# ESCUELA POLITÉCNICA NACIONAL

## FACULTAD DE INGENIERÍA EN SISTEMAS

## SISTEMA DE PRE-PLANIFICACIÓN DE ASIGNATURAS DEL PERÍODO ACADÉMICO ORDINARIO PARA LA FIS: UN ENFOQUE ÁGIL EN EL DESARROLLO DE FRONT-END Y BACK-END

## DESARROLLO DE BACK-END

**TRABAJO DE INTEGRACIÓN CURRICULAR PRESENTADO COMO REQUISITO PARA LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SOFTWARE**

**ALEX DANIEL VELASTEGUI SANTAMARÍA**

**alex.velastegui@epn.edu.ec**

**DIRECTOR: CARLOS EFRAÍN IÑIGUEZ JARRÍN**

**carlos.iniguez@epn.edu.ec**

**DMQ, agosto 2023**

# CERTIFICACIONES

Yo, ALEX DANIEL VELASTEGUI SANTAMARÍA 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.

_____

**ALEX DANIEL VELASTEGUI SANTAMARÍA**

Certifico que el presente trabajo de integración curricular fue desarrollado por ALEX DANIEL VELASTEGUI SANTAMARÍA, bajo mi supervisión.

_____

**CARLOS EFRAÍN IÑIGUEZ JARRÍN**
**DIRECTOR**

# 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.

ALEX DANIEL VELASTEGUI SANTAMARÍA

CARLOS EFRAIN IÑIGUEZ JARRIN

# DEDICATORIA

Dedico este trabajo a mis padres, quienes han sido un pilar fundamental en mi desarrollo, tanto como persona, como profesional. Su amor y apoyo a lo largo de toda mi vida han sido fundamentales para alcanzar este logro. A ustedes, queridos padres, les dedico este trabajo como un gesto de gratitud por todo lo que han hecho y sacrificado por mí.

# AGRADECIMIENTO

Agradezco sinceramente a todas las personas que contribuyeron en la realización de este trabajo de integración curricular. En especial a mi tutor, Carlos Iñiguez, por su orientación y acompañamiento durante todo este trayecto. De igual manera, quiero agradecer a mis amigos y compañeros de clase, su apoyo fue fundamental para superar todos los desafíos que se presentaron durante la carrera. Además, agradezco a la Escuela Politécnica Nacional que, durante 5 años me formó como profesional y me preparó para esta nueva etapa de mi vida, las experiencias vividas en este tiempo me formaron tanto como persona como profesional. Por último, quiero expresar mi profundo agradecimiento a mi familia, por la confianza que depositaron en mí y su apoyo para seguir adelante. Cada uno de ustedes han dejado una huella en mí y por eso estoy profundamente agradecido.

# ÍNDICE DE CONTENIDO

# RESUMEN

El Back-End es un componente relevante en el desarrollo de aplicaciones ya que brinda los procesos necesarios para que la aplicación funcione correctamente. El presente documento describe el desarrollo del Back-End del sistema web para la pre-planificación de asignaturas del período académico ordinario para la Facultad de Ingeniería de Sistemas (FIS) de la Escuela Politécnica Nacional.

El sistema ayudará a la FIS a estimar la cantidad de estudiantes que se matricularán en cada una de las asignaturas el siguiente periodo académico ordinario. En base a la estimación obtenida, la FIS puede planificar la cantidad de cursos requeridos para cada una de las asignaturas.

El desarrollo del Back-End sigue el enfoque ágil de Scrum como marco de trabajo, dividiendo el desarrollo en siete sprints. El documento dedica una sección para cada sprint, describiendo las historias de usuario planificadas, su implementación, la revisión del incremento y la sesión de retrospectiva que cierra el sprint. Previo a la descripción del desarrollo del componente, se detalla la tecnología, infraestructura, frameworks, librerías, etc, utilizadas.

El resultado es un componente Back-End con endpoints probados y listos para ser consumidos por el Front-End y formar en conjunto el sistema de pre-planificación.

**PALABRAS CLAVE:** FIS, pre-planificación, sistema web, Scrum, Back-End, endpoints.

# ABSTRACT

The Back-End is a relevant component in application development since it provides the necessary process for the correct functioning of the application. This document describes the Back-End component development for the regular academic period pre-planning web system. This system is addressed for Facultad de Ingeniería en Sistemas (FIS) of the Escuela Politécnica Nacional.

The system aims to support FIS in estimating the number of students who will enroll in each subject in the upcoming regular academic period. Based on the estimation, FIS can adequately allocate the required courses for each subject.

The Back-End development follows the Scrum agile approach, breaking down the development process into seven sprints. Each sprint is detailed in a specific document section, describing planned user stories, their implementation, increment review, and the concluding retrospective session. Before the component's development details, an overview of the technologies, infrastructure, frameworks, libraries, etc., employed is provided.

The outcome is a Back-End component with endpoints tested and ready to be consumed by the Front-End to collectively constitute the pre-planning system.

**KEYWORDS:** FIS, pre-planning, web system, Scrum, Back-End, endpoints.

# *1 DESCRIPCIÓN DEL COMPONENTE DESARROLLADO*

La apertura de los cursos, que se brindarán en un nuevo ciclo académico, también conocido como *Periodo Académico Ordinario* (PAO), es una de las actividades relevantes que recae en las autoridades académicas de la Facultad de Ingeniería de Sistemas (FIS) de la Escuela Politécnica Nacional.

Para abrir o cerrar cursos de un nuevo PAO, las autoridades necesitan conocer el número estimado de estudiantes que potencialmente se inscribirían en una u otra asignatura del nuevo PAO. Para calcular este número de manera estimada, la FIS ejecuta antes de concluir el ciclo académico activo, una encuesta dirigida a los estudiantes donde ellos pueden indicar las asignaturas que potencialmente cursarán el siguiente PAO. La encuesta se considera como una "preplanificación" debido a que, en función de los datos resultantes de la encuesta, las autoridades pueden tomar decisiones sobre cómo planificar la apertura o clausura de cursos en el nuevo PAO.

La malla curricular es un instrumento importante por considerar por los estudiantes de tercer nivel al momento de inscribirse en una asignatura. La malla curricular es un plan de estudios que especifica qué asignaturas se pueden cursar en cada nivel. Estas asignaturas pueden tener las siguientes restricciones [1]:

- Prerrequisitos: Conjunto de asignaturas que el estudiante debe haber aprobado previamente para cursar la asignatura a inscribirse.
- Pisos: Número de asignaturas que deben ser previamente aprobadas para cursar la asignatura.
- Correquisitos: Conjunto de materias del mismo nivel que el estudiante debe tomarlas juntas.

Es importante que la encuesta de "preplanificación" considere dichas restricciones. No obstante, la encuesta de "preplanificación" de la FIS está implementada a través de un Formulario de Google que, por limitaciones de la propia herramienta, no permite establecer las restricciones mencionadas, lo cual genera actualmente respuestas falsas de parte de los estudiantes. Por ejemplo, el estudiante puede seleccionar, como asignaturas posibles a cursar, cualquier asignatura de la malla curricular (desde primer nivel a noveno nivel)[2]. Es decir, el estudiante puede seleccionar asignaturas que ya ha aprobado, asignaturas que aún no ha aprobado o asignaturas cuya restricción es haber aprobado otras asignaturas que el estudiante aún no ha aprobado. En este sentido, los obtenidos por las autoridades

de la FIS están sujetos a errores y no aportan valor para tomar decisiones sobre qué cursos deben ser parte del nuevo PAO.

Como solución al problema, el objetivo del proyecto macro al cual está sujeto el presente trabajo, propone la creación de un sistema web que lleve a cabo la encuesta de preplanificación y que permita personalizar la encuesta a la realidad académica de cada estudiante. La personalización involucra que la encuesta pueda considerar dos aspectos claves: i) las asignaturas que cada estudiante ha cursado y aprobado con el fin de no permitir en la encuesta escoger asignaturas que el estudiante ya ha aprobado y ii) mostrar únicamente las asignaturas que cumplan con los prerrequisitos y pisos. De esta manera, los resultados arrojados por la encuesta permitirán a las autoridades de la FIS tomar decisiones reales para planificar los cursos que serán parte del nuevo PAO.

La implementación del sistema propuesto involucra el desarrollo de dos componentes principales: Front-End y Back-End. El desarrollo del Back-End, específicamente, involucra identificar y analizar las reglas del negocio para posteriormente traducir estas reglas a un lenguaje de programación.

Este componente se enfoca en el desarrollo del Back-End del sistema web para la preplanificación de asignaturas para los estudiantes de pregrado de la FIS, en sus diferentes carreras. El componente debe considerar la situación académica de cada estudiante además de las restricciones de acuerdo con la malla curricular de la FIS [3].

## 1.1 Objetivo general

Desarrollar el Back-End del sistema de preplanificación del PAO que permita comunicarse con el Front-End mediante servicios web, considerando las reglas del negocio de la malla curricular para cada estudiante.

## 1.2 Objetivos específicos

1. Identificar las reglas del negocio
2. Diseñar la arquitectura de software
3. Exponer las reglas del negocio para consumo del Front-End

## 1.3 Alcance

El alcance de este componente es el desarrollo del Back-End del sistema de preplanificación del PAO. El desarrollo incluye la implementación de las reglas del negocio y su exposición mediante APIs. En función del marco de trabajo escogido (Scrum), el desarrollo del Back-End se realizará a través de varios Sprints.

## 1.4 Marco teórico

### 1.4.1 Aplicaciones Web y su desarrollo

Una aplicación web, es un software que se ejecuta sobre un servidor web y es accedida mediante un navegador [4]. Su uso es ampliamente extendido en áreas como: comercio electrónico, salud, entrenamiento, entre otros [5]. Las aplicaciones web han tomado importancia debido a que permiten a los usuarios acceder, con conexión a internet, a sus servicios en cualquier parte del mundo y en cualquier horario [6].

Para el desarrollo de aplicaciones web, el Consorcio de la WWW (World Wilde Web), introdujo varios conceptos fundamentales, entre los que destacan: i) un lenguaje de marcas denominado HTML, para escribir documentos electrónicos, que pueden contener información y enlaces hacia otros documentos (HTML), ii) un protocolo para la comunicación de los documentos HTML entre cliente y servidor (HTTP) y iii) un método para nombrar o referirse a los documentos (URL) [5].

HTML (Hypertext Markup Language) es el lenguaje de marcado estándar utilizado en la creación de páginas web [7]. Mientras que HTML define la estructura de una página web, otro lenguaje de marcado conocido como CSS (Cascading Style Sheets) define aspectos de presentación (apariencia visual) a la estructura [8]. Separar la estructura de la presentación visual permite que el desarrollo web se torne más flexible y eficiente.

HTTP (Hypertext Transfer Protocol) es el protocolo de intercambio información entre el servidor y el cliente [9]. HTTP define nueve métodos para establecer la comunicación cliente-servidor. La **Tabla 1.1** muestra los métodos (también denominados “operaciones”) junto con su uso.

**Tabla 1.1.** Métodos de petición HTTP [10]

| **Método** | **Uso** |
|---|---|
| OPTIONS | Descripción de las opciones de comunicación |
| GET | Recuperación de recursos |

| | |
|---|---|
| HEAD | Similar a una petición GET, pero solamente obtiene la cabecera de la respuesta. |
| POST | Envío de nuevos recursos |
| PUT | Actualización de recursos existentes |
| DELETE | Eliminación de recursos |
| TRACE | Prueba de bucle de retorno de mensaje a lo largo de la ruta hacia el recurso de destino |
| CONNECT | Establecimiento de un túnel de conexión hacia el servidor |
| PATCH | Actualización parcial de recursos |

En términos simples, el desarrollo de una aplicación web consiste en el desarrollo de dos componentes: Back-End y Front-End [11]. El componente Front-End hace referencia a todo con lo que el usuario puede interactuar, es decir, las interfaces del usuario (IU) de la aplicación web [12]. El desarrollo de las IU involucra generalmente combinar HTML, CSS y JavaScript. El componente Back-End hace referencia a la lógica de negocio de la aplicación web (las funcionalidades que tiene la aplicación web) y su desarrollo involucra tratar con el servidor web que aloja la aplicación, la implementación de la lógica de negocio con un lenguaje de programación y la recuperación de datos, generalmente de una base de datos [12].

La comunicación entre Back-End y Front-End se realiza a través de APIs y el protocolo HTTP [13]. La **Figura 1.1** muestra de manera general la secuencia de comunicación [14]: 1) el Front-End realiza una petición usando una de las operaciones HTTP, 2) el Back-End procesa la petición, obteniendo o actualizando la información de la base de datos cuando sea necesario, 3) el Back-End envía una respuesta hacia el Front-End, 4) una vez recibida la respuesta, el Front-End actualiza la IU en caso de ser necesario.

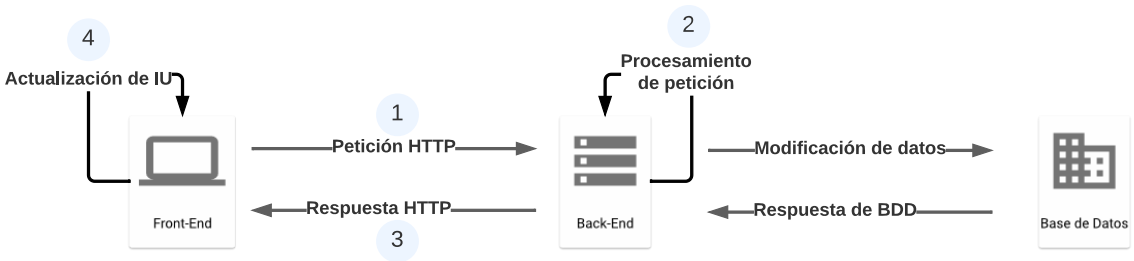



**Figura 1.1.** Secuencia de comunicación entre Back-End y Front-End

Como se puede deducir, los propósitos del Back-End son diferentes del Front-End, por lo tanto, las habilidades y conocimientos técnicos que requiere para abordar el desarrollo de

cada componente es diferente [15]. No obstante, la división entre los dos componentes permite a los desarrolladores no solo concentrarse en sus respectivas áreas sino también trabajar en paralelo permitiendo que la eficiencia y eficacia del desarrollo aumente considerablemente [15].

Considerando que el presente documento hace referencia al desarrollo del Back-End, las siguientes secciones profundizarán en este componente.

### 1.4.2 Desarrollo Back-End en contexto

El desarrollo del Back-End incluye: implementar lógica del negocio, administrar la base de datos y crear APIs que hacen posible la comunicación con el Front-End [14].

#### *1.4.2.1 API REST*

En [16] se define una API (Application Programming Interface) como “un conjunto de definiciones y protocolos que se usan para diseñar e integrar el software de aplicaciones”. Considerando esta definición en este proyecto, la API es el mecanismo por el cual se habilita la comunicación entre los dos componentes del proyecto: Front-End y Back-End.

Existen varias tecnologías para crear una API. No obstante, una comúnmente usada es REST (REpresentional State Transfer) [17]. REST es un estilo arquitectónico en donde las representaciones de los conceptos de negocio de la aplicación, denominados “recursos”, son expuestas como “interfaces” (funcionalidades de la aplicación) que pueden ser accedidas y consumidas mediante solicitudes HTTP [18]. Por lo tanto, una API REST, es aquella API que respeta esta arquitectura.

Para exponer la lógica del negocio mediante APIs REST, se definen *endpoints* (un URL de una API que responde a una solicitud HTTP). Un endpoint corresponde a un recurso u operación específica de la aplicación [19]. Por ejemplo, un endpoint puede estar diseñado para exponer una lista de usuarios o para crear un nuevo usuario en la base de datos. Para garantizar que la exposición de los endpoints sea segura, es necesario implementar mecanismos de autenticación y autorización para controlar el acceso a la lógica del negocio. Estos mecanismos se pueden conseguir con técnicas como OAuth, que es un estándar libre que permite a los usuarios acceder a los endpoints sin necesidad de compartir sus credenciales. Esto se logra creando un *token* (credencial) para cada usuario, que luego se utiliza para autenticar y autorizar sus solicitudes [20].

#### *1.4.2.2 JavaScript*

En el desarrollo del Back-End, existe una larga lista de lenguajes de programación, como por ejemplo PHP, Java, Python y JavaScript, siendo este último uno de los más utilizados actualmente [12].

Históricamente, las empresas de desarrollo de software no solían optar por JavaScript como lenguaje de programación para el desarrollo web. En su lugar, preferían usar PHP, Java o Python, relegando el uso de JavaScript únicamente al mejoramiento de la interactividad de las páginas HTML, es decir, la interfaz de usuario. No obstante, con la llegada de AJAX y la Web 2.0, JavaScript surgió como una opción competitiva frente a los lenguajes de programación previamente mencionados. Existen varias razones por las que se ha popularizado el uso de JavaScript en el desarrollo web. Destacan dos en particular [21]: 1) La versatilidad del lenguaje permite hacer uso de este tanto en Front-End, como en Back-End, promoviendo la reutilización de código y reduciendo la complejidad al no manejar múltiples lenguajes en un solo proyecto, y 2) la capacidad de integrarse perfectamente con API REST mediante JSON.

JavaScript al ser un lenguaje no tipado, complica la detección de errores durante el desarrollo, lo que resulta en un difícil mantenimiento de la aplicación [22]. Para solucionar este problema se ha desarrollado TypeScript, un lenguaje que extiende de JavaScript con mayores funcionalidades como la capacidad para crear interfaces, el tipado de datos, entre otros; bondades que han motivado la preferencia de usar TypeScript para el desarrollo del Back-End. Es importante mencionar que, para poder ejecutar una aplicación escrita en TypeScript, el código debe ser *transpilado* (traducido) a JavaScript para luego ser ejecutado en entornos como Node.js o navegadores web [23].

La combinación de las razones previamente mencionadas para el uso de JavaScript, junto con las ventajas añadidas de TypeScript, han hecho que este último, sea el escogido para el desarrollo del proyecto, especialmente, del componente Back-End.

#### *1.4.2.3 Bases de datos*

Dos de las categorías más utilizadas y populares de bases de datos son: relacionales y no relacionales. MongoDB y Amazon Neptune son ejemplos de motores de bases de datos dentro de la categoría “no relaciones”. Por otro lado, MySQL, PostgreSQL y SQL Server son ejemplos de motores de bases de datos dentro de la categoría “relacionales”. El uso de una u otra categoría de base de datos dependerá de las necesidades específicas del negocio, considerando factores como rendimiento y escalabilidad [24].

SQL Server es un motor de base de datos relacional desarrollado por Microsoft. Es ampliamente utilizado en el desarrollo web, por su capacidad para manejar grandes volúmenes de datos y su alta escalabilidad. Una gran ventaja de este motor es su integración con otros productos desarrollados por Microsoft como Azure, para desplegar el servidor en la nube, y Power BI, para analizar la información y tomar decisiones de negocio

[25]. Estas ventajas han motivado la selección de SQL Server como motor de base de datos para el desarrollo de este componente del proyecto.

#### *1.4.2.4 Arquitectura MVC*

La arquitectura de software define la estructura de alto nivel de un sistema. Establecer la arquitectura de software para el Back-End ayuda al correcto funcionamiento de una aplicación web. Una de las arquitecturas más utilizadas es el patrón Modelo-Vista-Controlador (MVC) la cual enfatiza la separación de la lógica de la aplicación en tres componentes principales: Modelo, Vista y Controlador [26].

