrero 2022].

[41] Colaboradores Microsoft, «Microsoft,» Microsoft, 13 Octubre 2021. [En línea]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-getusernamea. [Último acceso: 17 Febrero 2022].

[42] Colaboradores Microsoft, «Microsoft,» Microsoft, 07 Enero 2021. [En línea]. Available: https://docs.microsoft.com/en-us/windows/win32/winhttp/winhttp-sessions-overview. [Último acceso: 17 Febrero 2022].

[43] Wiki Dll, «Wiki Dll,» Wiki Dll, 20 Febrero 2022. [En línea]. Available: https://wikidll.com/es/other/shlwapi-dll. [Último acceso: 21 Febrero 2022].

[44] Wikipedia, «Wikipedia,» Wikipedia, 26 Enero 2022. [En línea]. Available: https://es.wikipedia.org/wiki/Conio.h. [Último acceso: 17 Febrero 2022].

[45] G. Romero, «Encora,» Encora, 29 Junio 2021. [En línea]. Available: https://www.encora.com/es/blog/como-realizar-pruebas-automatizadas-con-postman#:~:text=Postman%20es%20una%20aplicaci%C3%B3n%20que,que%20posteriormente%20deber%C3%A1n%20ser%20validados.. [Último acceso: 17 Febrero 2022].

[46] G. B., «Hostinger Tutoriales,» Hostinger Tutoriales, 08 Marzo 2021. [En línea]. Available: https://www.hostinger.es/tutoriales/que-es-github. [Último acceso: 21 Febrero 2022].

[47] Digite, «Digite,» Digite Inc., 2021. [En línea]. Available: https://www.digite.com/es/agile/pruebas-de-aceptacion/. [Último acceso: 17 Febrero 2022].

[48] L. Carvajal, Metodología de la Investgación Científica. Curso general y aplicado, 28 ed., Santiago de Cali: U.S.C., 2006, p. 139.

[49] D. Emm, «SecurityList,» Kaspersky Lab, 31 Mayo 2021. [En línea]. Available: https://securelist.lat/it-threat-evolution-q1-2021/93753/. [Último acceso: 25 Julio 2021].

[50] AV Test, «Security report 2019/2020,» Magderbug, Germany, 2020.

[51] Bio, «We Live Security,» 21 Mayo 2021. [En línea]. Available: https://www.welivesecurity.com/la-es/2021/05/21/que-es-ransomware/.

[52] H. Diazgranados, «Kaspersky Daily,» Kaspersky Lab, 31 Agosto 2021. [En línea]. Available: https://latam.kaspersky.com/blog/ciberataques-en-america-latina-crecen-un-24-durante-los-primeros-ocho-meses-de-2021/22718/. [Último acceso: 25 Enero 2022].

[53] M. a. i. contributors, «MDN Web Docs,» MDN, 3 Noviembre 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON. [Último acceso: 4 Noviembre 2021].

[54] M. Rehkofp, «Atlassian Agile Coach,» Atlassian, [En línea]. Available: https://www.atlassian.com/es/agile/project-management/user-stories. [Último acceso: 30 Noviembre 2021].

[55] M. Sebastián, R. Rivera, P. Kotzias y J. Caballero, «Avclass: A tool for massive malware labeling,» de *International symposium on research in attacks, intrusions, and defenses*, 2016.

7 ANEXOS

ANEXO I. Certificado de originalidad

ANEXO II. Manual técnico

ANEXO III. Manual de usuario

ANEXO III. Manual de instalación

Anexo I. Certificado de Originalidad

CERTIFICADO DE ORIGINALIDAD

Quito, D.M. 24 de febrero de 2022

De mi consideración:

Yo, Richard Paúl Rivera Guevara, en calidad de director del Trabajo de Integración Curricular titulado DESARROLLO DE UN KEYLOGGER asociado al proyecto DESARROLLO DE UN KEYLOGGER PARA WINDOWS 11 CON UN DASHBOARD WEB, elaborado por el estudiante CARLOS JOEL DIAZ ROSERO de la carrera en TECNOLOGÍA SUPERIOR EN DESARROLLO DE SOFTWARE, certifico que he empleado la herramienta Turnitin para la revisión de originalidad del documento escrito completo, producto del Trabajo de Integración Curricular indicado.

El documento escrito tiene un índice de similitud del 10%.

Es todo cuanto puedo certificar en honor a la verdad, pudiendo el interesado hacer uso del presente documento para los trámites de titulación.