- El modelo corresponde a la lógica de negocio y la estructura de datos.
- La vista es responsable de la presentación de la información mediante la interfaz gráfica, es decir, el Front-End.
- El controlador es el encargado de recibir las acciones del usuario, procesarlas usando el componente “modelo” (lógica del negocio y datos), actualizar la información de ser necesaria y entregarla al componente “vista” para su presentación.

La **Figura 1.2** muestra la interacción entre los tres componentes, donde el controlador es el enlace entre el modelo y la vista.

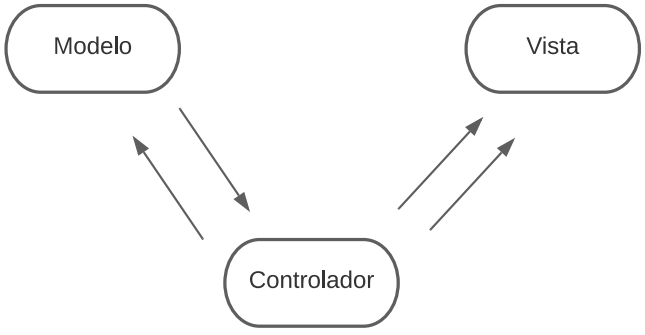



**Figura 1.2.** Interacción entre los componentes MVC

El principal beneficio de esta decisión de arquitectura es el trabajo en paralelo. Por ejemplo, mientras un desarrollador implementa la vista, otro desarrollador, en paralelo, puede implementar el controlador [27]. Esto reduce la carga de trabajo de cada desarrollador, haciendo la tarea del desarrollo más rápida y eficiente.

El presente proyecto se basa en la arquitectura MVC, debido a que permite el aislamiento de la lógica del negocio, evitando que el Front-End se vea afectado por cambios en el Back-End. Además, hace posible el trabajo en paralelo de estos dos componentes.

#### *1.4.2.5 Framework NestJS*

En el ámbito del desarrollo de software, un framework se define como un conjunto de herramientas desarrolladas previamente: características predefinidas, bibliotecas, módulos y patrones de diseño [28]. Este conjunto de herramientas permite a los desarrolladores centrarse en la implementación de la lógica del negocio, aumentando la eficiencia y eficacia del desarrollo.

NestJS es un framework para el desarrollo de aplicaciones Back-End eficientes y escalables. Usa el lenguaje de programación JavaScript, sin embargo, también soporta TypeScript. Este framework combina los principios de programación de varios paradigmas: Programación Orientada a Objetos (POO), Programación Funcional y Programación Funcional Reactiva [29].

Un proyecto en NestJS, normalmente tiene la siguiente estructura [30]:

- '/src': Directorio en el que se encuentra todo el código de la aplicación
- 'main.ts': Este archivo es el punto de entrada de la aplicación. Inicializa la aplicación e inicia el servidor.
- 'app.module.ts': Este archivo es el módulo principal. Define los providers, controllers y modules que componen la aplicación.
- 'tsconfig.json': Archivo de configuración de TypeScript para el proyecto.

Cuando se utiliza TypeScript para el proyecto, es necesario transpilarlo a JavaScript para que pueda ser ejecutado en el servidor. Ejecutar esta acción, crea un directorio llamado 'dist' dentro del proyecto. Este directorio es el que aloja todo el código de la aplicación escrito en JavaScript.

NestJS provee un conjunto de componentes que facilitan el desarrollo, de los cuales, en la **Tabla 1.2** se muestra aquellos que se han utilizado en el desarrollo del presente proyecto:

**Tabla 1.2.** Componentes del framework NestJS [29]

| **Componente** | **Decorador** | **Descripción** |
|---|---|---|
| Controllers | @Controller() | Son clases que se hacen responsables de manejar las peticiones HTTP, delegar tareas complejas a los providers y devolver respuestas al cliente |
| Providers | @Injectable() | Son clases que pueden ser inyectadas en otros componentes. |

| | | |
|---|---|---|
| Services | @Injectable() | Son un tipo de providers que manejan la lógica del negocio y el acceso a los datos |
| Pipes | @Pipe() | Son usados para transformar datos antes de que estos sean usados por un provider o un controller. |
| Guards | @Injectable() | Son clases que implementan la interfaz *CanActivate.* Tienen una única función, la de controlar el acceso basado en ciertas condiciones. Por ejemplo, pueden ser usados para validar que el token de acceso sea válido antes de acceder a los recursos. |
| Interceptors | @Injectable() | Son clases que implementan la interfaz *NestInterceptor*. Son usadas para interceptar peticiones y respuestas. |
| Filters | - | Son usados para manejar excepciones que pueden ocurrir durante el procesamiento de las peticiones. |
| Modules | @Module() | Es una clase que provee los metadatos necesarios para organizar la estructura de la aplicación. Define grupos de componentes que encajan como una parte modular de una aplicación global. Estos componentes son agrupados en 4 categorías:<br>• Providers: Componentes que serán inyectados y pueden ser ocupados a lo largo de todo el módulo.<br>• Controllers: Conjunto de controladores que deben ser instanciados.<br>• Imports: Módulos importados que exportan los Providers que son requeridos en este módulo<br>• Exports: Subconjunto de Providers que son dados por este módulo y van a estar disponibles en otros módulos |

En el presente proyecto, es necesario realizar entregas periódicas de la aplicación, lo que implica que el framework de desarrollo debe permitir que la aplicación sea escalable. Basado en esta necesidad, el framework escogido para el proyecto es NestJS.

#### *1.4.2.6 TypeORM*

El término ORM (*Object Relational Mapping* o Mapeo Relacional de Objetos) hace referencia a una técnica de programación que permite a los desarrolladores interactuar con

una base de datos usando el paradigma orientado a objetos. Las herramientas que implementan la técnica ORM proveen una capa de abstracción entre la base de datos y el lenguaje de programación, la cual permite a los desarrolladores trabajar con los datos de una manera más intuitiva y eficiente [31].

TypeORM es un ORM que soporta tanto TypeScript como JavaScript (ES5, ES6, ES7, ES8) y es compatible con algunas bases de datos como MySQL, SQLite y SQL Server [32]. La **Tabla 1.3** muestra algunos componentes de TypeORM que permiten el manejo de las bases de datos y que han sido utilizados en el desarrollo del presente componente del proyecto.

**Tabla 1.3.** Componentes de TypeORM [32]

| **Componente** | **Función** |
|---|---|
| DataSource | Es un objeto de configuración en el que se especifican los detalles de conexión a la base de datos y establecer la conexión con la misma. |
| Entities | Son clases que se mapearan a tablas en la base de datos. Dentro de estas clases se definen las columnas que conformarán las tablas. |
| Relations | Define las asociaciones que existen entre las diferentes entidades. Las relaciones posibles son uno a uno, uno a muchos, muchos a uno y muchos a muchos. En la relación muchos a muchos, el ORM crea la tabla intermedia necesaria. |
| EntityManager | Es el objeto principal que se usa para interactuar con la base de datos. Permite la creación, consulta, actualización y eliminación de los datos de cualquier entidad. |
| Repository | Es un objeto que permite interactuar con la base de datos. Es similar al EntityManager, sin embargo, sus operaciones están limitadas a una sola entidad. |
| QueryBuilder | Permite construir una sentencia SQL compleja. Soporta varios operadores, entre ellos: joins, subqueries y agregaciones. |
| Migrations | Son scripts que permiten realizar cambios a la base de datos sobre tablas, índices e incluso modificar los datos. |

El ORM facilita la interacción fluida entre la aplicación y la base de datos. En este proyecto, se eligió TypeORM como ORM debido a su compatibilidad con TypeScript y su soporte a varios motores de bases de datos.

### 1.4.3 DevOps

DevOps es un enfoque de desarrollo de software en el que se pone como prioridad la colaboración y comunicación entre el equipo de desarrollo y el equipo de operaciones. Este enfoque combina el desarrollo ágil, la integración y la entrega continua para crear un proceso de desarrollo mucho más eficiente. Mientras el desarrollo ágil ayuda a tener entregas de valor continuas, en donde se recibe retroalimentación acerca del producto, la integración y la entrega continua ayuda a detectar problemas de integración más rápido que en el desarrollo tradicional [33].

CI (Continuous Integration) es un componente clave para DevOps. Es una práctica donde los desarrolladores integran los cambios realizados en el código de manera continua en un repositorio compartido, en donde se realizarán las pruebas y se construirá el producto de manera automática. El propósito de esta práctica es encontrar y corregir errores lo más pronto posible en el ciclo de desarrollo, mejorando la calidad del software resultante [34].

Una plataforma que ayuda a la implementación de DevOps es Azure DevOps. En esta plataforma se encuentra un conjunto de servicios y herramientas, que ayudan a los usuarios a manejar proyectos de software desde la planificación hasta el despliegue. Algunos de los servicios/herramientas que ofrece son [35]:

- Azure Boards: Es una herramienta ágil que facilita la planificación de historias de usuario, permitiendo la división de estas en tareas. Usa el método Kanban para llevar el control de las historias y tareas.

- Azure Repos: Es un cliente de Git que permite la creación de repositorios para llevar el control del código fuente. Git, por su parte, es un sistema de control de versiones distribuido diseñado para llevar un control de los cambios realizado en archivos y/o directorios a lo largo del tiempo, especialmente cuando varias personas trabajan en el mismo proyecto [36].

- Azure Pipelines: Es una herramienta que facilita servicios de construcción y despliegue del software de manera eficiente. Es aquí en se lleva a cabo la integración continua, pues, con cada *commit* (cambio) que se realice en el repositorio, se inicia un proceso automático que generalmente incluye pruebas, construcción y despliegue de software. Con este proceso se comprueba que no existan problemas de integración con el código actualizado. Las tareas por realizar en este proceso automático son configuradas por el usuario.

- Azure Test Plans: Es una suite de herramientas que proporciona funcionalidades para realizar pruebas manuales y exploratorias, garantizando así la calidad del producto software antes de su lanzamiento.
- Azure Artifacts: Es una herramienta que permite a los desarrolladores crear, almacenar librerías de código de manera eficiente. Las librerías puedes ser compartidas con el equipo de trabajo y utilizadas dentro de los pipelines de manera sencilla.

En el presente proyecto se utiliza la plataforma Azure DevOps como parte de la implementación de DevOps, destacando el uso principal de tres herramientas clave: Azure Boards, Azure Repos y Azure Pipelines.

### 1.4.4 Scrum

Scrum es un marco de trabajo ágil que permite a las organizaciones abordar problemas complejos, a través de un enfoque iterativo incremental que entregue valor continuamente a través de soluciones adaptativas. Este marco de trabajo se basa en el empirismo en donde la base del conocimiento es la experiencia. Permite probar nuevas ideas rápidamente para determinar su eficacia y descartarlas de manera temprana en caso de que no funcionen [37].

En Scrum, el equipo de trabajo está compuesto por un Scrum Master, un Product Owner y desarrolladores. En este equipo multifuncional todos tienen las habilidades necesarias para terminar cada iteración, también llamadas “Sprint”. Además, el equipo es autoorganizado, lo que implica que toman decisiones internas sobre cómo realizar un trabajo en específico. Es ideal que el tamaño de este equipo sea menor o igual a diez personas y que la duración del Sprint sea de dos semanas [37].

Durante cada Sprint, suceden 4 eventos clave: planificación del Sprint, reuniones diarias o dailies, revisión del Sprint y retrospectiva del Sprint. En la planificación, a partir de un objetivo dado por el Product Owner, se seleccionan las características *del product backlog* (lista priorizada de características) que se realizarán dentro del Sprint en curso. Los dailies, son reuniones breves de máximo 15 minutos, en donde se comunica el progreso de cada miembro del equipo. En la revisión del sprint, se muestra el incremento realizado y se reciben comentarios de los stakeholders. La retrospectiva, es una reunión en donde el equipo analiza el trabajo realizado en el Sprint pasado en busca de oportunidades de mejora para el siguiente [37].

Scrum, tiene ciertos pilares esenciales que se deben adoptar en el equipo: transparencia, inspección y adaptación. La ***transparencia*** implica que todo proceso realizado en el proyecto junto con sus resultados debe ser visible para todos los implicados. La ***inspección*** implica que tanto el progreso como los resultados de los procesos deben ser inspeccionados frecuentemente con el fin de detectar problemas de manera temprana. La ***adaptación*** implica que, si luego de la inspección se identifica que algo no va de acuerdo con el plan, se deben realizar ajustes en el mismo para minimizar los problemas y maximizar el valor entregado [37].

Para que se puedan cumplir estos 3 pilares es necesario que los implicados en el proyecto vivan los valores del Scrum. Estos valores son cinco: ***compromiso*** para alcanzar los objetivos del Sprint y apoyarse mutuamente, ***enfoque*** en el trabajo para cumplir los objetivos de cada Sprint, ***apertura*** entre el equipo y los stakeholders sobre el trabajo y los problemas existentes, ***respeto*** entre los miembros del equipo como personas capaces e independientes, ***coraje*** para hacer lo correcto incluso si eso significa empezar desde cero nuevamente [37].

El uso de este marco de trabajo es recomendable en dominios complejos, en donde no existe mucho entendimiento del problema a resolver, existen muchas situaciones impredecibles y solo se puede aprender mediante la retrospectiva. En efecto, Scrum, a través de sus pilares y valores, crea un entorno seguro para el fracaso, permitiendo al equipo de trabajo descubrir información relevante mediante la experimentación [38].

En el presente proyecto se aplica Scrum como marco de trabajo, siguiendo los 4 eventos, garantizando los 3 pilares fundamentales y desarrollando los 5 valores que promulga el marco de trabajo.

# *2 METODOLOGÍA*

Siguiendo el marco de trabajo Scrum (sección 1.4.4), esta sección describe el desarrollo del componente propuesto en un total de siete sprints. En este sentido, el desarrollo del componente se describe en 7 sub-secciones, una por cada sprint realizado. Cada subsección describe un sprint siguiendo una estructura de 5 partes: la primera parte brinda una breve introducción del sprint y los 4 restantes corresponden a los eventos SCRUM que suceden durante cada Sprint (planificación del Sprint, reuniones diarias o dailies, revisión del Sprint y retrospectiva del Sprint):

- INTRODUCCIÓN. – Describe a) el objetivo del Sprint (lo que se espera del incremento al finalizar el Sprint) y b) las decisiones a nivel de diseño.

- PLANIFICACIÓN. – Se corresponde con el evento “Planificación del Sprint” y describe los *Items del Product Backlog* (PBI) seleccionados por el equipo para cumplir el objetivo descrito en la “INTRODUCCIÓN” y por cada PBI, se describe a) la estimación de esfuerzo para lograr el PBI y b) las tareas de grano fino involucradas en la construcción del PBI.
- IMPLEMENTACIÓN. – Se corresponde con el evento “Daily” donde se realiza una sincronización y socialización del trabajo realizado diariamente. En este sentido, esta sección describe, desde el punto de vista arquitectónico de sistema, las decisiones de ingeniería involucradas en la construcción de cada PBI y que fueron comunicadas al equipo durante las reuniones “daily”.
- REVISIÓN. – Como se mencionó en la sección 1.4.4, la “revisión del sprint” es un evento de SCRUM donde se muestra el incremento realizado en el Sprint y se reciben comentarios de los stakeholders. Parte de este evento es analizar el esfuerzo que llevó lograr el incremento resultante del Sprint, para esto, se compara el esfuerzo estimado al inicio del Sprint con el esfuerzo real realizado. En este sentido, esta sub-sección describe dicha comparación por cada PBI.
- RETROSPECTIVA. – Describe conclusiones relevantes que emergieron de la “reunión de la retrospectiva”. Las conclusiones son el resultado del auto análisis llevado a cabo por el equipo durante el desarrollo del Sprint y se convierten en oportunidades de mejora a considerarse en el desarrollo de los posteriores Sprint.

## 2.1 SPRINT 1

### 2.1.1 INTRODUCCIÓN

Este sprint tiene por objetivo “Elaborar formularios por carrera y malla curricular”. Se busca crear el formulario de preplanificación para los estudiantes. Se considerará la “carrera” como categoría de clasificación de los estudiantes.

En este sprint, se tomaron las siguientes decisiones para el componente Back-End:

1. Aplicar MVC como arquitectura de software, en donde el modelo y el controlador son parte del Back-End y la vista es parte del Font-End.
2. Usar NestJS y TypeORM para el desarrollo.
3. Usar Azure para albergar el servidor web y el servidor de base de datos.
4. Usar Azure DevOps para lograr la integración y el despliegue continuo.

5. Usar WebStorm como IDE.

### 2.1.2 PLANIFICACIÓN

En este sprint se planificaron 9 historias de usuario, la **Tabla 2.1** muestra, en detalle, cada una de las historias de usuario escogidas para el sprint, con su estimación y la división en tareas. Es importante mencionar que las tareas especificadas corresponden únicamente al componente Back-End y que las tareas relacionadas con el componente Front-End no son tomadas en cuenta en este documento.

**Tabla 2.1.** División de tareas del sprint 1

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 46. Completar encuesta: Como un estudiante<br>deseo completar la encuesta con las asignaturas<br>para reservar un cupo en la matricula del próximo semestre | 8 | • Crear modelo Período<br>• Crear endpoint para Login usando correo y contraseña<br>• Crear endpoint para exponer asignaturas por estudiante<br>• Crear modelo asignatura<br>• Crear modelo estudiante |
| 47. Verificar encuestas según malla: Como un estudiante<br>deseo que mi encuesta verifique los prerrequisitos, correquisitos y pisos de cada asignatura<br>para seleccionar las asignaturas en las cuales puedo matricularme | 8 | • Crear relación entre estudiante y materia<br>• Modificar modelo asignatura para agregar atributos<br>• Modificar endpoint de la asignatura, para marcar si está disponible para un estudiante<br>• Validar asignaturas disponibles con requisitos y correquisitos<br>• Modificar endpoint de asignatura para que solo devuelva las asignaturas de la carrera del estudiante |
| 55. Informar número máximo y mínimo de créditos: Como un coordinador | 2 | • Crear modelo configuración para almacenar valores de tipo clave-valor. |

| | | |
|---|---|---|
| deseo que la encuesta informe a los estudiantes el número máximo y mínimo de créditos para que conozcan cuántos créditos seleccionar | | • Crear endpoint que exponga las configuraciones. |
| 56. Encuesta y malla curricular: Como un coordinador<br>deseo que la encuesta presente los niveles, asignaturas, códigos y créditos de las asignaturas<br>para que los estudiantes tengan información para completar su encuesta | 3 | • Aumentar atributos en el modelo asignatura |
| 48. Completar de acuerdo con carrera: Como un coordinador<br>deseo que los estudiantes completen su encuesta<br>de acuerdo con su carrera. | 3 | • Crear modelo carrera<br>• Relacionar la modelo carrera con las entidades estudiante y asignatura. |
| 53. Indicaciones para estudiantes o grupo de estudiantes: Como un coordinador<br>deseo que la encuesta tenga indicaciones para un estudiante o para un grupo de estudiantes<br>para personalizar las indicaciones/directrices por estudiante o grupo de estudiantes | 3 | NA |
| 96. Diseño encuesta: Como un estudiante deseo que las materias se encuentren divididas por semestres para tener las asignaturas organizadas como la malla curricular | 5 | NA |
| 98. Mensaje de selección de correquisitos: Como estudiante | 3 | NA |

| | | |
|---|---|---|
| deseo que se me informe las materias que son correquisitos para seleccionarlas antes de enviar la encuesta | | |

### 2.1.3 IMPLEMENTACIÓN

De manera resumida y siguiendo la arquitectura MVC, la implementación de las historias de usuario de este Sprint involucró crear e implementar 4 controladores (capa Controlador), así como, 4 servicios y 4 entidades (en la capa Modelo), como lo resume la **Figura 2.1**.

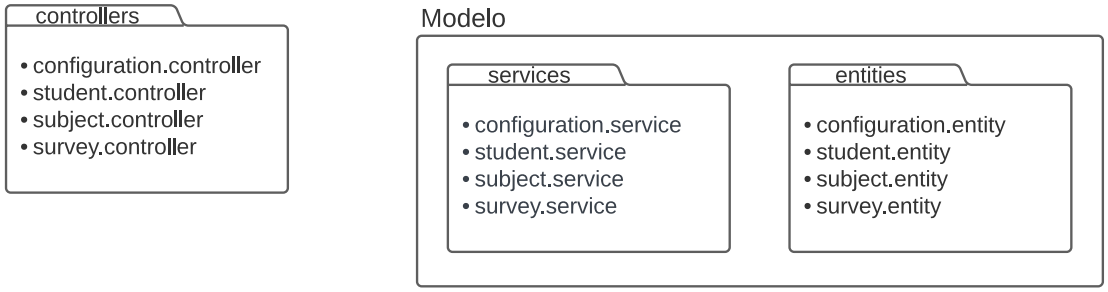



**Figura 2.1.** Controladores y modelos utilizados en el sprint 1

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.1**.

#### *2.1.3.1 Completar encuesta*

Para completar esta historia de usuario se requiere de tres procesos: i) Que el estudiante inicie sesión. ii) Obtener las asignaturas para ese estudiante en particular. iii) Guardar la respuesta del estudiante.

El logro de los tres procesos se muestra en la **Figura 2.2**. El proceso "i" se logra con la interacción del controlador *"student.controller"* y el modelo "*student.entity"* a través del servicio *"student.service"*. El proceso "ii" se logra con la interacción entre el controlador *"subject.controller"* y varias entidades de la Base de Datos. Es importante destacar que toda interacción entre estos componentes mediante un servicio, en este caso, el servicio *"subject.service".* Finalmente, el proceso "iii" se logra interactuando el controlador *"survey.controller"* con el modelo "*survey.entity"* a través del servicio *"survey.service".*

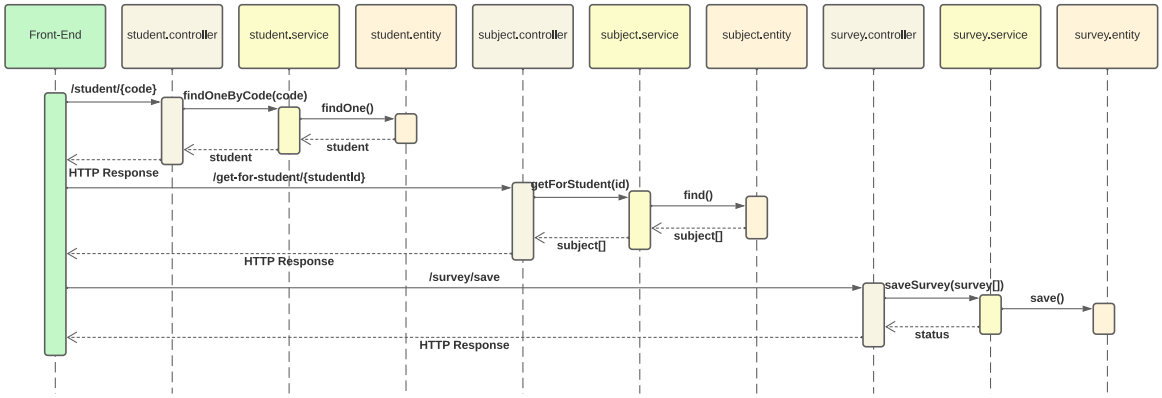



**Figura 2.2.** Procesos: Inicio de sesión de estudiante, Devolver asignaturas por estudiante, y Guardar encuesta de estudiante

*2.1.3.2 Verificar encuestas según malla curricular*

Utilizando la misma interacción definida en la **Figura 2.2**, se modificó el servicio *"subject.service"* para que, en lugar de retornar todas las asignaturas del periodo, retorne únicamente las asignaturas que el estudiante puede tomar.

*2.1.3.3 Informar número máximo y mínimo de créditos*

Los números máximo y mínimo de créditos son configuraciones asociadas a la encuesta. Tomando en cuenta esto, se implementó una entidad *"configuration.entity"* que puede almacenar configuraciones como clave-valor. Esto responde a la necesidad de tener escalabilidad y permitir que las configuraciones asociadas a la encuesta puedan crecer con el tiempo. La **Figura 2.3** muestra la interacción para obtener el número máximo y mínimo de créditos, haciendo uso del controlador *"configuration.controller"*, el servicio "*configuration.service*" y el modelo *"configuration.entity".*

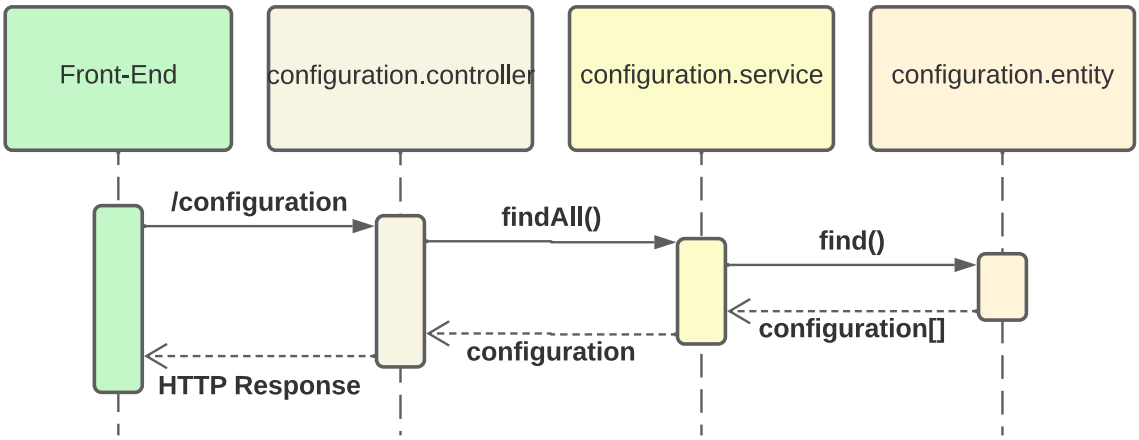



**Figura 2.3.** Exponer configuraciones

*2.1.3.4 Encuesta y malla curricular*

Para lograr esta historia de usuario, se modificó el modelo *“subject.entity”* agregando los nuevos atributos solicitados por el Product Owner:

- Nivel: Semestre referencial en el que se encuentra la asignatura
- Código: Código identificador de la asignatura
- Créditos: Créditos que aporta la asignatura.

*2.1.3.5 Completar de acuerdo con carrera*

Para lograr este objetivo, se creó el modelo *"career.entity"* y se estableció una relación tanto con el modelo *"subject.entity"* como con *"student.entity"*. Esta relación permite determinar a qué carrera pertenece el estudiante y mostrar únicamente las asignaturas asociadas a dicha carrera. De esta manera, se garantiza una visualización precisa y filtrada de las asignaturas correspondientes al estudiante en cuestión.

### 2.1.4 REVISIÓN

La **Tabla 2.2** muestra la comparación entre el esfuerzo estimado inicialmente para todas las historias de usuario en el sprint y el esfuerzo real registrado. Es evidente que la estimación inicial no fue precisa, posiblemente debido a la falta de experiencia del equipo, ya que son relativamente nuevos y aún están aprendiendo a dimensionar adecuadamente las historias.

**Tabla 2.2.** Comparación de estimaciones del sprint 1

| **Historia de usuario** | **Estimación inicial** | **Estimación real** |
|---|---|---|
| 46. Completar encuesta: Como un estudiante<br>deseo completar la encuesta con las asignaturas<br>para reservar un cupo en la matricula del próximo semestre | 8 | 21 |
| 47. Verificar encuestas según malla: Como un estudiante<br>deseo que mi encuesta verifique los prerrequisitos, correquisitos y pisos de cada asignatura<br>para seleccionar las asignaturas en las cuales puedo matricularme | 8 | 8 |
| 55. Informar número máximo y mínimo de créditos: Como un coordinador | 2 | 8 |

| | | |
|---|---|---|
| deseo que la encuesta informe a los estudiantes el número máximo y mínimo de créditos para que conozcan cuántos créditos seleccionar | | |
| 56. Encuesta y malla curricular: Como un coordinador deseo que la encuesta presente los niveles, asignaturas, códigos y créditos de las asignaturas para que los estudiantes tengan información para completar su encuesta | 3 | 3 |
| 48. Completar de acuerdo con carrera: Como un coordinador deseo que los estudiantes completen su encuesta de acuerdo con su carrera. | 3 | 5 |
| 53. Indicaciones para estudiantes o grupo de estudiantes: Como un coordinador deseo que la encuesta tenga indicaciones para un estudiante o para un grupo de estudiantes para personalizar las indicaciones/directrices por estudiante o grupo de estudiantes | 3 | 3 |
| 96. Diseño encuesta: Como un estudiante deseo que las materias se encuentren divididas por semestres para tener las asignaturas organizadas como la malla curricular | 5 | 3 |
| 98. Mensaje de selección de correquisitos: Como estudiante deseo que se me informe las materias que son correquisitos para seleccionarlas antes de enviar la encuesta | 3 | 8 |

### 2.1.5 RETROSPECTIVA

En este sprint se identificó la eficiente asignación de tareas como un punto positivo. Para mejorar, se propone implementar reuniones diarias en videollamada y fortalecer la comunicación con el Product Owner para comprender mejor los requerimientos del proyecto.

## 2.2 SPRINT 2

### 2.2.1 INTRODUCCIÓN

Este sprint tiene por objetivo “Personalizar formularios por estudiante” y busca que el estudiante tenga una encuesta ajustada a sus necesidades.

### 2.2.2 PLANIFICACIÓN

En este sprint se planificaron 4 historias de usuario, la **Tabla 2.3** muestra en detalle cada historia de usuario, junto con su estimación y las tareas necesarias para llevarla a cabo.

**Tabla 2.3.** División de tareas del sprint 2

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 50. Actualizar encuesta: Como un estudiante<br>deseo actualizar mi encuesta<br>para actualizar mi intención de matrícula. | 13 | • Modificar servicio de activar periodo<br>• Exponer endpoint para activar una encuesta |
| 138. Número máximo de créditos: Como un coordinador<br>deseo que la suma de créditos de acuerdo con las asignaturas seleccionadas sea igual o menor a 15 créditos<br>para que los estudiantes se matriculen correctamente con el máximo de créditos | 5 | NA |
| 139. Número mínimo de créditos: Como un coordinador<br>deseo que la suma de créditos de acuerdo con las asignaturas seleccionadas sea igual o mayor a 9 créditos<br>para que los estudiantes se matriculen correctamente con el mínimo de créditos | 5 | NA |
| 141. Visualizar materias actuales del estudiante: Como un estudiante | 2 | • Modificar servicio de obtener estudiante |

| | | |
|---|---|---|
| deseo ver las asignaturas que estoy cursando actualmente para tenerlas en cuenta al momento de llenar mi encuesta. | | |

### 2.2.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.4**.

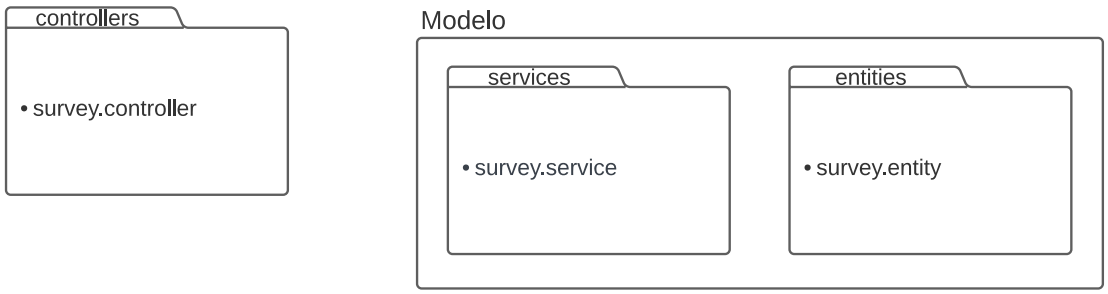



**Figura 2.4.** Controladores y modelos utilizados en el sprint 2

*2.2.3.1 Actualizar encuesta*

Hasta ahora, la creación de encuestas para cada estudiante se realizaba al momento de crear el periodo. No obstante, esto se considera poco eficiente ya que puede haber periodos que no se activen. Por lo tanto, se ha modificado la lógica para crear las encuestas de los estudiantes al activar el periodo, como se muestra en la **Figura 2.4**.

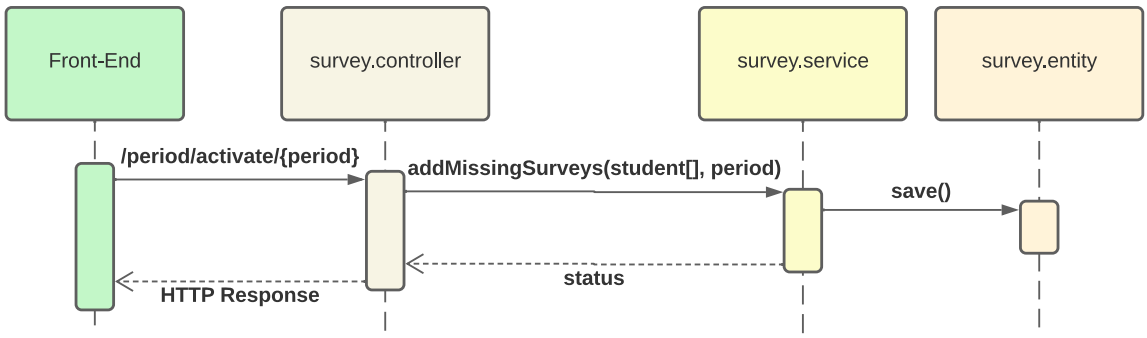



**Figura 2.5.** Agregar encuestas al activar periodo

Una vez que un estudiante envía la encuesta, esta se vuelve no editable; impidiendo al estudiante realizar modificaciones posteriores. Para evitar esta limitación, se implementó un nuevo endpoint que permite activar la encuesta de un estudiante en particular. La **Error!**

**Reference source not found.** representa el funcionamiento detallado de este nuevo endpoint.

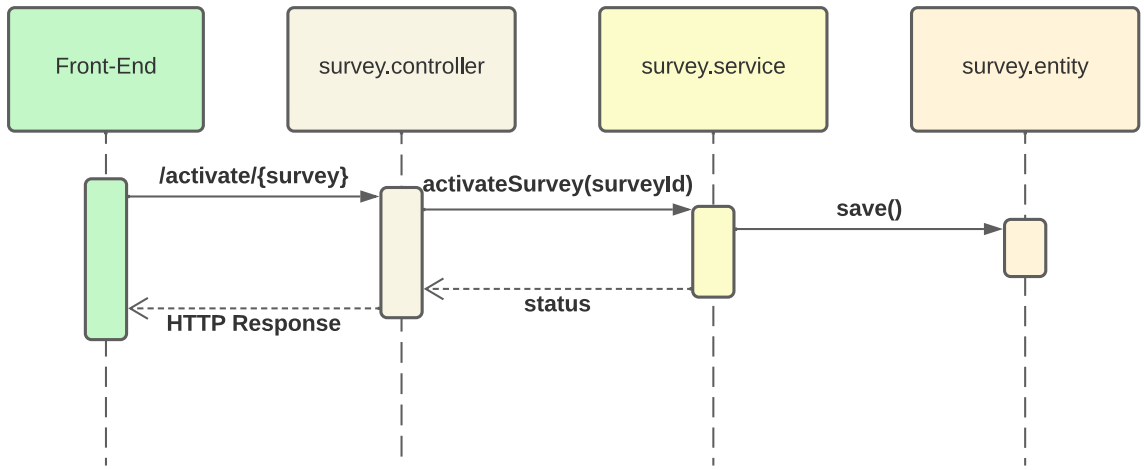



**Figura 2.6.** Activar encuesta de un estudiante

*2.2.3.2 Visualizar materias actuales del estudiante*

Para obtener información completa sobre un estudiante, se utiliza el endpoint indicado en la **Figura 2.2**. Con el fin de aumentar esta funcionalidad, se ha realizado una modificación en el servicio mencionado en la **Figura 2.2**, para que retorne también todas las materias que el estudiante está cursando actualmente.

### 2.2.4 REVISIÓN

En la **Tabla 2.4**, se puede observar la comparativa del esfuerzo estimado al inicio del sprint de todas las historias de usuario, contra el esfuerzo real. En este caso, la estimación inicial coincide en mayor medida con la estimación real. Conforme aumenta la experiencia del equipo, las estimaciones se vuelven más precisas.

**Tabla 2.4.** Comparativa de estimaciones del sprint 2

| Historia de usuario | Estimación inicial | Estimación real |
|---|---|---|
| 50. Actualizar encuesta: Como un estudiante deseo actualizar mi encuesta para actualizar mi intención de matrícula. | 13 | 8 |
| 138. Número máximo de créditos: Como un coordinador deseo que la suma de créditos de acuerdo con las asignaturas seleccionadas sea igual o menor a 15 créditos | 5 | 5 |

| | | |
|---|---|---|
| para que los estudiantes se matriculen correctamente con el máximo de créditos | | |
| 139. Número mínimo de créditos: Como un coordinador<br>deseo que la suma de créditos de acuerdo con las asignaturas seleccionadas sea igual o mayor a 9 créditos<br>para que los estudiantes se matriculen correctamente con el mínimo de créditos | 5 | 5 |
| 141. Visualizar materias actuales del estudiante: Como un estudiante<br>deseo ver las asignaturas que estoy cursando actualmente<br>para tenerlas en cuenta al momento de llenar mi encuesta. | 2 | 2 |

### 2.2.5 RETROSPECTIVA

En este sprint no se realizó la retrospectiva

## 2.3 SPRINT 3

### 2.3.1 INTRODUCCIÓN

El objetivo de este sprint es “Elaborar formularios de administración”. Se propone la construcción del panel administrativo, en donde poder controlar: periodos, mallas curriculares y estudiantes.

Para el desarrollo de este sprint se concluyó que la estructura de objetos de negocio construida hasta el momento (**Figura 2.7**) no era escalable por las siguientes razones:

- No permite varios periodos
- Existen configuraciones generales, es decir, no están asociadas a un periodo, por lo que todos los periodos tendrán las mismas configuraciones
- El estudiante se relaciona directamente con las asignaturas, esto quiere decir que, si en un futuro la malla curricular sufre un cambio, se deberá eliminar los registros actuales para ingresar los nuevos registros.

- Solo se permite una encuesta por estudiante, lo cual significa que, para el siguiente periodo se deberá eliminar la encuesta anterior.