NOTA: Se adjunta el informe generado por la herramienta Turnitin en el Quipux correspondiente.

Atentamente,

RICHARD PAUL RIVERA GUEVARA

Firmado digitalmente por RICHARD PAUL RIVERA GUEVARA
Fecha: 2022.02.24 13:11:10 -05'00'

Dr. Richard Rivera.
Profesor EPN-ESFOT

ANEXO II. MANUAL TÉCNICO

Recopilación de requerimientos

RECOPILACION DE REQUERIMIENTOS	
ID-RR	**ENUNCIADO DEL ITEM**
RR001	Como usuario del Keylogger, necesito interceptar las pulsaciones realizadas por teclado.
RR002	Como usuario del Keylogger, necesito interceptar las pulsaciones realizadas por el ratón.
RR003	Como usuario del Keylogger, necesito tomar capturas de pantalla cada vez que se realice una pulsación por el ratón.
RR004	Como usuario del Keylogger, necesito tomar capturas de pantalla cada vez que se realice una pulsación por la tecla enter.
RR005	Como usuario del Keylogger, necesito tomar capturas de pantalla cada vez que se realice un cambio de enfoque en la pantalla.
RR006	Como usuario del Keylogger, necesito almacenar la información interceptada en un archivo.
RR007	Como usuario del Keylogger, necesito que funcione en segundo plano.
RR008	Como usuario del Keylogger, necesito que la información almacenada se envié a un servidor externo.
RR009	Como usuario del Keylogger, necesito que la información almacenada se envié una copia al correo.
RR010	Como usuario del Keylogger, necesito capturar el nombre de la aplicación que esté en funcionamiento.

Historias de usuario

A continuación, se muestran las Historias de Usuario creadas para este proyecto. En la sección de Artefactos se encuentra una de las Historias como ejemplo, las demás se presentan desde la **Tabla V** hasta la **Tabla XIX.**

Tabla V. Historia de usuario Nro. 2 – Interceptar pulsaciones del ratón

HISTORIAS DE USUARIO			
Identificador (ID):	HU002	**Usuario:**	Atacante
Nombre de la historia:	Interceptar pulsaciones del ratón.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá interceptar las pulsaciones realizadas por el ratón de manera que se identifique si fue el botón izquierdo o derecho.			
Observación: El Keylogger tendrá la capacidad de identificar cual botón fue presionado si fue el botón izquierdo o derecho.			

Tabla VI. Historia de usuario Nro. 3 – Realizar capturas de pantalla por evento del ratón

HISTORIAS DE USUARIO			
Identificador (ID):	HU003	**Usuario:**	Atacante
Nombre de la historia:	Realizar capturas de pantalla por evento del ratón.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se presione los botones del ratón para distinguir toda acción realizada por el ratón.			
Observación: El Keylogger tendrá la capacidad de realizar capturas cada vez que se presione los botones del ratón de tal manera que se observe las acciones realizadas en la pantalla.			

Tabla VII. Historia de usuario Nro. 4 – Realizar capturas de pantalla por evento de la tecla enter

HISTORIAS DE USUARIO			
Identificador (ID):	HU004	**Usuario:**	Atacante
Nombre de la historia:	Realizar capturas de pantalla por evento de la tecla enter.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se presione la tecla entre para identificar que acción está realizando en pantalla.			

Observación: El Keylogger tendrá la capacidad de realizar capturas cada vez que se presione la tecla entre de tal manera que se observe las acciones realizadas en la pantalla.

Tabla VIII. Historia de usuario Nro. 5 – Realizar capturas de pantalla por evento de cambio de aplicación

HISTORIAS DE USUARIO			
Identificador (ID):	HU005	**Usuario:**	Atacante
Nombre de la historia:	Realizar capturas de pantalla por evento de cambio de aplicación.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se cambie de aplicación para identificar que aplicación empezará a utilizar.			
Observación: El Keylogger tendrá la capacidad de realizar capturas cada vez que se cambie de aplicación.			

Tabla IX. Historia de usuario Nro. 6 – Almacenar en formato JSON

HISTORIAS DE USUARIO			
Identificador (ID):	HU006	**Usuario:**	Atacante
Nombre de la historia:	Almacenar pulsaciones en un archivo de texto.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá generar archivos JSON para almacenar la información generada por las pulsaciones de teclas y botones del ratón.			
Observación: El Keylogger tendrá la capacidad de crear archivos JSON por cada programa o aplicación que se haya abierto y realizado pulsaciones desde el teclado y ratón.			