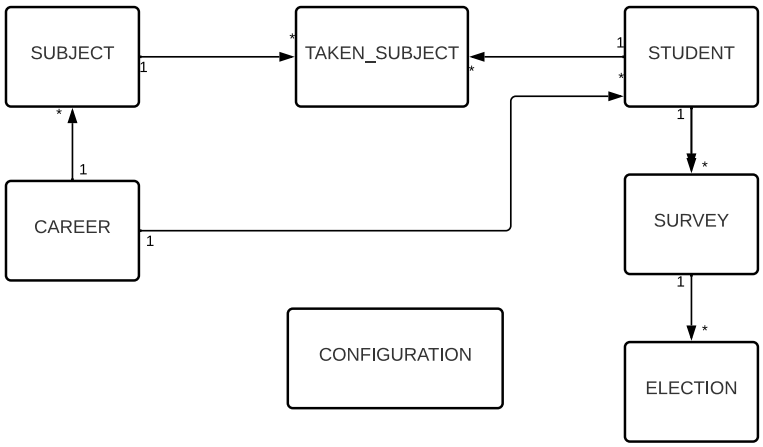



**Figura 2.7.** Estructura antigua de objetos de negocio

En este sentido, se propuso una nueva estructura que permita la creación de varios periodos, con configuraciones, currículos y encuestas para cada uno de ellos. Esta estructura se presenta en la **Figura 2.8**.

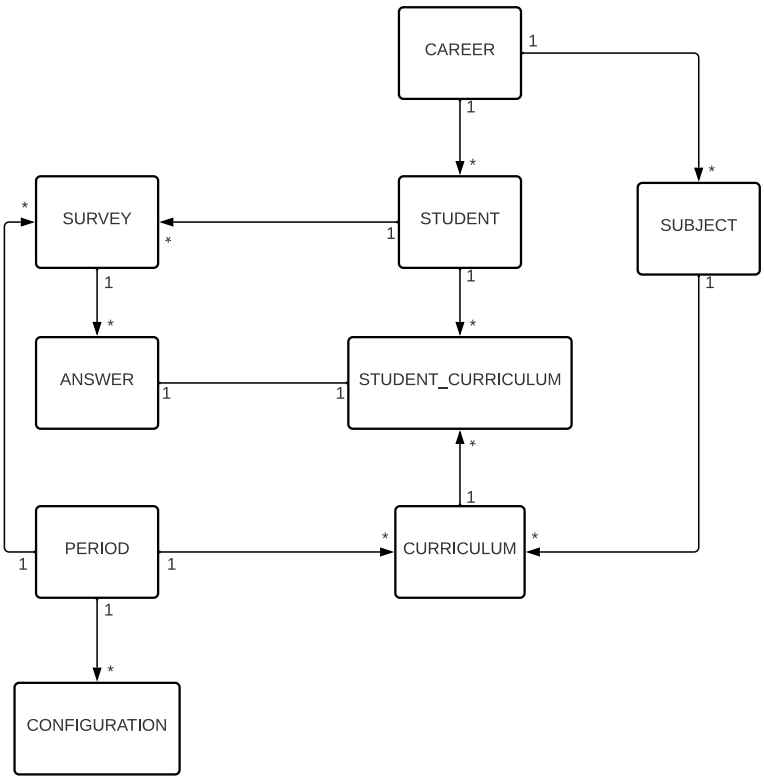



**Figura 2.8.** Nueva estructura de objetos de negocio

Como beneficio adicional de este cambio estructural, está la reducción del número de controladores a 3. Cada controlador cumple funciones específicas:

- Student.controller: Se encarga de todas las peticiones relacionadas con los estudiantes.
- Survey.controller: Se encarga de todas las peticiones relacionadas con la encuesta, como guardar la encuesta, obtener las materias de la encuesta y obtener las configuraciones de la encuesta.
- Curriculum.controller: Se encarga de todas las peticiones relacionadas con el currículo. Por ejemplo, listar los periodos o crear una nueva malla curricular.

### 2.3.2 PLANIFICACIÓN

En este sprint se planificaron 2 historias de usuario, la **Tabla 2.5** muestra en detalle cada historia de usuario, junto con su estimación y las tareas necesarias para llevarla a cabo.

**Tabla 2.5.** División de tareas del sprint 3

| Historia de usuario | Estimación | Tareas |
|---|---|---|

| | | |
|---|---|---|
| 97. Creación de malla curricular: Como un coordinador<br>deseo crear una malla curricular por PAO<br>para configurar sus asignaturas, pisos, prerrequisitos y correquisitos. | 13 | • Crear endpoint para guardar una malla.<br>• Crear entidad periodo.<br>• Relacionar las entidades malla y periodo.<br>• Crear endpoint para exponer una malla de un periodo en específico.<br>• Crear endpoint para guardar configuraciones por periodo.<br>• Endpoint para activar periodo<br>• Modificación del servicio de encuesta para devolver solo materias disponibles<br>• Crear servicio para guardar una encuesta |
| 178. Ingreso a la encuesta estudiantes: Como coordinador<br>deseo que los estudiantes utilicen su email institucional y una contraseña<br>para ingresar a su encuesta personalizada. | - | NA |

### 2.3.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.9**.

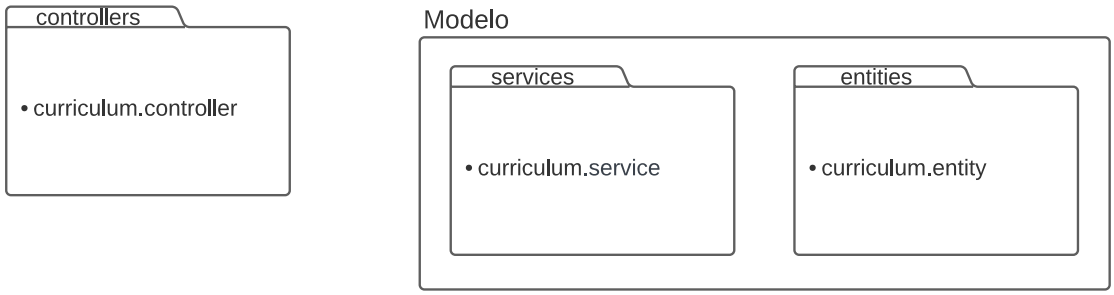



**Figura 2.9.** Controladores y modelos utilizados en el sprint 3

*2.3.3.1 Creación de malla curricular*

Para completar esta historia de usuario y tomando en cuenta la nueva estructura, se tienen que realizar dos procesos. i) Guardar la malla curricular de un periodo. ii) Devolver la malla curricular de un periodo en específico.

La **Figura 2.9** muestra la interacción del controlador *"curriculum.controller"*, y el modelo *"curriculum.entity"* a través del servicio *"curriculum.service"*, que hizo posible el proceso i.

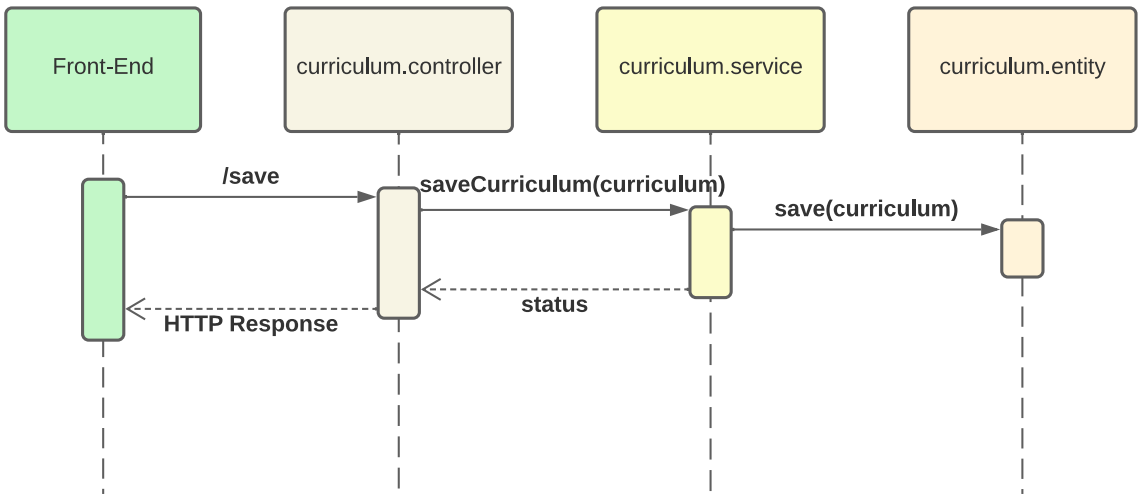



**Figura 2.10.** Guardar malla curricular

El proceso ii se logra a partir de la interacción presentada en **Figura 2.10**. En este caso, es responsabilidad del Front-End enviar el periodo y la carrera del que se requiere obtener la malla curricular.

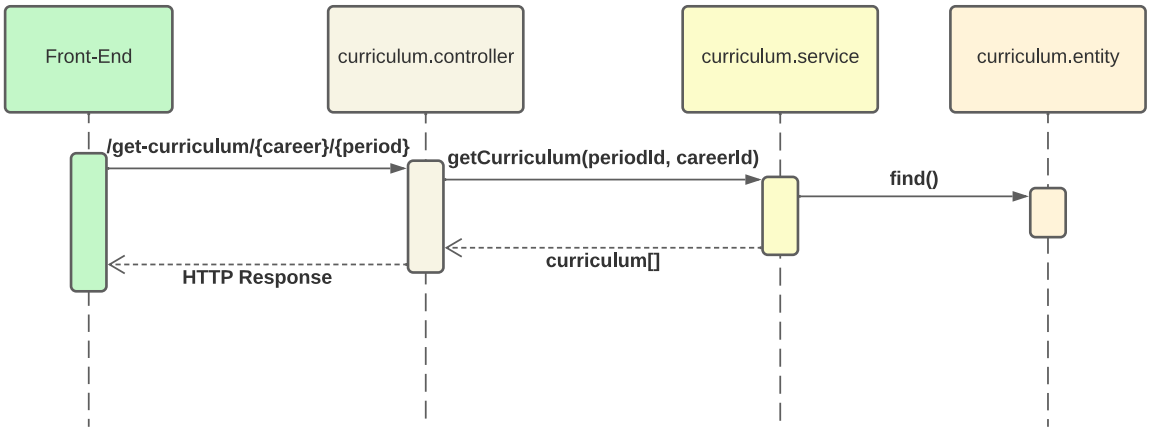



**Figura 2.11.** Obtener malla curricular

### 2.3.4 REVISIÓN

Como se puede apreciar en la **Tabla 2.6**, el esfuerzo asignado a la historia de usuario 97 fue subestimado. Esto se debió a que fue necesario realizar una reestructuración para poder cumplir con los requisitos.

**Tabla 2.6.** Comparativa de estimaciones del sprint 3

| Historia de usuario | Estimación inicial | Estimación real |
|---|---|---|
| 97. Creación de malla curricular: Como un coordinador<br>deseo crear una malla curricular por PAO<br>para configurar sus asignaturas, pisos, prerrequisitos y correquisitos. | 13 | 34 |
| 178. Ingreso a la encuesta estudiantes: Como coordinador<br>deseo que los estudiantes utilicen su email institucional y una contraseña<br>para ingresar a su encuesta personalizada. | - | - |

### 2.3.5 RETROSPECTIVA

En este sprint no se realizó la retrospectiva

## 2.4 SPRINT 4

### 2.4.1 INTRODUCCIÓN

El objetivo de este sprint es "Generar malla curricular por PAO" para corregir errores en el flujo existente. Además, se busca implementar la capacidad de configurar el mensaje de la cabecera de la encuesta. También, como parte de este sprint se trabajará en la creación de perfiles de usuario para controlar el acceso a las diferentes partes del sistema, es decir, un usuario tendrá acceso únicamente a partes del sistema que estén definidas en su perfil de acceso asignado.

### 2.4.2 PLANIFICACIÓN

En este sprint se planificaron 4 historias de usuario. Una descripción más detallada de las historias de usuario se encuentra en la **Tabla 2.7**, que proporciona una estimación para cada una de ellas, así como las tareas específicas del Back-End requeridas para su implementación.

**Tabla 2.7.** División de tareas del sprint 4

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 185. Ingreso sistemas autoridades: Como coordinador deseo que las autoridades utilicen su email institucional y contraseña para ingresar al sistema de planificación PAO. | 13 | • Crear endpoint para buscar estudiante por correo.<br>• Crear endpoint para actualizar los tokens de usuario. |
| 177. Creación de usuarios y perfiles: Como administrador deseo crear usuarios y perfiles para que tengan acceso a páginas web específicas. | 13 | • Crear modelo para roles.<br>• Crear endpoint para exponer todos los roles.<br>• Crear endpoint para modificar el rol de un usuario. |
| 179. Mensaje cabecera encuesta: Como coordinador deseo un formulario para ingresar un mensaje y enlaces a documentos externos para que los estudiantes lean el mensaje al inicio de la encuesta | 5 | • Relacionar a las configuraciones con las carreras.<br>• Crear endpoint para exponer configuraciones por carrera. |
| 205. Corrección de flujo para creación de malla curricular: Como coordinador deseo que se | 5 | • Crear endpoint para exponer asignaturas por carrera. |

| | | |
|---|---|---|
| corrija el flujo de creación de malla curricular para terminar el proceso de creación sin ninguna novedad | | • Crear endpoint para eliminar un currículo.<br>• Crear endpoint para exponer periodos usando el id. |

### 2.4.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.12**.

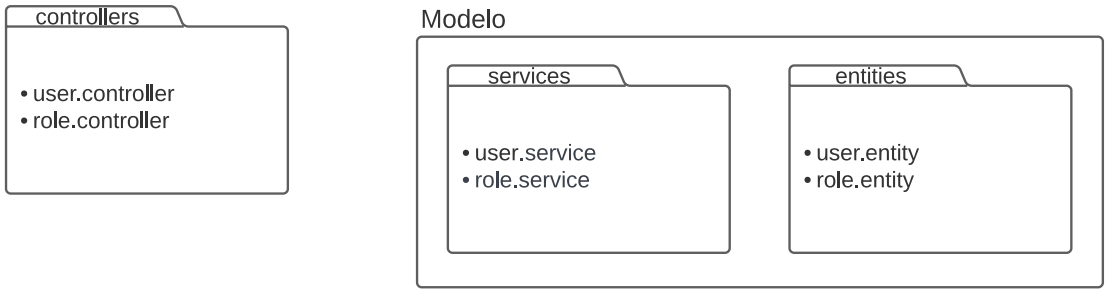



**Figura 2.12.** Controladores y modelos utilizados en el sprint 4

*2.4.3.1 Ingreso sistemas autoridades*

Hasta el momento, los estudiantes eran considerados los usuarios del sistema, cada uno identificado por un código único que permitía una búsqueda eficiente. Ahora, un nuevo usuario del sistema aparece: las autoridades. Esto implica la necesidad de buscar usuarios por un parámetro diferente, en este caso, el correo electrónico.

Para reflejar esta nueva realidad, se han realizado cambios en los nombres de las entidades, servicios y controladores. Ahora se utilizan los términos "*user*" en lugar de "*student*" para que los nombres sean más adecuados y representativos de la diversidad de usuarios en el sistema.

En la **Figura 2.13**, se muestra la interacción de estos tres componentes: entidad, servicio y controlador, para realizar la búsqueda de un usuario por su correo electrónico. Esta actualización permite una correcta identificación y gestión de los usuarios, ya sean estudiantes o autoridades, mejorando así la funcionalidad del sistema.

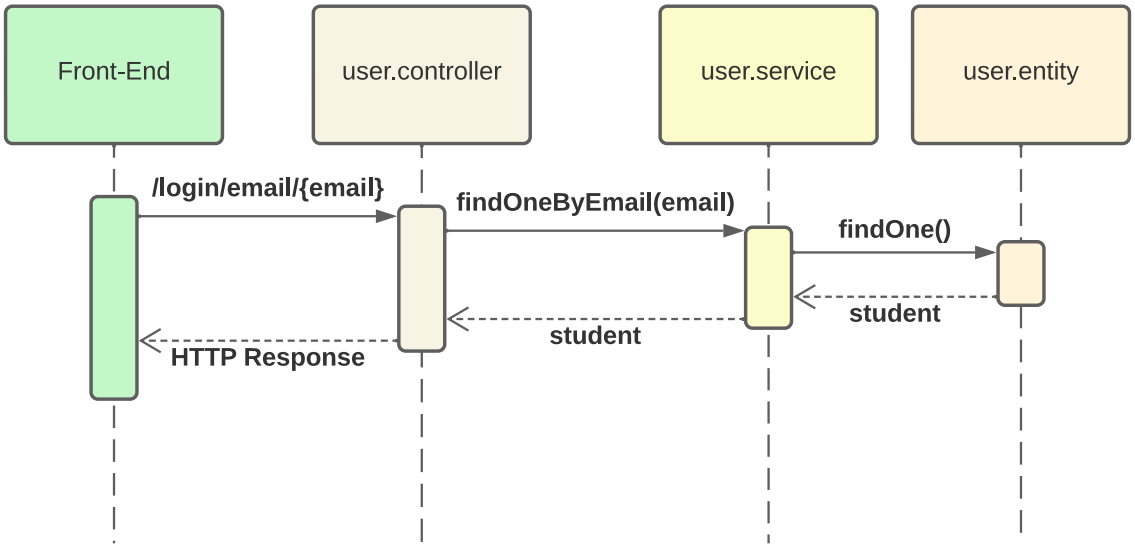



**Figura 2.13.** Buscar usuario por email

Se utilizará Firebase como servicio de autenticación de usuarios, lo que implica almacenar el token devuelto por Firebase en nuestra base de datos local. Inicialmente, un usuario no tendrá ningún token registrado. Sin embargo, cuando el usuario registre su contraseña por primera vez, se actualizará la información del usuario para incluir este token. La **Figura 2.14** muestra el proceso detallado de cómo se lleva a cabo la actualización y almacenamiento del token en nuestro sistema.

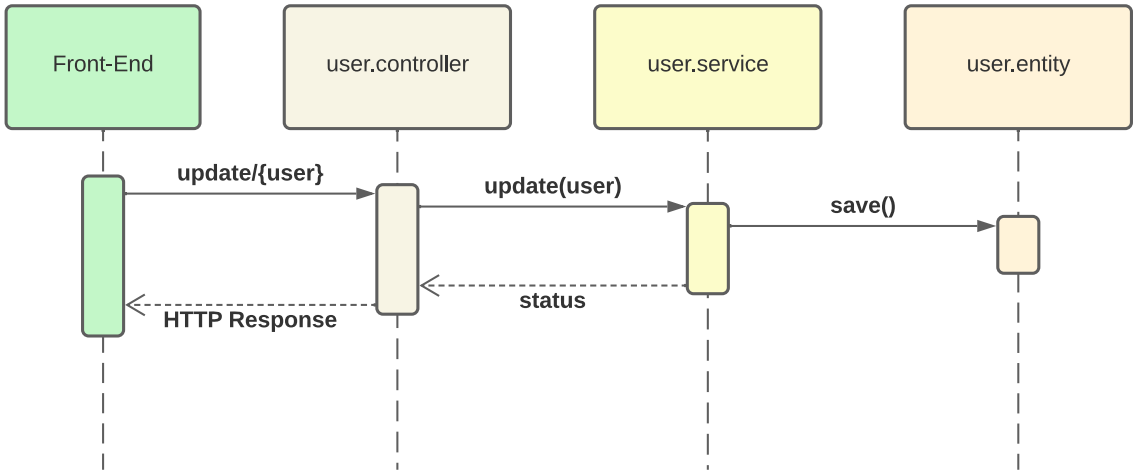



**Figura 2.14.** Actualizar información de usuario

*2.4.3.2 Creación de usuarios y perfiles*

Es fundamental asignar roles a cada usuario dentro del sistema, ya que un estudiante no debe tener la capacidad de configurar la malla curricular de un período. Con este objetivo, se ha creado el modelo "*role.entity*". Además, se ha implementado un nuevo endpoint para exponer todos los roles disponibles en el sistema, lo que permitirá asignar los roles

adecuados a cada usuario según sus funciones y responsabilidades. Este nuevo endpoint requiere de la interacción del controlador y el modelo a través de un servicio, como se muestra en la **Figura 2.15**.

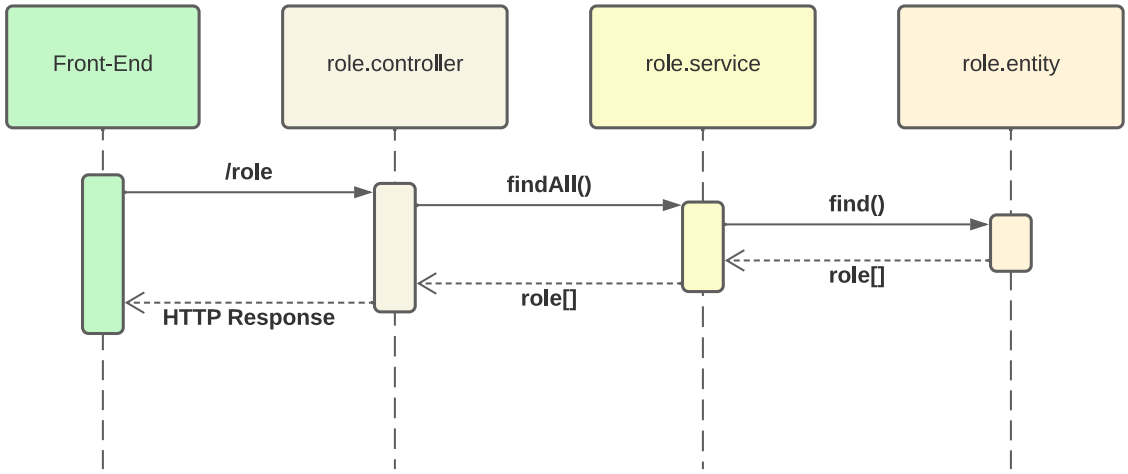



**Figura 2.15.** Obtener roles

En algunos casos, es posible que un usuario cambie su rol a lo largo del tiempo. Para abordar esta necesidad, se ha creado un nuevo endpoint que permite actualizar el rol de un usuario. En la **Figura 2.16**, se muestra el flujo de interacción entre el controlador, el servicio y el modelo correspondiente para llevar a cabo esta actualización de rol. Esta implementación garantiza la flexibilidad necesaria para adaptar los roles de los usuarios según sea necesario en el sistema.

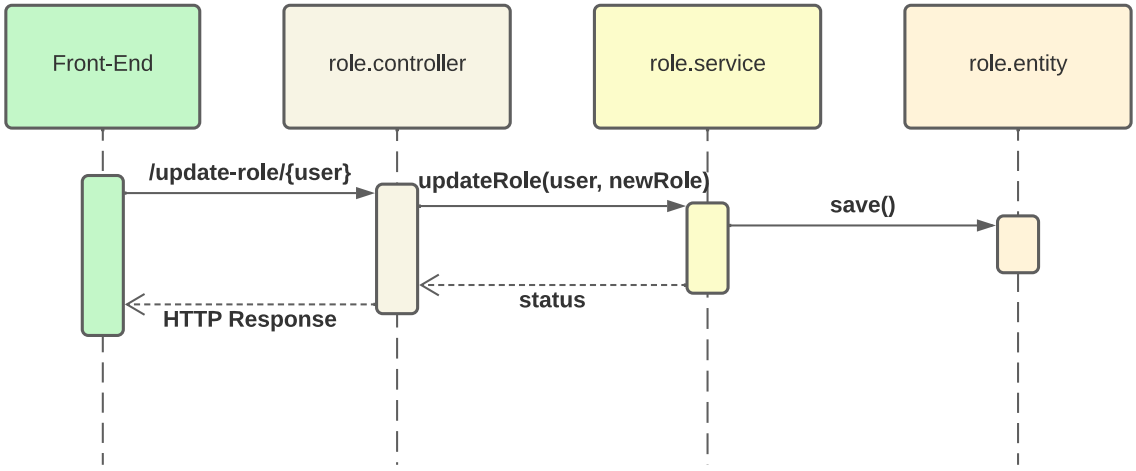



**Figura 2.16.** Actualizar rol de usuario

*2.4.3.3 Mensaje cabecera encuesta*

Actualmente es posible establecer configuraciones específicas para cada periodo, pero estas se comparten entre todas las carreras. Para permitir configuraciones específicas por

carrera, se ha implementado una relación entre las configuraciones y las carreras. Esto ofrece una mayor personalización y adaptabilidad de las configuraciones según las necesidades particulares de cada carrera en el sistema.

Una vez que se disponen de configuraciones para cada carrera, se ha creado un nuevo endpoint que recibe como parámetro la carrera y devuelve las configuraciones correspondientes exclusivamente a esa carrera. De esta manera, se facilita el acceso y la obtención de las configuraciones específicas para una carrera en particular, asegurando un manejo más eficiente y adecuado de las configuraciones según las necesidades de cada contexto académico.

#### *2.4.3.4 Corrección de flujo para creación de malla curricular*

Durante la creación de una malla curricular, es necesario seleccionar las asignaturas que se agregarán a la misma. En este proceso no es posible filtrar las asignaturas por carrera, lo que podría permitir seleccionar asignaturas de una carrera diferente a la que se está creando la malla.

Para resolver este problema, se ha implementado un nuevo endpoint que recibe la carrera, como parámetro, y devuelve las asignaturas correspondientes únicamente a dicha carrera. Con esto se garantiza que las asignaturas seleccionadas sean coherentes con la carrera en la que se está creando la malla curricular.

Por otro lado, para mejorar la eficiencia en la búsqueda de periodos, se ha creado un nuevo endpoint que permite buscarlos utilizando el identificador (ID) correspondiente. Esto facilita y agiliza el proceso de encontrar periodos específicos en el sistema.

### 2.4.4 REVISIÓN

En la **Tabla 2.8**, se presenta una comparativa entre el esfuerzo estimado al inicio del sprint y el esfuerzo real dedicado a cada historia de usuario. Se evidencia que hubo una subestimación en la primera historia de usuario, que abordaba la autenticación de los usuarios en el sistema. Esta subestimación se debió a un desconocimiento inicial por parte del equipo con relación a la tecnología utilizada. Este aprendizaje ha servido como lección para futuros proyectos y permitirá realizar estimaciones más precisas en situaciones similares.

**Tabla 2.8.** Comparativa de estimaciones del sprint 4

| Historia de usuario | Estimación inicial | Estimación real |
|---|---|---|

| | | |
|---|---|---|
| 185. Ingreso sistemas autoridades: Como coordinador<br>deseo que las autoridades utilicen su email institucional y contraseña<br>para ingresar al sistema de planificación PAO. | 13 | 34 |
| 177. Creación de usuarios y perfiles: Como administrador<br>deseo crear usuarios y perfiles<br>para que tengan acceso a páginas web específicas. | 13 | 13 |
| 179. Mensaje cabecera encuesta: Como coordinador<br>deseo un formulario para ingresar un mensaje y enlaces a documentos externos<br>para que los estudiantes lean el mensaje al inicio de la encuesta | 5 | 5 |
| 205. Corrección de flujo para creación de malla curricular: Como coordinador deseo que se corrija el flujo de creación de malla curricular para terminar el proceso de creación sin ninguna novedad | 5 | 5 |

### 2.4.5 RETROSPECTIVA

En base a la experiencia de este sprint, se destaca que la división de tareas fue una práctica positiva. Para mejorar, se sugiere enfocarse en esta área para lograr una estimación más precisa. Además, se recomienda comenzar a actualizar la wiki y el manual de usuario. Por otro lado, es importante dejar de intentar resolver los problemas de manera individual y comunicar al equipo de forma más rápida. Estas mejoras promoverán una mayor colaboración y eficiencia en el proyecto.

## 2.5 SPRINT 5

### 2.5.1 INTRODUCCIÓN

El objetivo de este sprint es “Corregir errores y mejorar experiencia de usuario”. Este objetivo surge después de la revisión del sistema realizada por el Product Owner, donde

identificó una lista de errores que incluyen problemas de funcionamiento y experiencia de usuario. Estos problemas fueron documentados en forma de historias de usuario.

### 2.5.2 PLANIFICACIÓN

En este sprint se planificaron 8 historias de usuario. La **Tabla 2.9** presenta un desglose detallado de cada historia de usuario, incluyendo su estimación y las tareas necesarias para su cumplimiento.

**Tabla 2.9.** División de tareas del sprint 5

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 254. Navegación del sistema:<br>Como un usuario del sistema<br>quiero botones y rutas de navegación<br>para comprender el mapa de navegación | 1 | NA |
| 253. Error en la sesión del usuario<br>Pasos:<br>1. Ingreso el usuario y contraseña<br>2. Clic en "Ingresar"<br>3. Navegar en el sistema<br>4. El sistema me regresa la página de ingreso | 2 | NA |
| 252. Error en correquisitos<br>1. Crear 2 correquisitos a la asignatura Ingeniería de Software: Sistemas de Información y Bases de Datos.<br>2. Crear 1 correquisito a la asignatura Bases de Datos: Ingeniería de Software.<br>3. Crear 1 correquisito a la asignatura Sistemas de | 5 | NA |

| | | |
|---|---|---|
| Información: Ingeniería de Software.<br>4. En una encuesta seleccionar las tres asignaturas de 4 nivel de software: Ingeniería de Software, Sistemas de Información y Bases de Datos.<br>5. Intentar enviar la encuesta. | | |
| 183. Créditos matrículas: Como coordinador<br>deseo un formulario de configuración de créditos mínimos y máximos para matrículas<br>para que los estudiantes reciban un mensaje en caso de pérdida de la gratuidad o ampliación de créditos. | 8 | • Modificar atributos de configuraciones para agregar un mensaje.<br>• Al crear un periodo, crear todas las configuraciones asociadas a los créditos. |
| 182. Inicio y fin de encuesta: Como coordinador<br>deseo un formulario para configurar fecha de inicio y finalización de la encuesta<br>para que los estudiantes tengan un mensaje inicio y finalización de la encuesta | 5 | • Al crear un periodo, crear todas las configuraciones asociadas a las fechas de envío. |
| 250. Error al iniciar sesión<br>1. Ingreso usuario y contraseña<br>2. Clic en el botón "Ingresar"<br>3. Autenticación satisfactoria | 2 | NA |

| | | |
|---|---|---|
| 4. Se muestra una página en blanco con un mensaje de error | | |
| 246. Contador de créditos: Como coordinador<br>deseo que el estudiante visualice el número de créditos seleccionados en la encuesta | 3 | NA |

### 2.5.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.17**.

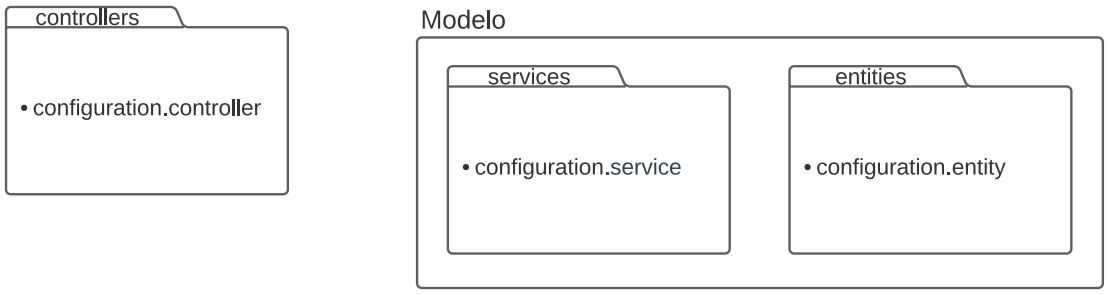



**Figura 2.17.** Controladores y modelos utilizados en el sprint 5

*2.5.3.1 Créditos matrículas*

Hasta el momento de este Sprint, los atributos de "*configuration.entity*" eran: *id, key, value*, *career_id* y *period_id*. En este Sprint, se agrega el atributo *message* a la lista de atributos con el fin de mostrar un mensaje al usuario, al momento de que una configuración determinada no se haya cumplido. Con esto, cuando en una encuesta no se cumpla con los créditos máximos o mínimos, el mensaje registrado en el atributo *message* que corresponde con dicha configuración será mostrado.

Además, es importante asegurar que todos los periodos cuenten con configuraciones de créditos máximos y mínimos. Para esto, se implementa la creación automática de dichas configuraciones con valores por defecto al momento de crear un nuevo periodo.

*2.5.3.2 Inicio y fin de la encuesta*

Tanto las fechas de inicio como de fin de la encuesta son consideradas configuraciones y, por lo tanto, se almacenan de manera similar a los créditos mínimos y máximos. Para

garantizar que todos los periodos tengan estas configuraciones establecidas, se implementa la lógica de negocio de crear automáticamente las configuraciones al momento de crear un nuevo periodo. Esto asegura que cada periodo tenga definidas adecuadamente las fechas de inicio y fin correspondientes a la encuesta.

### 2.5.4 REVISIÓN

La **Tabla 2.10** muestra una comparativa de la estimación inicial y la estimación real para cada una de las historias. En este caso, las dos estimaciones son iguales, esto refleja que el equipo ha ganado experiencia tanto con la tecnología utilizada como en el dominio del negocio.

**Tabla 2.10.** Comparativa de estimaciones del sprint 5

| **Historia de usuario** | **Estimación inicial** | **Estimación real** |
|---|---|---|
| 254. Navegación del sistema: Como un usuario del sistema<br>quiero botones y rutas de navegación<br>para comprender el mapa de navegación | 1 | 1 |
| 253. Error en la sesión del usuario<br>Pasos:<br>5. Ingreso el usuario y contraseña<br>6. Clic en "Ingresar"<br>7. Navegar en el sistema<br>8. El sistema me regresa la página de ingreso | 2 | 2 |
| 252. Error en correquisitos<br>6. Crear 2 correquisitos a la asignatura Ingeniería de Software: Sistemas de Información y Bases de Datos.<br>7. Crear 1 correquisito a la asignatura Bases de Datos: Ingeniería de Software.<br>8. Crear 1 correquisito a la asignatura Sistemas de Información: Ingeniería de Software.<br>9. En una encuesta seleccionar las tres asignaturas de 4 nivel de software: | 5 | 5 |

| | | |
|---|---|---|
| Ingeniería de Software, Sistemas de Información y Bases de Datos.<br>10. Intentar enviar la encuesta. | | |
| 183. Créditos matrículas: Como coordinador deseo un formulario de configuración de créditos mínimos y máximos para matrículas para que los estudiantes reciban un mensaje en caso de pérdida de la gratuidad o ampliación de créditos. | 8 | 8 |
| 182. Inicio y fin de encuesta: Como coordinador deseo un formulario para configurar fecha de inicio y finalización de la encuesta<br>para que los estudiantes tengan un mensaje inicio y finalización de la encuesta | 5 | 5 |
| 250. Error al iniciar sesión<br>5. Ingreso usuario y contraseña<br>6. Clic en el botón "Ingresar"<br>7. Autenticación satisfactoria<br>8. Se muestra una página en blanco con un mensaje de error | 2 | 2 |
| 246. Contador de créditos: Como coordinador deseo que el estudiante visualice el número de créditos seleccionados en la encuesta | 3 | 3 |

### 2.5.5 RETROSPECTIVA

En la retrospectiva del sprint, se identificaron varias acciones para mejora del equipo. Se acordó comenzar a utilizar un diseño centrado en el usuario y realizar pruebas de estrés. Además, se propuso dejar de buscar culpables, llegar tarde a los dailys, resolver problemas individualmente y dejar las pruebas para el último. Se sugirió hacer diferente el desarrollo, comenzando con interfaces de usuario definidas y comunicar los avances a todo el equipo de desarrollo. Finalmente, se recomendó reconocer y felicitar el éxito alcanzado, además de fomentar una comunicación más efectiva en el equipo.

## 2.6 SPRINT 6

### 2.6.1 INTRODUCCIÓN

El objetivo del sprint es implementar la funcionalidad de "Carga de currículum académico de estudiantes". Esto permitirá importar datos personales y detalles del currículum académico de cada estudiante a través de archivos CSV. Además, se tienen en cuenta las historias de usuario para mejorar la experiencia del usuario en el sistema.

### 2.6.2 PLANIFICACIÓN

En este sprint se planificaron 9 historias de usuario, la **Tabla** muestra en detalle cada historia de usuario, junto con su estimación y las tareas necesarias para llevarla a cabo.

**Tabla 2.11.** División de tareas del sprint 6

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 301. Formato de las interfaces: Como un usuario del sistema quiero que los botones en todas las interfaces se encuentren bien posicionados para que el sistema mantenga concordancia entre sí. | 5 | NA |
| 284. Cargar datos de estudiantes matriculados: Como coordinador deseo cargar los datos de los estudiantes matriculados desde un archivo pare facilitar el manejo de aproximadamente mil estudiantes. | 8 | • Crear endpoint para recibir archivo CSV.<br>• Crear servicio para crear usuarios a partir de un archivo CSV.<br>• Validación de la cabecera del archivo CSV. |
| 254. Navegación del sistema: Como un usuario del sistema quiero botones y rutas de navegación para comprender el mapa de navegación | 1 | NA |
| 168. Copia malla curricular: Como un coordinador deseo copiar una malla curricular de un PAO anterior | 13 | • Crear endpoint para copiar malla<br>• Crear servicio para copiar malla |

| | | |
|---|---|---|
| para copiar asignaturas, pisos, prerrequisitos y correquisitos en el próximo PAO. | | |
| 289. Mensaje de créditos: Como coordinador<br>deseo que los mensajes sobre el número de créditos se encuentren en la parte inferior de la encuesta y exista una confirmación<br>para obligar al estudiante a leer y aceptar el mensaje antes de enviar la encuesta | 1 | NA |
| 252. Error en los correquisitos:<br>1. Crear 2 correquisitos a la asignatura Ingeniería de Software: Sistemas de Información y Bases de Datos<br>2. Crear 1 correquisito a la asignatura Bases de Datos: Ingeniería de Software<br>3. Crear 1 correquisito a la asignatura Sistemas de Información: Ingeniería de Software<br>4. En una encuesta seleccionar las tres asignaturas de 4 nivel de software: Ingeniería de Software, Sistemas de Información y Bases de Datos<br>5. Intentar enviar la encuesta | 5 | NA |

| | | |
|---|---|---|
| 283. Previsualizar la encuesta: Como coordinador<br>deseo previsualizar la encuesta<br>para verificar sus datos y validaciones. | 8 | • Crear servicio que exponga todas las materias de todas las carreras del periodo activo. |
| 286. Tipos de mensajes: Como coordinador<br>deseo que existan diferentes tipos de mensajes<br>para que el estudiante entiende cada tipo de mensaje | 3 | NA |
| 285. Cargar datos de currículums académicos: Como coordinador<br>deseo cargar los datos de los currículums académicos de los estudiantes matriculados desde un archivo<br>pare facilitar el manejo de aproximadamente mil currículums académicos. | 13 | • Crear endpoint para recibir archivo CSV.<br>• Crear servicio para crear currículums académicos a partir de un archivo CSV.<br>• Validación de la cabecera del archivo CSV. |

### 2.6.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario, considerando la participación de los elementos en la **Figura 2.18**.