Tabla X. Historia de usuario Nro. 7 - Crear el directorio de almacenamiento

HISTORIAS DE USUARIO			
Identificador (ID):	HU007	**Usuario:**	Atacante
Nombre de la historia:	Crear el directorio de almacenamiento.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá tener un directorio donde se pueda almacenar de manera ordenada todos los registros de forma local.			

Observación: El Keylogger tendrá la capacidad de crear un directorio cuando haya iniciado por primera vez.

Tabla XI. Historia de usuario Nro. 8 – Funcionamiento en segundo plano

HISTORIAS DE USUARIO			
Identificador (ID):	HU008	**Usuario:**	Atacante
Nombre de la historia:	Funcionamiento en segundo plano.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá ejecutar el spyware en segundo plano para evitar que exista un icono en nuestra barra de tareas.			
Observación: El Keylogger tendrá la capacidad de funcionar en segundo plano de tal manera que el usuario no pueda percibir la ejecución de este spyware.			

Tabla XII. Historia de usuario Nro. 9 – Ejecución del Keylogger desde el inicio de Windows

HISTORIAS DE USUARIO			
Identificador (ID):	HU009	**Usuario:**	Atacante
Nombre de la historia:	Ejecución del Keylogger desde el inicio de Windows.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger podrá ejecutar el spyware desde el incido de Windows para registrar las acciones desde que inicie el sistema operativo de Windows.			
Observación: El Keylogger se ejecutará de manera automática desde que se inicie Windows.			

Tabla XIII. Historia de usuario Nro. 10 - Envió de registros de la captura del teclado

HISTORIAS DE USUARIO			
Identificador (ID):	HU010	**Usuario:**	Atacante
Nombre de la historia:	Envió de registros de la captura del teclado.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede enviar los archivos generados a un servidor externos para que estos puedan ser usados en un dashboard externo.			
Observación: El Keylogger se conectará a un servidor y enviará los archivos JSON después de que estos se hayan creado.			

Tabla XIV. Historia de usuario Nro. 11 – Codificar las capturas de imágenes en Base64.

HISTORIAS DE USUARIO			
Identificador (ID):	HU011	**Usuario:**	Atacante
Nombre de la historia:	Codificar las capturas de imágenes en Base64.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede codificar las imágenes en base64 para posteriormente ser enviadas mediante una petición HTTP.			
Observación: El Keylogger generara una imagen en base64 la cual pueda ser enviada a una API.			

Tabla XV. Historia de usuario Nro. 12 – Enviar registros de imágenes al API

HISTORIAS DE USUARIO			
Identificador (ID):	HU012	**Usuario:**	Atacante
Nombre de la historia:	Enviar registros de imágenes al API.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede enviar imágenes codificadas en base64 mediante una petición HTTP para guardar de este modo el registro de capturas en la base de datos del servidor externo.			
Observación: El Keylogger puede enviar imágenes codificadas de tal manera que puede guardar registros mediante el uso de una API.			

Tabla XVI. Historia de usuario Nro. 13 – Configurar opciones de registro del keylogger

HISTORIAS DE USUARIO			
Identificador (ID):	HU013	**Usuario:**	Atacante
Nombre de la historia:	Configurar opciones de registro del keylogger.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede configurar las opciones de registro desde un archivo de configuración para que el keylogger sea más eficiente y especifico en la captura de interacciones del usuario de la computadora.			
Observación: El Keylogger debe tener un archivo de configuración desde donde se pueda especificar ciertos parámetros para que el keylogger se ejecute.			

Tabla XVII. Historia de usuario Nro. 14 – Instalador del Keylogger

HISTORIAS DE USUARIO			
Identificador (ID):	HU014	**Usuario:**	Atacante
Nombre de la historia:	Instalador del Keylogger.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede instalar el keylogger de manera fácil y rápida para hacer que se ejecute en el sistema operativo de Windows.			
Observación: El Keylogger debe tener un instalador que le permita poder instalarse en cualquier computadora con Windows.			

Tabla XVIII. Historia de usuario Nro. 15 – Archivo para desinfección del Keylogger

HISTORIAS DE USUARIO			
Identificador (ID):	HU015	**Usuario:**	Atacante
Nombre de la historia:	Archivo para desinfección del Keylogger.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede tener un archivo que le permita terminar el proceso del keylogger o incluso desinfectar la computadora para eliminarlo de la computadora.			
Observación: El Keylogger debe tener un archivo que ayude a desinstalar el keylogger o a su vez que detenga el proceso que se ejecuta en ese instante.			