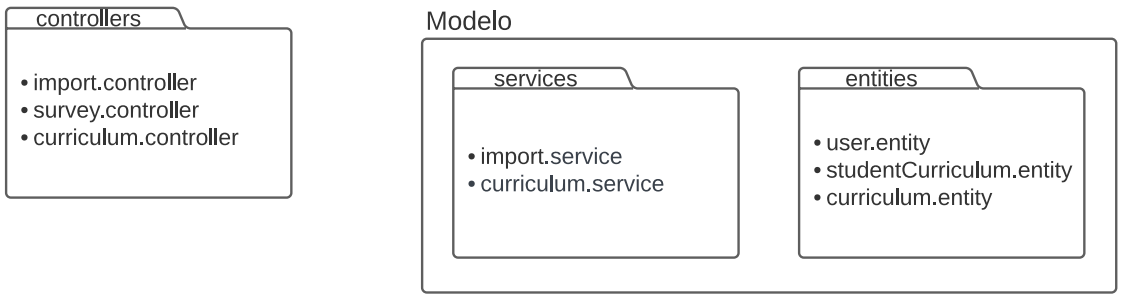



**Figura 2.18.** Controladores y modelos utilizados en el sprint 6

*2.6.3.1 Cargar datos de estudiantes matriculados*

Se requiere la importación de datos de estudiantes matriculados en el sistema. Cada estudiante debe contar con cuatro atributos obligatorios: i) un código único, ii) nombres completos, iii) correo electrónico y iv) carrera. Por lo tanto, para realizar la importación de estudiantes, es necesario que el archivo CSV contenga esta información para cada estudiante. La **Figura 2.19** muestra la plantilla que se utiliza para este propósito.

| code | fullName | email | careerCode |
|---|---|---|---|
| 201810202 | CARLOS JULIO BAYAS CHAVES | carlos.bayas@epn.edu.ec | Software |
| 202020413 | SANTIAGO ANDRES BEJARANO JIMENEZ | santiago.bejarano@epn.edu.ec | Software |
| 202020966 | BRYAN ALEJANDRO GUANO LUPERA | bryan.guano@epn.edu.ec | Software |

**Figura 2.19.** Plantilla de importación de estudiantes

Se puede observar el proceso de importación de estudiantes en la **Figura 2.20**. Este proceso inicia con la recepción del archivo CSV por parte del controlador, se procesa el archivo en el servicio para finalmente escribir los datos en la entidad.

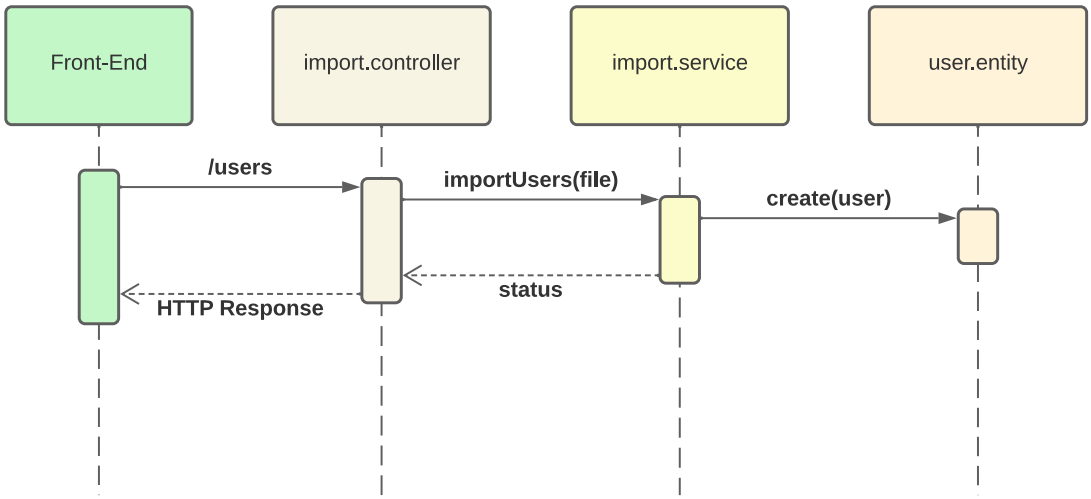



**Figura 2.20.** Proceso de importación de estudiantes

*2.6.3.2 Copia malla curricular*

Debido a que los cambios entre mallas curriculares suelen ser mínimos, se requiere que se pueda copiar una malla curricular de un periodo anterior en un periodo nuevo. Esto se logra enviando al controlador los datos del nuevo periodo (código y nombre) y el período del cual se quiere copiar la malla curricular como se muestra en la **Figura 2.21**.

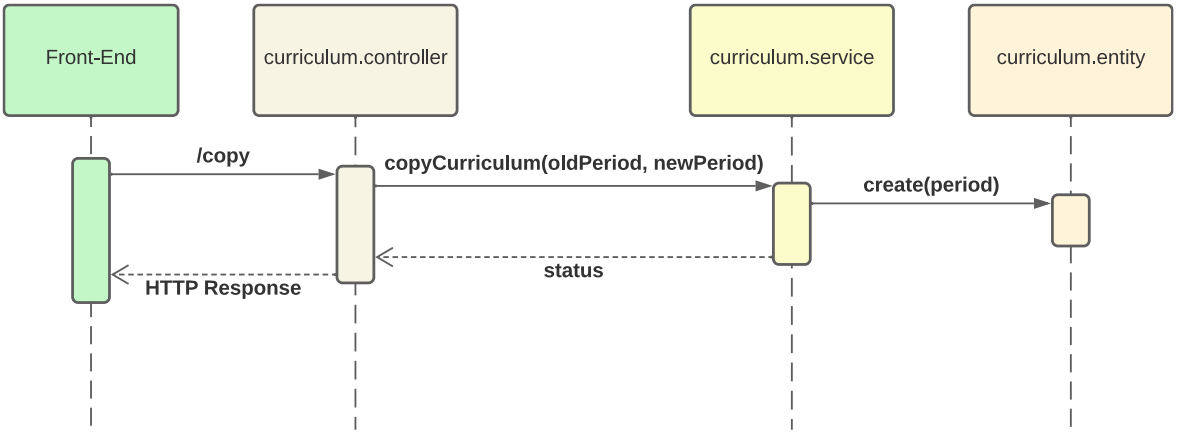



**Figura 2.21.** Proceso para copiar una malla curricular

*2.6.3.3 Previsualizar la encuesta*

Una autoridad debe tener la capacidad de previsualizar una encuesta antes de enviarla a los estudiantes. La finalidad de esta previsualización es verificar que las configuraciones y validaciones de la encuesta sean correctas. Para lograr esto, es necesario mostrar todas las asignaturas de todas las carreras del periodo activo.

El proceso descrito en la **Figura 2.22** ilustra la interacción que se da entre el controlador y el modelo para exponer la data necesaria.

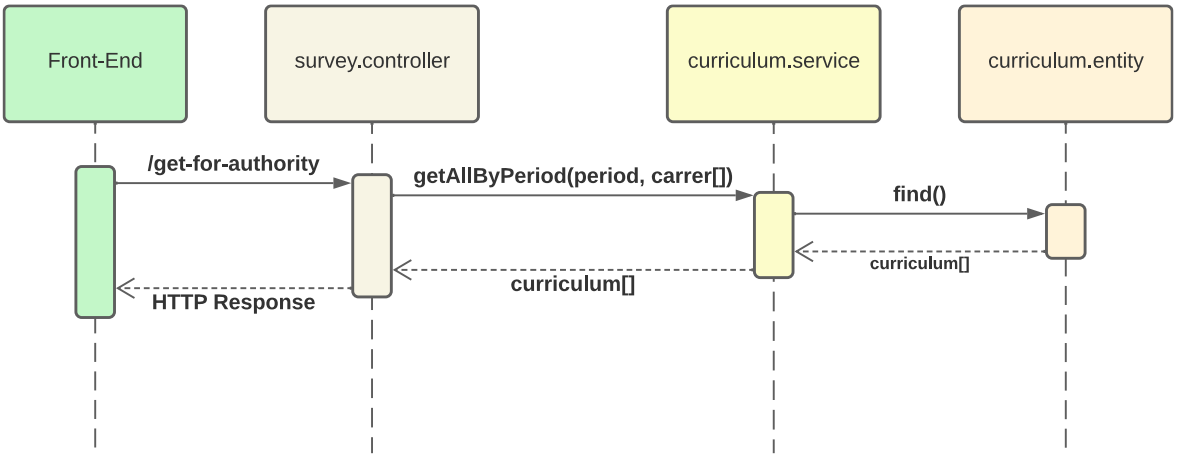



**Figura 2.22.** Proceso para previsualizar la encuesta

*2.6.3.4 Cargar datos de currículums académicos*

Se requiere la importación de currículums académicos de los estudiantes existentes en el sistema. Para lograr esta importación, el archivo CSV que se espera recibir tiene el formato mostrado en la **Figura 2.23**. Los campos necesarios son: i) código único del estudiante, ii) código de la asignatura, iii) código de la carrera y iv) estado de la asignatura (Aprobada, Cursando, Pendiente).

| studentCode | subjectCode | careerCode | state |
|---|---|---|---|
| 202011247 | ISWD913 | SW | Pendiente |
| 202011247 | ISWD922 | SW | Pendiente |
| 202011247 | PRLD105 | SW | Pendiente |

**Figura 2.23.** Plantilla de importación de currículum académico

Se puede observar este proceso de importación en la **Figura 2.24**. Se inicia con la recepción del archivo CSV por parte del controlador, se procesa el archivo en el servicio para finalmente escribir los datos en la entidad.

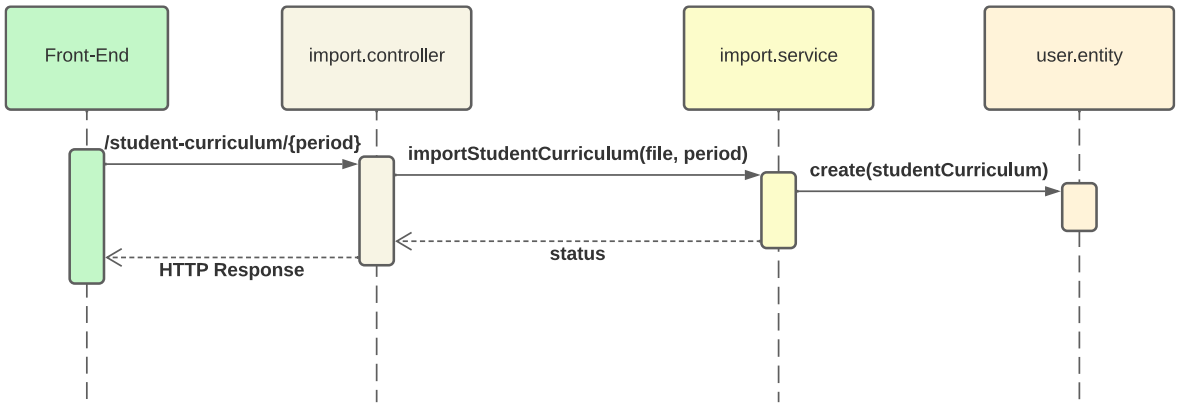



**Figura 2.24.** Proceso de importación de currículums académicos

### 2.6.4 REVISIÓN

La **Tabla** presenta una comparativa entre la estimación inicial y la estimación real para cada una de las historias. Se puede observar una tendencia a sobrestimar el esfuerzo requerido. Esta situación puede ser resultado de la subestimación del esfuerzo en sprints anteriores, lo cual indica que el equipo aún no cuenta con la experiencia suficiente en este aspecto.

**Tabla 2.12.** Comparativa de estimaciones del sprint 6

| Historia de usuario | Estimación inicial | Estimación real |
|---|---|---|
| 301. Formato de las interfaces: Como un usuario del sistema<br>quiero que los botones en todas las interfaces se encuentren bien posicionados<br>para que el sistema mantenga concordancia entre sí. | 5 | 5 |

| | | |
|---|---|---|
| 284. Cargar datos de estudiantes matriculados: Como coordinador<br>deseo cargar los datos de los estudiantes matriculados desde un archivo<br>pare facilitar el manejo de aproximadamente mil estudiantes. | 8 | 8 |
| 254. Navegación del sistema: Como un usuario del sistema<br>quiero botones y rutas de navegación<br>para comprender el mapa de navegación | 1 | 1 |
| 168. Copia malla curricular: Como un coordinador<br>deseo copiar una malla curricular de un PAO anterior<br>para copiar asignaturas, pisos, prerrequisitos y correquisitos en el próximo PAO. | 13 | 8 |
| 289. Mensaje de créditos: Como coordinador<br>deseo que los mensajes sobre el número de créditos se encuentren en la parte inferior de la encuesta y exista una confirmación<br>para obligar al estudiante a leer y aceptar el mensaje antes de enviar la encuesta | 1 | 1 |
| 252. Error en los correquisitos:<br>6. Crear 2 correquisitos a la asignatura Ingeniería de Software: Sistemas de Información y Bases de Datos<br>7. Crear 1 correquisito a la asignatura Bases de Datos: Ingeniería de Software<br>8. Crear 1 correquisito a la asignatura Sistemas de Información: Ingeniería de Software<br>9. En una encuesta seleccionar las tres asignaturas de 4 nivel de software: Ingeniería de Software, Sistemas de Información y Bases de Datos | 5 | 5 |

| | | |
|---|---|---|
| Intentar enviar la encuesta | | |
| 283. Previsualizar la encuesta: Como coordinador<br>deseo previsualizar la encuesta<br>para verificar sus datos y validaciones. | 8 | 8 |
| 286. Tipos de mensajes: Como coordinador<br>deseo que existan diferentes tipos de mensajes<br>para que el estudiante entiende cada tipo de mensaje | 3 | 3 |
| 285. Cargar datos de currículums académicos: Como coordinador<br>deseo cargar los datos de los currículums académicos de los estudiantes matriculados desde un archivo<br>para facilitar el manejo de aproximadamente mil currículums académicos. | 13 | 8 |

### 2.6.5 RETROSPECTIVA

En la retrospectiva del sprint, se identificaron varias acciones para mejora del equipo. Se acordó comenzar a actualizar el manual de usuario cuando se complete una característica. Además, se propuso dejar de actualizar la wiki al final del sprint. Se sugirió hacer diferente la planificación del sprint al realizar actividades de grooming. Finalmente, se recomendó mantener la velocidad constante de desarrollo.

## 2.7 SPRINT 7

### 2.7.1 INTRODUCCIÓN

El objetivo del sprint es implementar la funcionalidad de "Generar reportes personalizados". Esto permitirá presentar la información a las autoridades de manera resumida y comprensible, facilitando la toma de decisiones para el próximo PAO.

Además, se incluye una historia de usuario para abordar una brecha de seguridad mediante la implementación de tokens de acceso para consumir los endpoints expuestos. Esto mejorará la seguridad del sistema y garantizará un acceso controlado a la información.

### 2.7.2 PLANIFICACIÓN

En este sprint se planificaron 7 historias de usuario, la **Tabla** muestra en detalle cada historia de usuario, junto con su estimación y las tareas necesarias para llevarla a cabo.

**Tabla 2.13.** División de tareas del sprint 7

| **Historia de usuario** | **Estimación** | **Tareas** |
|---|---|---|
| 245. Token de seguridad: Como desarrollador deseo que los endpoints tengan un token de acceso para impedir accesos no autorizados a la información. | 21 | • Utilizar Guards y tokens de Google para autorizar el acceso.<br>• Utilizar AES para crear tokens para casos como el registro de usuarios. |
| 49. Reporte de estudiantes que no llenan la encuesta: Como coordinador deseo un reporte de los estudiantes por carrera que aún no llenan su encuesta para obtener sus nombres, carrera y correo electrónico. | 5 | • Generar tablero para reportes y crear enlace para que sea incrustado en el Front-End.<br>• Usar Power BI para generar reporte |
| 323. Reporte general gráfico de encuestas: Como autoridad deseo un reporte gráfico de las encuestas para planificar número total de cupos por asignatura y el total de posible 2da matrícula para el próximo PAO. | 5 | • Usar Power BI para generar reporte |
| 326. Botón de confirmación para envío de encuesta: Como administrador deseo que se visualice un checkbox en la encuesta para que los estudiantes lean y acepten las condiciones | 5 | NA |
| 251. Reporte general de encuestas: Como una autoridad deseo un reporte de las encuestas para planificar número de grupos por asignaturas y total | 5 | • Usar Power BI para generar reporte |

| | | |
|---|---|---|
| de posible 2da matrícula para el próximo PAO. | | |
| 322. Consultar datos importados: Como coordinador deseo visualizar/consultar los datos de los estudiantes y sus currículums académicos importados desde archivos CSV para verificar los datos importados. | 5 | • Usar Power BI para generar reporte |
| 327. Recargar previsualización de la encuesta: Como autoridad deseo que se vuelva a inicializar el estado de la previsualización de la encuesta luego de hacer clic en enviar. | 3 | NA |

### 2.7.3 IMPLEMENTACIÓN

A continuación, se describe en detalle la implementación de cada historia de usuario.

#### *2.7.3.1 Token de seguridad*

Con el fin de mejorar la seguridad del sistema, se ha implementado el uso de tokens de acceso. Estos tokens se dividen en dos tipos aceptados:

- Token de usuario: Este tipo de token se obtiene al registrar al usuario en el sistema. Proporciona una autenticación para validar la identidad del usuario y asegurar el acceso autorizado.
- Frase cifrada con AES256: Se dispone de un segundo tipo de token para el caso del registro de un usuario. El token se genera mediante el cifrado de una frase secreta utilizando el algoritmo de cifrado AES256. La clave privada necesaria para realizar el cifrado solo es conocida por el Back-End y el Front-End del sistema, lo que garantiza que solo ellos puedan generar y verificar este tipo de token.

Al implementar estos tokens de acceso, se refuerza la seguridad del sistema al permitir una autenticación robusta y proteger el acceso a los endpoints expuestos. La **Figura 2.25** describe el flujo que siguen todas las peticiones HTTP que se realicen al sistema.