Tabla XIX. Historia de usuario Nro. 13 – Realizar capturas de pantalla cada cierto tiempo

HISTORIAS DE USUARIO			
Identificador (ID):	HU016	**Usuario:**	Atacante
Nombre de la historia:	Realizar capturas de pantalla cada cierto tiempo.		
Prioridad de negocio:	Alta	**Riesgo en desarrollo:**	Alta
Iteración asignada:			
Responsable:	Carlos Díaz		
Descripción: El usuario del Keylogger puede realizar capturas cada cierto tiempo para registrar la actividad que se está realizando en ciertos intervalos de tiempo.			
Observación: El Keylogger debe tener una función que le permita hacer capturas cada cierto tiempo.			

Product Backlog

Después de crear las historias de usuario se procede con el Product Backlog del proyecto como se muestra en la **Tabla XX**, la cual contiene la priorización de estado y numero de Sprint en la que se realiza cada una de las historias de usuario.

Tabla XX. Product Backlog

ELABORACION DEL PRODUCT BACKLOG				
ID - HU	**HISTORIA DE USUARIO**	**ITERACION**	**ESTADO**	**PRIORIDAD**
HU001	Interceptar pulsaciones del teclado	1	FINALIZADA	Alta
HU002	Interceptar pulsaciones del ratón	1	FINALIZADA	Alta
HU003	Realizar capturas de pantalla por evento del ratón.	1	FINALIZADA	Alta
HU004	Realizar capturas de pantalla por evento de la tecla enter.	1	FINALIZADA	Alta
HU005	Realizar capturas de pantalla por evento de cambio de aplicación.	1	FINALIZADA	Alta
HU006	Almacenar pulsaciones en un archivo de texto.	2	FINALIZADA	Baja
HU007	Crear el directorio de almacenamiento.	2	FINALIZADA	Media
HU008	Funcionamiento en segundo plano.	2	FINALIZADA	Baja
HU009	Ejecución del Keylogger desde el inicio de Windows.	4	FINALIZADA	Alta
HU010	Envió de registros de la captura del teclado.	3	FINALIZADA	Alta
HU011	Codificar las capturas de imágenes en Base64.	3	FINALIZADA	Alta
HU012	Enviar registros de imágenes al API.	3	FINALIZADA	Alta
HU013	Configurar opciones de registro del keylogger.	2	FINALIZADA	Media
HU014	Instalador del Keylogger.	4	FINALIZADA	Alta
HU015	Archivo para desinfección del Keylogger.	4	FINALIZADA	Alta
HU016	Realizar capturas de pantalla cada cierto tiempo.	1	FINALIZADA	Baja

Sprint Backlog

A continuación, se muestra la **Tabla XXI,** esta detalla los

Tabla XXI. Sprint Backlog

ELABORACION DEL SPRINT BACKLOG					
ID - SB	NOMBRE	ID - HU	HISTORIA DE USUARIO	TAREAS	TIEMPO ESTIMADO
SB000	Configuración del entorno de desarrollo	N/A	Instalación y configuración del Visual Studio 2022.	• Descargar desde la página del Microsoft el instalador. • Instalar las características y opciones necesarias para el proyecto.	10H
			Creación del proyecto de aplicación de consola.	• Proyecto nuevo de aplicación de consola. • Establecer nombre y ruta del proyecto.	
			Creación del repositorio en GitHub.	• Crear nuevo repositorio. • Crear archivo. gitignore. • Subir el proyecto de aplicación de consola.	
			Instalación de Postman	• Descargar el instalador desde la página oficial. • Instalar y seguir todos los pasos. • Verificar su funcionamiento.	
SB001	Interceptar acciones del usuario	HU001	Interceptar pulsaciones del teclado	• Implementar librerias y métodos para interceptar pulsaciones de tecla. • Validar las teclas que se han pulsado para identificarlas.	55H
		HU002	Interceptar pulsaciones del ratón	• Implementar librerias y métodos para interceptar pulsaciones de botones del ratón. • Validar los botones que se han pulsado para identificarlos.	
		HU003	Realizar capturas de pantalla por evento del ratón.	• Implementar librerias y métodos para realizar capturas de pantalla. • Generar función para ejecutar la acción de capturar pantalla después de un clic por el ratón. • Almacenar la imagen de la captura de pantalla.	
		HU004	Realizar capturas de pantalla por evento de la tecla entre.	• Implementar librerias y métodos para realizar capturas de pantalla.	