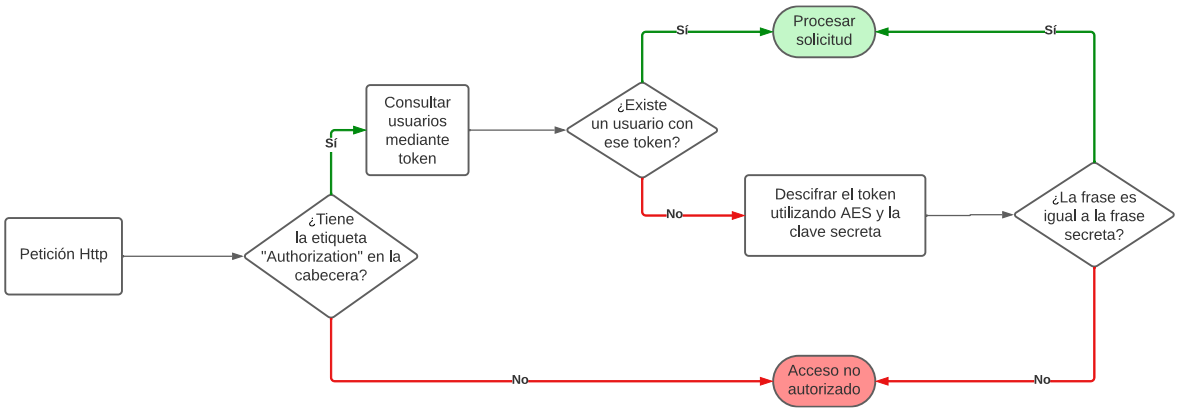
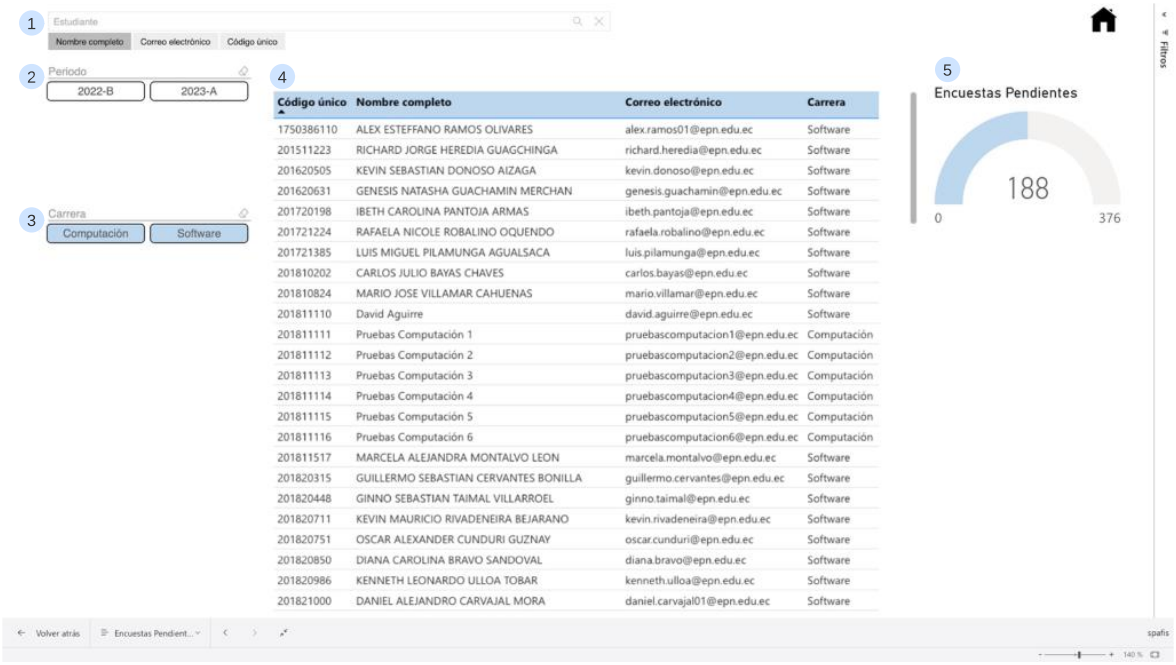



**Figura 2.25.** Flujo de autorización de peticiones HTTP

### *2.7.3.2 Reporte de estudiantes que no llenan la encuesta*

Se generó el reporte de la **Figura 2.26**, utilizando Power BI. El reporte está formado por las siguientes partes:

1. Barra de búsqueda para filtrar a los estudiantes por su nombre completo, correo electrónico o código único.
2. Periodo en el cual se quieren buscar las encuestas pendientes.
3. Carrera en la cual se quieren buscar las encuestas pendientes.
4. Información de los estudiantes que tienen encuestas pendientes.
5. Cantidad de encuestas pendientes versus el total de encuestas.




| Código único | Nombre completo | Correo electrónico | Carrera |
|---|---|---|---|
| 1750386110 | ALEX ESTEFFANO RAMOS OLIVARES | alex.ramos01@epn.edu.ec | Software |
| 201511223 | RICHARD JORGE HEREDIA GUAGCHINGA | richard.heredia@epn.edu.ec | Software |
| 201620505 | KEVIN SEBASTIAN DONOSO AIZAGA | kevin.donoso@epn.edu.ec | Software |
| 201620631 | GENESIS NATASHA GUACHAMIN MERCHAN | genesis.guachamin@epn.edu.ec | Software |
| 201720198 | IBETH CAROLINA PANTOJA ARMAS | ibeth.pantoja@epn.edu.ec | Software |
| 201721224 | RAFAELA NICOLE ROBALINO OQUENDO | rafaela.robalino@epn.edu.ec | Software |
| 201721385 | LUIS MIGUEL PILAMUNGA AGUALSACA | luis.pilamunga@epn.edu.ec | Software |
| 201810202 | CARLOS JULIO BAYAS CHAVES | carlos.bayas@epn.edu.ec | Software |
| 201810824 | MARIO JOSE VILLAMAR CAHUENAS | mario.villamar@epn.edu.ec | Software |
| 201811110 | David Aguirre | david.aguirre@epn.edu.ec | Software |
| 201811111 | Pruebas Computación 1 | pruebascomputacion1@epn.edu.ec | Computación |
| 201811112 | Pruebas Computación 2 | pruebascomputacion2@epn.edu.ec | Computación |
| 201811113 | Pruebas Computación 3 | pruebascomputacion3@epn.edu.ec | Computación |
| 201811114 | Pruebas Computación 4 | pruebascomputacion4@epn.edu.ec | Computación |
| 201811115 | Pruebas Computación 5 | pruebascomputacion5@epn.edu.ec | Computación |
| 201811116 | Pruebas Computación 6 | pruebascomputacion6@epn.edu.ec | Computación |
| 201811517 | MARCELA ALEJANDRA MONTALVO LEON | marcela.montalvo@epn.edu.ec | Software |
| 201820315 | GUILLERMO SEBASTIAN CERVANTES BONILLA | guillermo.cervantes@epn.edu.ec | Software |
| 201820448 | GINNO SEBASTIAN TAIMAL VILLARROEL | ginno.taimal@epn.edu.ec | Software |
| 201820711 | KEVIN MAURICIO RIVADENEIRA BEJARANO | kevin.rivadeneira@epn.edu.ec | Software |
| 201820751 | OSCAR ALEXANDER CUNDURI GUZNAY | oscar.cunduri@epn.edu.ec | Software |
| 201820850 | DIANA CAROLINA BRAVO SANDOVAL | diana.bravo@epn.edu.ec | Software |
| 201820986 | KENNETH LEONARDO ULLOA TOBAR | kenneth.ulloa@epn.edu.ec | Software |
| 201821000 | DANIEL ALEJANDRO CARVAJAL MORA | daniel.carvajal01@epn.edu.ec | Software |

**Figura 2.26.** Reporte de encuestas pendientes

*2.7.3.3 Reporte general de encuestas*

Se generó el reporte de la **Figura 2.27**, utilizando Power BI. El reporte está formado por las siguientes partes:

1. Barra de búsqueda para filtrar a los estudiantes por su nombre completo, correo electrónico o código único.
2. Periodo del cual se quiere ver las respuestas de las encuestas.
3. Carrera de la cual se quieren ver las respuestas de las encuestas. Además, se puede filtrar por semestre de una carrera específica.
4. Número de matrícula, para ver la cantidad de estudiantes que piensan tomar una asignatura por primera o segunda ocasión.
5. Tabla dinámica con la cantidad de respuestas que tuvo una asignatura en particular.
6. Gráfico de barras con la cantidad de respuestas que tuvo una asignatura en particular.
7. Cantidad de encuestas contestadas versus el total de encuestas.
8. Cantidad de estudiantes que piensan hacer segunda matrícula en alguna asignatura.

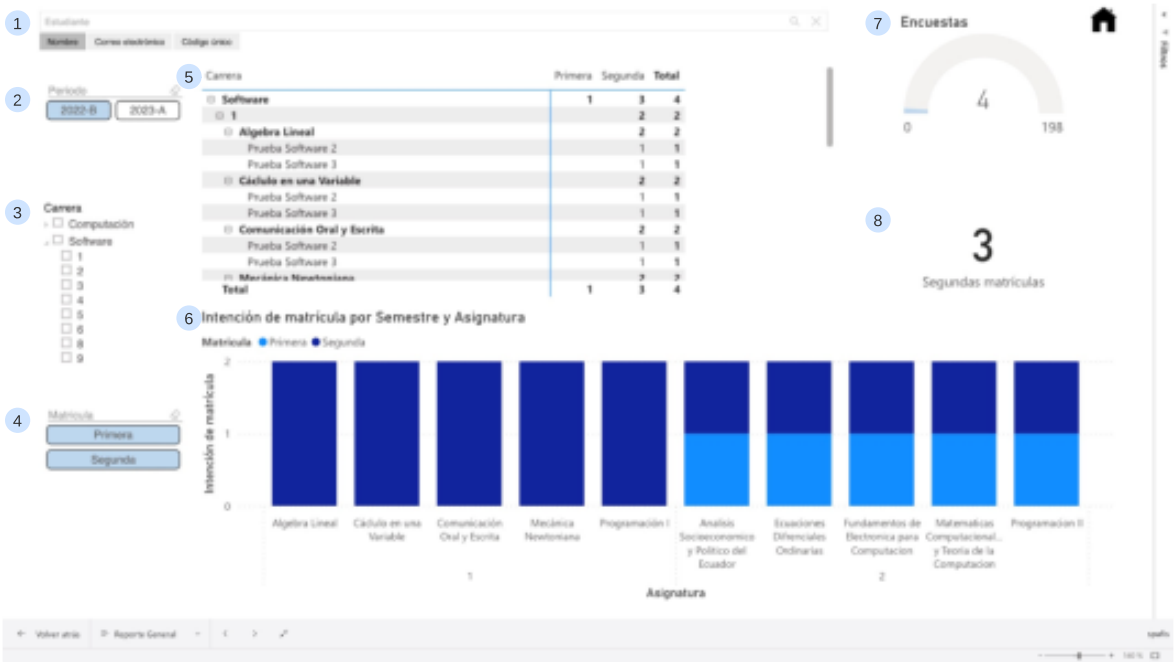



**Figura 2.27.** Reporte general de encuestas

*2.7.3.4 Reporte general gráfico de encuestas*

Este punto fue cubierto con la sección 6 de la **Figura 2.27**. En esta sección se puede ver un gráfico de barras con las respuestas de los estudiantes. También, se observa claramente las respuestas que son de primera y segunda matrícula.

*2.7.3.5 Consultar datos importados*

Se generó el reporte de la **Figura 2.28**, utilizando Power BI. El reporte está formado por las siguientes partes:

1. Barra de búsqueda para filtrar a los estudiantes por su nombre completo, correo electrónico o código único.
2. Periodo del que se quieren ver los datos importados.
3. Estado de las asignaturas que se quieren observar.
4. Estudiantes encontrados para los filtros utilizados.
5. Curriculum académico del estudiante seleccionado en la sección 4.
6. Porcentaje de asignaturas aprobadas del estudiante seleccionado en la sección 4.
7. Cantidad de asignaturas aprobadas del estudiante seleccionado en la sección 4.
8. Cantidad de asignaturas en curso del estudiante seleccionado en la sección 4.
9. Cantidad de asignaturas pendientes del estudiante seleccionado en la sección 4.

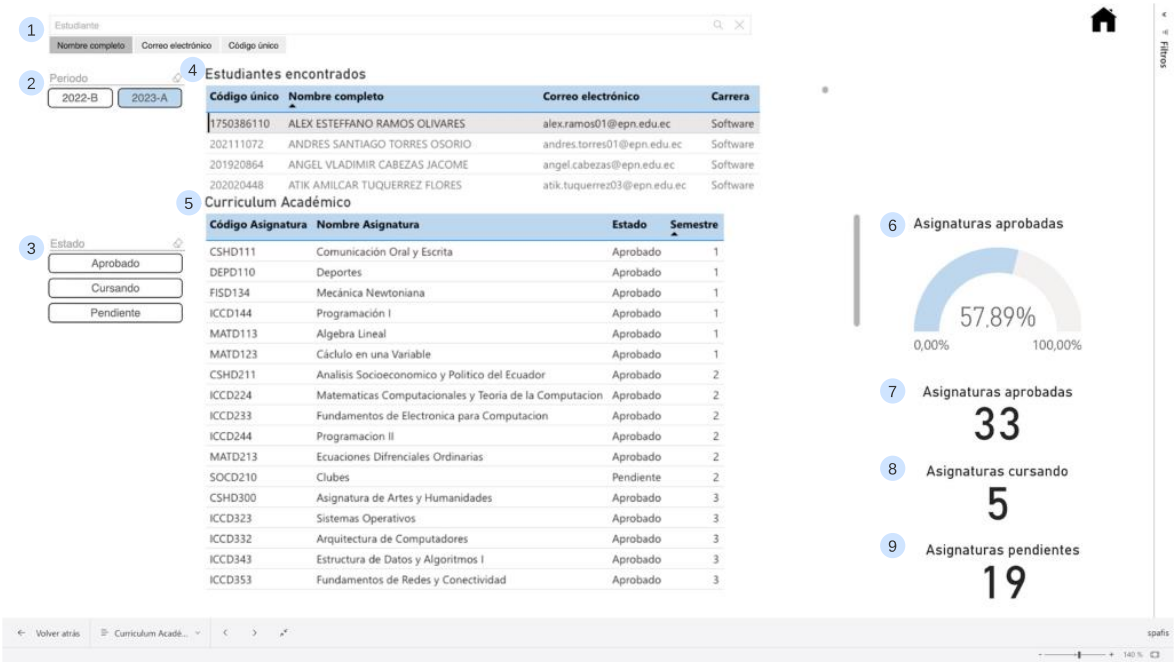



**Figura 2.28.** Reporte de datos importados

### 2.7.4 REVISIÓN

En la **Tabla** se presenta una comparativa entre la estimación inicial y la estimación real para cada historia. Durante este sprint, las estimaciones fueron adecuadas en general. Sin embargo, se identificó que hubo una gran cantidad de trabajo en la historia de usuario relacionada con el token de seguridad. Esta tarea debería haberse abordado desde el inicio del proyecto para minimizar su impacto al final. Es importante considerar la priorización y planificación adecuada de las tareas para garantizar una distribución equilibrada del trabajo a lo largo del proyecto.

**Tabla 2.14.** Comparativa de estimaciones del sprint 7

| Historia de usuario | Estimación inicial | Estimación real |
|---|---|---|
| 245. Token de seguridad: Como desarrollador deseo que los endpoints tengan un token de acceso para impedir accesos no autorizados a la información. | 21 | 21 |
| 49. Reporte estudiantes que no llenan la encuesta: Como coordinador deseo un reporte de los estudiantes por carrera que aún no llenan su encuesta | 5 | 5 |

| | | |
|---|---|---|
| para obtener sus nombres, carrera y correo electrónico. | | |
| 323. Reporte general gráfico de encuestas: Como autoridad deseo un reporte gráfico de las encuestas para planificar número total de cupos por asignatura y el total de posible 2da matrícula para el próximo PAO. | 5 | 5 |
| 326. Botón de confirmación para envío de encuesta: Como administrador deseo que se visualice un checkbox en la encuesta para que los estudiantes lean y acepten las condiciones | 5 | 5 |
| 251. Reporte general de encuestas: Como una autoridad deseo un reporte de las encuestas para planificar número de grupos por asignaturas y total de posible 2da matrícula para el próximo PAO. | 5 | 5 |
| 322. Consultar datos importados: Como coordinador deseo visualizar/consultar los datos de los estudiantes y sus currículums académicos importados desde archivos CSV para verificar los datos importados. | 5 | 5 |
| 327. Recargar previsualización de la encuesta: Como autoridad deseo que se vuelva a inicializar el estado de la previsualización de la encuesta luego de hacer clic en enviar. | 3 | 3 |

### 2.7.5 RETROSPECTIVA

En la retrospectiva del sprint, se identificaron varias acciones para mejora del equipo. Se propuso no dejar la deuda técnica al final, pues esto consume más esfuerzo que si se hiciera al inicio del proyecto. Además, se recomendó actualizar la definición de terminado conforme las características del proyecto cambien, en este caso, actualizarla para incluir los reportes de Power BI.

## *3 RESULTADOS, CONCLUSIONES Y RECOMENDACIONES*

## 3.1 Resultados

Tras la finalización de 40 historias de usuario desarrolladas a lo largo de 7 sprints, se tiene como resultado un conjunto de 36 endpoints cuyo detalle se muestra en la **Tabla 3.1**. Estos endpoints representan todas las reglas del negocio y estarán disponibles para su consumo desde el Front-End.

**Tabla 3.1.** Endpoints expuestos por el Back-End

| **Endpoint** | **Descripción** |
|---|---|
| /login | Obtiene una lista de todos los usuarios |
| /login/code/{code} | Obtiene un usuario por su código único |
| /login/id/{id} | Obtiene un usuario por su ID |
| /login/email/{email} | Obtiene un usuario por su correo electrónico |
| /login/email/role/{email} | Obtiene el rol de un usuario |
| /login/update-role/{id} | Actualiza el rol de un usuario |
| /login/update/{id} | Actualiza la información de un usuario |
| /survey/save/{surveyId} | Guarda las respuestas de una encuesta |
| /survey/activate/{surveyId} | Activa una encuesta |
| /survey/get-for-student/{id} | Obtiene la encuesta con las posibles asignaturas para un estudiante en particular |
| /survey/get-for-authority | Obtiene una encuesta con todas las asignaturas del periodo |
| /survey/configuration | Obtiene las configuraciones del periodo activo |
| /survey/configuration/period/{periodId} | Obtiene las configuraciones de un periodo en particular |
| /survey/configuration/career/{careerId} | Obtiene las configuraciones por carrera del periodo activo |
| /survey/configuration/career/{careerId}/ period/{periodId} | Obtiene las configuraciones por carrera de un periodo en particular |
| /survey/configuration/update | Actualiza las configuraciones |
| /curriculum/get-all-periods | Obtiene todos los periodos |
| /curriculum/period/id/{id} | Obtiene un periodo por su ID |
| /curriculum/period/code/{code} | Obtiene un periodo por su código |
| /curriculum/get-active-period | Obtiene el periodo activo |
| /curriculum/get-all-subjects | Obtiene todas las asignaturas |

| | |
|---|---|
| /curriculum/get-subjects-by-career/{idCareer} | Obtiene todas las asignaturas de una carrera |
| /curriculum/get-all-careers | Obtiene todas las carreras |
| /curriculum/get-curriculum/{careerId}/{periodId} | Obtiene los curriculums de una carrera en un periodo en particular |
| /curriculum/copy/{codePeriod} /{namePeriod}/{periodReferenceId} | Copia los curriculums de un periodo existente a uno nuevo |
| /curriculum/subject/save | Guarda una nueva asignatura |
| /curriculum/save | Guarda un curriculum en un periodo |
| /curriculum/delete/{curriculumId} | Elimina un curriculum de un periodo |
| /curriculum/period/save | Crea un periodo |
| /curriculum/period/activate/{periodId} | Activa un periodo |
| /role | Obtiene todos los roles |
| /role/id/{id} | Obtiene un rol por su ID |
| /role/code/{code} | Obtiene un rol por su código |
| /import/users | Importa usuarios desde un archivo CSV |
| /import/student-curriculum/period/{periodId} | Importa el curriculum académico de los estudiantes desde un archivo CSV |
| /auth/public-key | Obtiene una frase para la generación del token de acceso |

La exposición de estos 36 endpoints fue posible gracias a la implementación de 6 controladores, 11 servicios y 10 entidades, formando así la estructura de negocios final, que se muestra en la **Figura 3.1**. Mediante el uso de TypeORM, se realizaron transformaciones de las entidades para crear un modelo de base de datos utilizando el motor SQLServer.

La infraestructura fue desplegada en Azure, abarcando tanto el servidor web como el servidor de base de datos. Gracias a la gran variedad de planes que ofrece Azure para estos servidores, se asegura la escalabilidad de los recursos conforme el negocio así lo requiera.

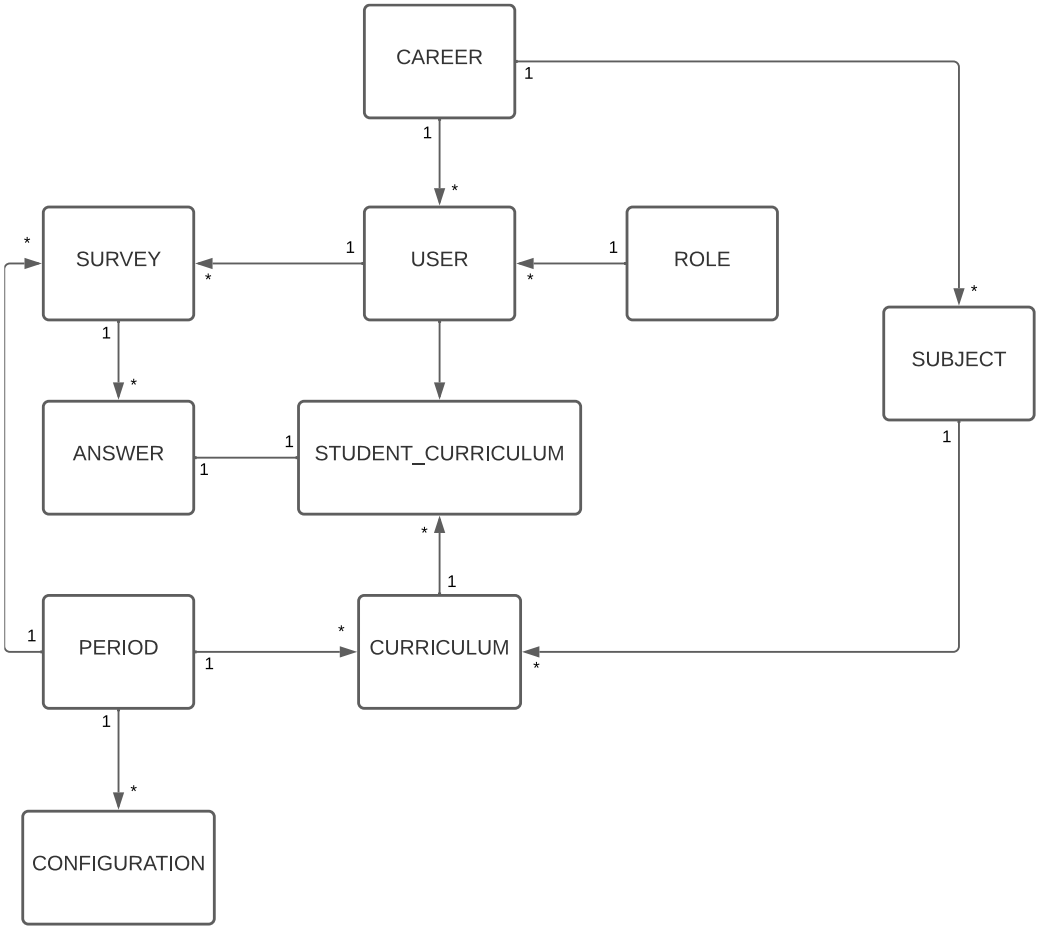



**Figura 3.1.** Estructura de objetos de negocio final

Finalmente, para garantizar un funcionamiento óptimo del componente Back-End, se ejecutaron varias pruebas unitarias. Mientras el desarrollo de las pruebas se realizó de manera incremental a lo largo de los sprints, su ejecución se realizó de manera acumulativa en cada sprint y validando que los resultados sean satisfactorios (por ejemplo, el conjunto de pruebas ejecutadas en el sprint 3 incluían el conjunto de pruebas desarrolladas en el sprint actual (sprint 3) y anteriores (sprints 2 y 1), como lo muestra la captura de pantalla de las pruebas ejecutadas en la **Figura 3.2**. Asimismo, al momento de hacer la integración continua mediante Azure Pipelines, las pruebas se ejecutaban en Azure antes de que el incremento pasara a producción. De esta manera, se aseguró el correcto funcionamiento de cada módulo antes de avanzar hacia la siguiente etapa de desarrollo.

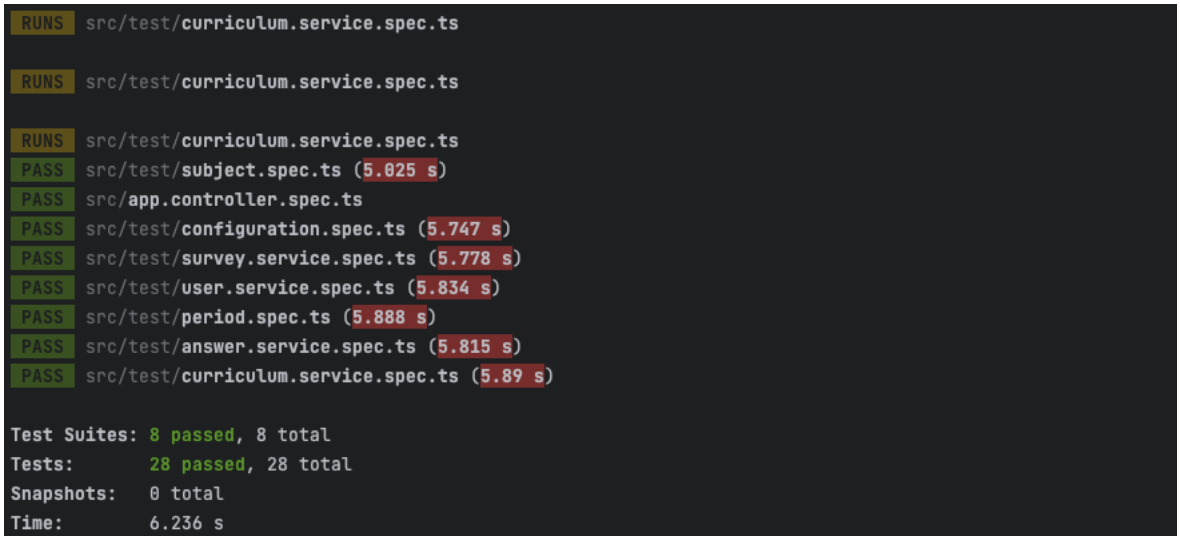



**Figura 3.2.** Ejecución de pruebas unitarias

Las pruebas unitarias contribuyeron no solo en identificar y resolver, de manera temprana, posibles errores en el código, sino también en asegurar la calidad del producto final. Verificar el comportamiento de cada módulo en condiciones controladas, minimizó la probabilidad de fallos y maximizó la confiabilidad del sistema en su conjunto.

## 3.2 Conclusiones

Esta sección presenta las conclusiones derivadas del trabajo realizado, describiéndolas en función de cada objetivo específico del proyecto.

**En relación con el objetivo específico "1. Identificar las reglas del negocio"**, se obtuvo una lista de historias de usuario (product backlog) que representan las reglas del negocio. Las historias de usuario fueron priorizadas por el Product Owner, según el valor que éstas aportaban, y se incluyeron en cada uno de los sprints. Las historias de usuario pueden ser consultadas por el lector en la sección "PLANIFICACIÓN" de cada sprint.

**En relación con el objetivo específico "2. Diseñar la arquitectura del software"**, una de las decisiones de diseño adoptada por el equipo fue aplicar el patrón de arquitectura MVC. En esta arquitectura, el controlador (C) y el modelo (M) son elementos integrantes del componente Back-End, mientras que la vista (V) es un elemento representado por el componente Front-End; el cual no es parte del desarrollo de este trabajo y documento. Los esquemas arquitectónicos que explican la construcción de los controladores y modelos (conjunto de servicios y entidades) pueden ser consultados por el lector, en la sección “IMPLEMENTACIÓN” de cada sprint. En resumen, se desarrolló un total de 6

controladores, 11 servicios y 10 entidades. El motor de base de datos utilizado fue SQLServer y toda la infraestructura fue desplegada en Azure

**En relación con el objetivo específico "3. Exponer las reglas del negocio para consumo del Front-End"**, se desarrolló, probó y expuso 36 endpoints que representan las reglas del negocio del sistema y que están disponibles para ser consumidos por el componente Front-End. La documentación de estos endpoints puede ser consultada en el ANEXO I.

Las conclusiones descritas previamente han permitido conseguir el objetivo general “Desarrollar el Back-End del sistema de preplanificación del PAO que permita comunicarse con el Front-End mediante servicios web, considerando las reglas del negocio de la malla curricular para cada estudiante”.

## 3.3 Recomendaciones

En esta sección, se presentan las recomendaciones derivadas del aprendizaje y la experiencia adquirida durante el desarrollo del proyecto. Estas recomendaciones se han formulado con el propósito de mejorar futuros procesos y proyectos similares, así como para fortalecer la calidad y eficiencia del trabajo realizado.

- Realizar las retrospectivas al final de cada sprint. Dedicar tiempo para el autoanálisis del equipo al final de cada sprint, permite al equipo identificar lo que se está haciendo mal y realizar los ajustes necesarios para mejorar, o por el otro lado, identificar qué es lo que se está haciendo bien para no dejar de hacerlo. Mediante esta práctica se puede mejorar el rendimiento de los futuros sprints.

- Es esencial contar con una definición de terminado (DoD) clara y actualizada. El DoD se convierte en un artefacto mediante el cual, el equipo de desarrollo puede determinar si el progreso logrado al final de cada sprint es realmente un avance entregable o si aún quedan aspectos por completar. Por esta razón, resulta fundamental que todo el equipo tenga un entendimiento claro de esta definición de terminado y que ésta sea continuamente ajustada conforme la evolución de las necesidades del proyecto. Mantener actualizada esta definición asegura que los entregables cumplan con los estándares de calidad y requisitos previamente establecidos, lo que a su vez contribuye a un desarrollo eficiente y exitoso del proyecto.

- Es de vital importancia aclarar todos los requerimientos con el Product Owner durante la fase de planificación. Al dedicar un mayor tiempo a esta etapa y

enfocarse en la clarificación de cada historia de usuario, se logra reducir los malentendidos, lo que resulta en la reducción de la cantidad de retrabajo que el equipo de desarrollo podría enfrentar posteriormente.

# *4 REFERENCIAS BIBLIOGRÁFICAS*


[1] "Malla Curricular – Bachillerato General Unificado – Ministerio de Educación", el 23 de junio de 2022. https://educacion.gob.ec/malla-curricular-bachillerato-general-unificado/ (consultado el 23 de noviembre de 2022).

[2] "Pre-planificación de Materias para la Carrera de SOFTWARE para el PAO 2023-A (DTIC)". https://forms.office.com/Pages/DesignPageV2.aspx?subpage=design&token=5abb23ddaefc46589b614c08ac1d7bd5&id=ak4qaH-nWEmjrJ4mbRiqN9y7HxLsEQpMmS9VI5PXklRUMlMxMlNDMElFODdUVE1RVUxHMEc4WjdZNy4u (consultado el 1 de marzo de 2023).

[3] "MALLA CURRICULAR Carrera: SOFTWARE". https://fis.epn.edu.ec/images/mallas_curriculares/MallaCurricularSoftware20.pdf (consultado el 1 de marzo de 2023).

[4] M. A. Ali Hammoudeh, A. Alobaid, A. Alwabli, y F. Alabdulmunim, "The Study on Assessment of Security Web Applications", doi: 10.3991/ijim.v15i23.27357.

[5] M. Jazayeri, "Some trends in Web application development", *FoSE 2007: Future of Software Engineering*, pp. 199–213, 2007, doi: 10.1109/FOSE.2007.26.

[6] D. Chady, "A SHORT STUDY ON THE CURRENT STATUS OF WEB APPLICATIONS SECURITY IN AFRICA AND ACROSS THE WORLD", *MATTER: International Journal of Science and Technology*, vol. 5, pp. 229–238, 2019, doi: 10.20319/mijst.2019.52.229238.

[7] G. Stuart, "HTML5 and the Canvas Element", *Introducing JavaScript Game Development*, pp. 3–16, 2017, doi: 10.1007/978-1-4842-3252-1_1.

[8] J. Attardi, *Modern CSS: Master the key concepts of CSS for modern web development*. Springer, 2020.

[9] R. Fielding y J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", jun. 2014, doi: 10.17487/RFC7230.

[10] "Hypertext Transfer Protocol (HTTP) Method Registry". https://www.iana.org/assignments/http-methods/http-methods.xhtml (consultado el 1 de junio de 2023).

[11] A. Pavlenko, N. Askarbekuly, S. Megha, y M. Mazzara, "Micro-frontends: application of microservices to web front-ends", doi: 10.22667/JISIS.2020.05.31.049.

[12] H. M. Abdullah y A. M. Zeki, "Frontend and backend web technologies in social networking sites: Facebook as an example", *Proceedings - 3rd International Conference on Advanced Computer Science Applications and Technologies, ACSAT 2014*, pp. 85–89, abr. 2014, doi: 10.1109/ACSAT.2014.22.

[13] B. M. Adam, A. Rachmat Anom Besari, y M. M. Bachtiar, “Backend Server System Design Based on REST API for Cashless Payment System on Retail Community”, *IES 2019 - International Electronics Symposium: The Role of Techno-Intelligence in Creating an Open Energy System Towards Energy Democracy, Proceedings*, pp. 208–213, sep. 2019, doi: 10.1109/ELECSYM.2019.8901668.

[14] D. Zhang, S. Lin, Y. Fu, y S. Huang, “The communication system between web application host computers and embedded systems based on Node.JS”, *Proceedings - 2017 10th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, CISP-BMEI 2017*, vol. 2018-January, pp. 1–5, feb. 2018, doi: 10.1109/CISP-BMEI.2017.8302325.

[15] S. Sotnik, T. Shakurova, y V. Lyashenko, “Development Features Web-Applications”, *International Journal of Academic and Applied Research*, vol. 7, núm. 1, pp. 2643–9603, 2023, Consultado: el 27 de mayo de 2023. [En línea]. Disponible en: www.ijeais.org/ijaar

[16] “¿Qué es una API y cómo funciona?” https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces (consultado el 13 de junio de 2023).

[17] A. Rodriguez, “RESTful Web services: The basics The basics Develop skills on this topic”, 2008.

[18] D. Corradini, A. Zampieri, M. Pasqua, y M. Ceccato, “Restats: A Test Coverage Tool for RESTful APIs”, *Proceedings - 2021 IEEE International Conference on Software Maintenance and Evolution, ICSME 2021*, pp. 594–598, ago. 2021, doi: 10.1109/ICSME52107.2021.00063.

[19] A. Singjai, U. Zdun, O. Zimmermann, y C. Pautasso, “Patterns on Deriving APIs and their Endpoints from Domain Models”, *ACM International Conference Proceeding Series*, jul. 2021, doi: 10.1145/3489449.3489976.

[20] S. Sciancalepore, G. Piro, D. Caldarola, G. Boggia, y G. Bianchi, “OAuth-IoT: An access control framework for the Internet of Things based on open standards”, *Proc IEEE Symp Comput Commun*, pp. 676–681, sep. 2017, doi: 10.1109/ISCC.2017.8024606.

[21] D. Thomas, “Programming the World in a Browser Real Men Don’t Do JavaScript Do They?!”, *JOURNAL OF OBJECT TECHNOLOGY*, vol. 6, núm. 10, pp. 25–29, Consultado: el 9 de junio de 2023. [En línea]. Disponible en: http://www.jot.fm/issues/issue_2007_10/column3

[22] G. Bierman, M. Abadi, y M. Torgersen, “Understanding TypeScript”, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8586 LNCS, pp. 257–281, 2014, doi: 10.1007/978-3-662-44202-9_11.

[23] J. Bogner y M. Merkel, “To Type or Not to Type? A Systematic Comparison of the Software Quality of JavaScript and TypeScript Applications on GitHub”, *Proceedings - 2022 Mining Software Repositories Conference, MSR 2022*, pp. 658–669, 2022, doi: 10.1145/3524842.3528454.

[24] M. Chen *et al.*, “SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review”, *Big Data and Cognitive Computing 2023, Vol. 7, Page 97*, vol. 7, núm. 2, p. 97, may 2023, doi: 10.3390/BDCC7020097.

[25] “SQL Server technical documentation - SQL Server | Microsoft Learn”. https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16&viewFallbackFrom=sql-server-2022 (consultado el 9 de junio de 2023).

[26] M. Ma, J. Yang, P. Wang, W. Liu, y J. Zhang, “Light-Weight and Scalable Hierarchical-MVC Architecture for Cloud Web Applications”, *Proceedings - 6th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2019 and 5th IEEE International Conference on Edge Computing and Scalable Cloud, EdgeCom 2019*, pp. 40–45, jun. 2019, doi: 10.1109/CSCLOUD/EDGECOM.2019.00017.

[27] A. Singh, P. Chawla, K. Singh, y A. K. Singh, “Formulating an MVC Framework for Web Development in Java”, *Proceedings of the 2nd International Conference on Trends in Electronics and Informatics, ICOEI 2018*, pp. 926–929, nov. 2018, doi: 10.1109/ICOEI.2018.8553746.

[28] A. Gachet, “A New Component in the Classification of DSS Development Tools”, *http://dx.doi.org/10.3166/jds.12.271-280*, vol. 12, núm. 3–4, pp. 271–281, 2012, doi: 10.3166/JDS.12.271-280.

[29] “Documentation | NestJS - A progressive Node.js framework”. https://docs.nestjs.com/ (consultado el 5 de junio de 2023).

[30] “First steps | NestJS - A progressive Node.js framework”. https://docs.nestjs.com/first-steps (consultado el 5 de junio de 2023).

[31] “Getting Started with Object-relational Mapping”, *Pro LINQ Object Relational Mapping with C# 2008*, pp. 3–15, sep. 2008, doi: 10.1007/978-1-4302-0597-5_1.

[32] “TypeORM - Amazing ORM for TypeScript and JavaScript (ES7, ES6, ES5). Supports MySQL, PostgreSQL, MariaDB, SQLite, MS SQL Server, Oracle, WebSQL databases. Works in NodeJS, Browser, Ionic, Cordova and Electron platforms.” https://typeorm.io/ (consultado el 6 de junio de 2023).

[33] J. Smeds, K. Nybom, y I. Porres, “DevOps: A definition and Perceived Adoption Impediments”, *Lecture Notes in Business Information Processing*, vol. 212, pp. 166–177, 2015, doi: 10.1007/978-3-319-18612-2_14/COVER.

[34] M. Shahin, M. Ali Babar, y L. Zhu, “Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices”, *IEEE Access*, vol. 5, pp. 3909–3943, 2017, doi: 10.1109/ACCESS.2017.2685629.

[35] “What is Azure DevOps? - Azure DevOps | Microsoft Learn”. https://learn.microsoft.com/en-us/azure/devops/user-guide/what-is-azure-devops?view=azure-devops (consultado el 9 de junio de 2023).

[36] “Git”. https://git-scm.com/ (consultado el 9 de junio de 2023).

[37] K. Schwaber y J. Sutherland, “The Scrum Guide The Definitive Guide to Scrum: The Rules of the Game”, 2020.

[38] S. Kenneth, “Rubin. Essential Scrum: A Practical Guide to the Most Popular Agile Process (Addison-Wesley Signature Series (Cohn))”, 2012.

# *5 ANEXOS*

## 5.1 ANEXO I

Enlace hacia la documentación detallada de cada uno de los endpoints expuestos por el Back-End: https://backendtitulacionv2.azurewebsites.net/docs