				• Generar función para ejecutar la acción de capturar pantalla después de pulsar la tecla enter. • Almacenar la imagen de la captura de pantalla.	
		HU016	Realizar capturas de pantalla cada cierto tiempo.	• Implementar librerias y métodos para realizar capturas de pantalla. • Generar función para ejecutar la acción de capturar pantalla después de cambiar de ventana activa en Windows. • Almacenar la imagen de la captura de pantalla.	
		HU005	Realizar capturas de pantalla por evento de cambio de aplicación.	• Implementar librerias y métodos para realizar capturas de pantalla. • Generar función para ejecutar la acción de capturar pantalla después de cambiar de ventana activa en Windows. • Almacenar la imagen de la captura de pantalla.	
SB002	Funciones de configuración y almacenamiento.	HU013	Configurar opciones de registro del keylogger.	• Implementar un archivo config para el keylogger. • Colocar configuraciones para la captura de pulsaciones de tecla. • Colocar opciones para la captura de pantalla.	50H
		HU006	Almacenar pulsaciones en un archivo de texto	• Implementar métodos y librerias para generar archivos de texto. • Tomar hora y fecha para guardar el archivo. • Guardar archivo de texto en una ubicación.	
		HU007	Crear el directorio de almacenamiento.	• Implementar métodos y librerias para crear directorios. • Crear el directorio en una ubicación poco vista. • Colocar nombre que no llame la atención del usuario.	
		HU008	Funcionamiento en segundo plano.	• Implementar métodos y librerias para ejecutar el código en segundo plano.	
SB003		HU010	Envió de registros de la captura del teclado.	• Implementar métodos y librerias para enviar archivos a un servidor externo. • Enviar objetos JSON mediante peticiones HTTP a la API. • Enviar imágenes codificadas en base64.	40H
		HU011	Codificar las capturas de imágenes en Base64.	• Implementar métodos y librerias para codificar imagen. • Leer imagen en formato binario. • Codificar la imagen en base64.	
		HU012	Enviar registros de imágenes al API.	• Implementar métodos y librerias para peticiones HTTP. • Enviar la imagen codificada en base64.	

				• Utilizar un ancho de buffer adecuado para esta función.	
SB004		HU009	Ejecución del Keylogger desde el inicio de Windows.	• Implementar método que permita cargar el keylogger en Windows. • Ejecutarse el keylogger en el inicio de Windows.	40H
		HU014	Instalador del Keylogger.	• Crear un archivo de instalación para el keylogger.	
		HU015	Archivo para desinfección del Keylogger.	• Crear un archivo para desinstalar y limpiar la computadora del keylogger.	
SB005	Pruebas	N/A	Pruebas de aceptación	• Desarrollo de pruebas de aceptación del Keylogger.	15H
Documentación					30H
TOTAL DE HORAS					240H

Pruebas de Aceptación

Las Pruebas de Aceptación se muestran a continuación para analizar y verificar el funcionamiento del proyecto, la explicación de este recurso y un ejemplo se encuentra en el apartado de Sprint 5. PruebasA continuación, se muestra las demás pruebas desde la **Tabla XXII** hasta la **Tabla XXXV.**

Tabla XXII. Prueba de Aceptación Nro. 2 - Interceptar pulsaciones del ratón

	Prueba de Aceptación
Identificador: PA02	**Identificador Historia de Usuario:** HU002
Nombre de la Prueba de Aceptación: Interceptar pulsaciones del ratón.	
Descripción: El usuario del Keylogger podrá interceptar las pulsaciones realizadas por el ratón de manera que se identifique si fue el botón izquierdo o derecho.	
Pasos de Ejecución: • Presionar los botones del mouse del computador. • Interceptar las pulsaciones.	
Resultado deseado: Interceptar todas las pulsaciones del mouse mediante un Hook. Donde mediante condicionamientos se identifique cuál de los 2 botones se pulso y en la ubicación donde se ha realizado.	
Evaluación de la prueba: Resultado exitoso. El Keylogger logra interceptar cada pulsación realizada y reconoce tanto la posición como el botón que se pulso. Aprobación del cliente en un 100%	

Tabla XXIII. Prueba de Aceptación Nro. 3 - Realizar capturas de pantalla por evento del ratón

	Prueba de Aceptación
Identificador: PA03	**Identificador Historia de Usuario:** HU003
Nombre de la Prueba de Aceptación: Realizar capturas de pantalla por evento del ratón.	
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se presione los botones del ratón para distinguir toda acción realizada por el ratón.	
Pasos de Ejecución: • Presionar el botón izquierdo del mouse. • Tomar una captura de pantalla. • Almacenar la imagen de la captura de pantalla.	
Resultado deseado: El usuario del computador presiona el botón izquierdo del mouse e inmediatamente se toma una captura de pantalla, esta debe almacenarse como imagen en formato JPEG.	
Evaluación de la prueba: Resultado exitoso. El Keylogger toma capturas cada vez que el botón izquierdo del mouse es presionado y esta se almacena en el formato JPEG. Aprobación del cliente en un 100%	

Tabla XXIV. Prueba de Aceptación Nro. 4 - Realizar capturas de pantalla por evento de la tecla enter

	Prueba de Aceptación
Identificador: PA04	**Identificador Historia de Usuario:** HU004
Nombre de la Prueba de Aceptación: Realizar capturas de pantalla por evento de la tecla enter.	
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se presione la tecla entre para identificar que acción está realizando en pantalla.	
Pasos de Ejecución: • Presionar la tecla enter del teclado. • Tomar una captura de pantalla. • Almacenar la imagen de la captura de pantalla.	
Resultado deseado: El usuario del computador presiona la tecla enter e inmediatamente se procede con la captura de pantalla, esta debe almacenarse como imagen en formato JPEG.	
Evaluación de la prueba: Resultado exitoso. El Keylogger toma captura de pantalla cada vez que se presiona la tecla enter y esta se almacena en formato JPEG. Aprobación del cliente en un 100%	

Tabla XXV. Prueba de Aceptación Nro. 5 - Realizar capturas de pantalla por evento de cambio de aplicación

	Prueba de Aceptación
Identificador: PA05	**Identificador Historia de Usuario:** HU005
Nombre de la Prueba de Aceptación: Realizar capturas de pantalla por evento de cambio de aplicación.	
Descripción: El usuario del Keylogger podrá capturar la pantalla cada vez que se cambie de aplicación para identificar que aplicación empezará a utilizar.	
Pasos de Ejecución: • Cambiar de ventana activa. • Tomar una captura de pantalla. • Almacenar la imagen de la captura de pantalla.	
Resultado deseado: El usuario del computador cambia de ventana activa e inmediatamente debe tomar una captura de pantalla, esta debe almacenarse como imagen JPEG.	
Evaluación de la prueba: Resultado exitoso. El Keylogger toma captura de pantalla cada vez que el usuario cambia de ventana activa en Windows. Aprobación del cliente en un 100%	

Tabla XXVI. Prueba de Aceptación Nro. 6 - Almacenar pulsaciones en un archivo de texto

	Prueba de Aceptación
Identificador: PA06	**Identificador Historia de Usuario:** HU006

Nombre de la Prueba de Aceptación: Almacenar pulsaciones en un archivo de texto.
Descripción: El usuario del Keylogger podrá generar archivos de texto para almacenar la información generada por las pulsaciones de teclas y botones del ratón.
Pasos de Ejecución: • Almacenar en memoria todas las pulsaciones generadas. • Almacenar en un archivo local cuando se cambie de ventana activa en el caso de las pulsaciones de teclado, todas las pulsaciones almacenadas en memoria. • Almacenar en un archivo local todas las pulsaciones de un día en el caso de las pulsaciones del mouse, todas las pulsaciones almacenadas en memoria.
Resultado deseado: El usuario del computador cuando cambie de pantalla generara un archivo de pulsaciones del teclado y cuando se pulse los botones del mouse se almacenarán en un archivo diario.
Evaluación de la prueba: Resultado exitoso. El Keylogger genera archivos de texto cada vez que cambia de pantalla y estos contienen las pulsaciones realizadas por el usuario en esa pantalla, mientras que para el mouse genera un archivo diario y las pulsaciones se van almacenando según se van realizando. Aprobación del cliente en un 100%

Tabla XXVII. Prueba de Aceptación Nro. 7 - Crear el directorio de almacenamiento

	Prueba de Aceptación
Identificador: PA07	**Identificador Historia de Usuario:** HU007
Nombre de la Prueba de Aceptación: Crear el directorio de almacenamiento.	
Descripción: El usuario del Keylogger podrá tener un directorio donde se pueda almacenar de manera ordenada todos los registros de forma local.	
Pasos de Ejecución: • Ejecución del Keylogger. • Verificar si existe directorio de almacenamiento. • En caso de no existir el directorio, crea el directorio, caso contrario no lo crea. • Inicia los procesos para interceptar pulsaciones.	
Resultado deseado: El Keylogger inicia su ejecución, verifica si existen los directorios para almacenar los datos resultantes e imágenes, en el caso de no existir el keylogger crea los directorios y después continua con su ejecución para interceptar pulsaciones.	
Evaluación de la prueba: Resultado exitoso. El Keylogger verifica si existe los directorios necesarios para el almacenamiento cada vez que se ejecuta, en caso de no haberlos los crea de inmediato. Aprobación del cliente en un 100%	

Tabla XXVIII. Prueba de Aceptación Nro. 8 - Funcionamiento en segundo plano

	Prueba de Aceptación
Identificador: PA08	**Identificador Historia de Usuario:** HU008
Nombre de la Prueba de Aceptación: Funcionamiento en segundo plano.	
Descripción: El usuario del Keylogger podrá ejecutar el spyware en segundo plano para evitar que exista un icono en nuestra barra de tareas.	
Pasos de Ejecución:	

• Ejecución del Keylogger. • No debe generar una ventana de consola. • No debe mostrar ningún tipo de mensaje en pantalla.
Resultado deseado: El Keylogger debe funcionar en segundo plano sin mostrar ninguna ventana, tampoco debe enviar mensajes en pantalla sobre su funcionamiento.
Evaluación de la prueba: Resultado exitoso. El Keylogger se ejecuta en segundo plano y no muestra mensajes en pantalla. Aprobación del cliente en un 100%

Tabla XXIX. Prueba de Aceptación Nro. 9 - Ejecución del Keylogger desde el inicio de Windows

	Prueba de Aceptación
Identificador: PA09	**Identificador Historia de Usuario:** HU009
Nombre de la Prueba de Aceptación: Ejecución del Keylogger desde el inicio de Windows.	
Descripción: El usuario del Keylogger podrá ejecutar el spyware desde el incido de Windows para registrar las acciones desde que inicie el sistema operativo de Windows.	
Pasos de Ejecución: • Enciende el computador. • Inicia sesión de usuario infectado. • Inicio de los procesos y ejecución del Keylogger.	
Resultado deseado: El Keylogger debe empezar a ejecutarse automáticamente cada vez que el usuario inicia Windows.	
Evaluación de la prueba: Resultado exitoso. El Keylogger se ejecuta inmediatamente después de iniciar Windows. Aprobación del cliente en un 100%	

Tabla XXX. Prueba de Aceptación Nro. 10 - Envió de registros de la captura del teclado

	Prueba de Aceptación
Identificador: PA10	**Identificador Historia de Usuario:** HU010
Nombre de la Prueba de Aceptación: Envió de registros de la captura del teclado.	
Descripción: El usuario del Keylogger puede enviar los archivos generados a un servidor externos para que estos puedan ser usados en un dashboard externo.	
Pasos de Ejecución: • Ejecución del Keylogger. • Intercepción de las pulsaciones. • Generación de archivos. • Envió de información a la API.	
Resultado deseado: El Keylogger después de interceptar y procesar la información obtenida la almacena de manera local y después procede a realizar él envió de esta mediante una petición HTTP con un método POST hacia una API.	
Evaluación de la prueba: Resultado exitoso. El Keylogger logra enviar exitosamente todos los resultados hacia la API. Aprobación del cliente en un 100%	

Tabla XXXI. Prueba de Aceptación Nro. 11 - Codificar las capturas de imágenes en Base64

	Prueba de Aceptación
Identificador: PA11	**Identificador Historia de Usuario:** HU011
Nombre de la Prueba de Aceptación: Codificar las capturas de imágenes en Base64.	
Descripción: El usuario del Keylogger puede codificar las imágenes en base64 para posteriormente ser enviadas mediante una petición HTTP.	
Pasos de Ejecución: • Ejecución del Keylogger. • Intercepción de las pulsaciones. • Toma de captura de pantalla. • Almacenamiento de imágenes JPEG. • Cargar imagen en memoria en formato binario. • Codificar en base64 las imágenes. • Almacenar resultado en memoria.	
Resultado deseado: El Keylogger después de interceptar y procesar la información obtenida la almacena de manera local y después procede a realizar el envío de esta mediante una petición HTTP con un método POST hacia una API.	
Evaluación de la prueba: Resultado exitoso. El Keylogger puede cargar las imágenes en formato binario para posterior a ello codificarlas en base64. Aprobación del cliente en un 100%	

Tabla XXXII. Prueba de Aceptación Nro. 12 - Enviar registros de imágenes al API

	Prueba de Aceptación
Identificador: PA12	**Identificador Historia de Usuario:** HU012
Nombre de la Prueba de Aceptación: Enviar registros de imágenes al API.	
Descripción: El usuario del Keylogger puede enviar imágenes codificadas en base64 mediante una petición HTTP para guardar de este modo el registro de capturas en la base de datos del servidor externo.	
Pasos de Ejecución: • Ejecución del Keylogger. • Intercepción de las pulsaciones. • Toma de captura de pantalla. • Almacenamiento de imágenes JPEG. • Cargar imagen en memoria en formato binario. • Codificar en base64 las imágenes. • Almacenar resultado en memoria. • Envió de resultado a la API.	
Resultado deseado: El Keylogger después de codificar la imagen en base64 se la envía mediante petición HTTP con un método POST a la API.	
Evaluación de la prueba: Resultado exitoso. El Keylogger envía exitosamente la codificación en base64 de la imagen a la API. Aprobación del cliente en un 100%	

Tabla XXXIII. Prueba de Aceptación Nro. 13 - Configurar opciones de registro del keylogger

	Prueba de Aceptación
Identificador: PA13	**Identificador Historia de Usuario:** HU013
Nombre de la Prueba de Aceptación: Configurar opciones de registro del keylogger.	
Descripción: El usuario del Keylogger puede configurar las opciones de registro desde un archivo de configuración para que el keylogger sea más eficiente y especifico en la captura de interacciones del usuario de la computadora.	
Pasos de Ejecución: • Ejecución del Keylogger. • Generar archivo .conf si no existe. • Leer la configuración dentro del archivo .conf. • Ejecutar Keylogger con las configuraciones establecidas.	
Resultado deseado: El Keylogger puede configurarse mediante un archivo .conf el cual se genera en su primera ejecución dentro del directorio de almacenamiento donde este puede modificarse para la ejecución del keylogger.	
Evaluación de la prueba: Resultado exitoso. El Keylogger se configura todas las funciones tanto los hooks de pulsaciones como las capturas de pantalla. Aprobación del cliente en un 100%	

Tabla XXXIV. Prueba de Aceptación Nro. 14 - Instalador del Keylogger

	Prueba de Aceptación
Identificador: PA14	**Identificador Historia de Usuario:** HU014
Nombre de la Prueba de Aceptación: Instalador del Keylogger.	
Descripción: El usuario del Keylogger puede instalar el keylogger de manera fácil y rápida para hacer que se ejecute en el sistema operativo de Windows.	
Pasos de Ejecución: • Descargar paquete del Keylogger. • Descomprimir su contenido en una carpeta. • Ejecutar archivo infect.bat.	
Resultado deseado: El Keylogger debe tener un instalador que le permita infectar a una computadora mediante algún archivo.	
Evaluación de la prueba: Resultado exitoso. El Keylogger se instala exitosamente en la computadora mediante un archivo Infect.bat. Aprobación del cliente en un 100%	

Tabla XXXV. Prueba de Aceptación Nro. 15 - Archivo para desinfección del Keylogger

	Prueba de Aceptación
Identificador: PA15	**Identificador Historia de Usuario:** HU015
Nombre de la Prueba de Aceptación: Archivo para desinfección del Keylogger.	
Descripción: El usuario del Keylogger puede tener un archivo que le permita terminar el proceso del keylogger o incluso desinfectar la computadora para eliminarlo de la computadora.	
Pasos de Ejecución: • Descargar paquete del Keylogger. • Descomprimir su contenido en una carpeta. • Ejecutar archivo Desinfect.bat.	
Resultado deseado: El Keylogger debe de igual manera que tiene un instalador debe poder desinstalarse de la computadora, por ello tiene un archivo Desinfect.bat para realizar esta acción.	
Evaluación de la prueba: Resultado exitoso. El Keylogger se desinstala exitosamente de la computadora infectada mediante el archivo Desinfect.bat. Aprobación del cliente en un 100%	

ANEXO III. MANUAL DE USUARIO

El manual de usuario del keylogger, se muestra en el video al cual se puede acceder por medio del siguiente enlace:

ANEXO IV. MANUAL DE INSTALACIÓN

El código fuente y sus respectivas instrucciones a seguir para la instalación del keylogger, se encuentra en un repositorio de GitHub, a continuación, el link del proyecto: https://github.com/JoelDiaz93/Cpp-Keylogger