**UNIVERSIDAD TECNOLÓGICA**

**EMILIANO ZAPATA DEL ESTADO DE MORELOS**

**DIVISIÓN ACADÉMICA DE TECNOLOGÍAS DE LA INFORMACIÓN Y DISEÑO**

**MODULO EVALUACIÓN DOCENTE PARA EL SISTEMA INTEGRAL DE SERVICIOS ACADÉMICOS DE LA UNIVERSIDAD DE CIENCIAS JURÍDICAS DE MORELOS S.C**

**REPORTE DE ESTADÍA**

**QUE PARA OBTENER EL GRADO DE:**

**INGENIERÍA EN DESARROLLO Y GESTIÓN DE SOFTWARE**

|  |  |  |
| --- | --- | --- |
| **PRESENTA:**  **NOE MARTINEZ FLORES** | | |
| ASESORA EMPRESARIAL |  | ASESOR UNIVERSITARIO |
| **DRA. MARTHA FABIOLA WENCES DÍAZ** |  | **I.T.I ERICK MIRELES MERCHANT** |

**EMILIANO ZAPATA, MOR., ABRIL DE 2023**

|  |
| --- |
| **CONTENIDO** |

Índice de figuras

Índice de tablas

Agradecimientos

Resumen

Summary

[CAPÍTULO 1. DESCRIPCIÓN DEL PROYECTO 9](#_Toc163811531)

[1.1 Datos generales de la empresa 9](#_Toc163811532)

[1.2 Antecedentes del proyecto 10](#_Toc163811533)

[1.3 Objetivo general 10](#_Toc163811534)

[1.4 Objetivos específicos 10](#_Toc163811535)

[1.5 Justificación 11](#_Toc163811536)

[1.6 Alcances 11](#_Toc163811537)

[1.7 Restricciones 12](#_Toc163811538)

[CAPÍTULO 2. MARCO DE REFERENCIA 13](#_Toc163811539)

[2.1 Conceptos, metodologías y herramientas 13](#_Toc163811540)

[2.1.1 Lenguajes de Programación 13](#_Toc163811541)

[2.1.2 Herramientas de prototipado 15](#_Toc163811542)

[2.1.3 Herramientas para desarrollo frontend 17](#_Toc163811543)

[2.1.4 Herramientas para backend 19](#_Toc163811544)

[2.1.5 Herramientas de desarrollo 20](#_Toc163811545)

[2.1.6 Arquitectura 22](#_Toc163811546)

[2.1.7 Metodologías empleadas 23](#_Toc163811547)

[2.2 Propuesta de solución 25](#_Toc163811548)

[CAPÍTULO 3. DESARROLLO 27](#_Toc163811549)

[3.1 Inicio 27](#_Toc163811550)

[3.2 Planeación 28](#_Toc163811551)

[3.2.1 Diagramas de Casos de Uso 30](#_Toc163811552)

[3.2.2 Plan de Reporte de Pruebas 31](#_Toc163811553)

[3.2.3 Análisis y Diseño 32](#_Toc163811554)

[3.3 Ejecución 38](#_Toc163811555)

[3.3.1 Construcción y Codificación 38](#_Toc163811556)

[3.4 Control 59](#_Toc163811557)

[3.5 Cierre 59](#_Toc163811558)

[CAPÍTULO 4. CONCLUSIONES 61](#_Toc163811559)

[4.1 Cumplimiento de objetivos 61](#_Toc163811560)

[4.2 Resultados 62](#_Toc163811561)

[4.3 Contribuciones 62](#_Toc163811562)

**REFERENCIAS**

**ANEXOS**

|  |
| --- |
| **ÍNDICE DE FIGURAS** |

[*Figura 1.1 Logotipo de la empresa* 9](#_Toc163805981)

[*Figura 3.1 Diagrama de gantt* 28](#_Toc163805982)

[*Figura 3.2 Documento Formal de Requisitos* 29](#_Toc163805983)

[*Figura 3.3 Diagrama de casos de uso* 31](#_Toc163805984)

[*Figura 3.4 Diagrama de Despliegue* 33](#_Toc163805985)

[*Figura 3.5 Diagrama de Componentes* 34](#_Toc163805986)

[*Figura 3.6 Diagrama de Clases* 35](#_Toc163805987)

[*Figura 3.7 Diagrama de Actividades* 36](#_Toc163805988)

[*Figura 3.8 Diagrama de Secuencia* 36](#_Toc163805989)

[*Figura 3.9 Arquitectura del Proyecto* 37](#_Toc163805990)

[*Figura 3.10 Árbol de navegabilidad* 38](#_Toc163805991)

[*Figura 3.11 Encarpetado Backend* 39](#_Toc163805992)

[*Figura 3.12 Encarpetado* 41](#_Toc163805993)

[*Figura 3.13 Iterador* 42](#_Toc163805994)

[*Figura 3.14 Iterador* 43](#_Toc163805995)

[*Figura 3.15 Gateway* 44](#_Toc163805996)

[*Figura 3.16 Controlador* 44](#_Toc163805997)

[*Figura 3.17 Tabla BoostrapVue* 46](#_Toc163805998)

[*Figura 3.18 Llamada API* 46](#_Toc163805999)

[*Figura 3.19 Cabeceras para la tabla* 47](#_Toc163806000)

[*Figura 3.20 Acciones iteradas* 47](#_Toc163806001)

[*Figura 3.21 Modals en vista* 48](#_Toc163806002)

[*Figura 3.22 Función llamado de un Modal* 49](#_Toc163806003)

[*Figura 3.23 Secciones* 50](#_Toc163806004)

[*Figura 3.24 Inicio Evaluación Docente* 50](#_Toc163806005)

[*Figura 3.25 Registro de Sección* 51](#_Toc163806006)

[*Figura 3.26 Preguntas* 52](#_Toc163806007)

[*Figura 3.27 Preguntas Abiertas o Múltiples* 53](#_Toc163806008)

[*Figura 3.28 Registro pregunta* 53](#_Toc163806009)

[*Figura 3.29 Resultados evaluación docente* 55](#_Toc163806010)

[*Figura 3.30 Detalles Evaluación docente* 56](#_Toc163806011)

[*Figura 3.31 Inicio Evaluación Docente alumno* 56](#_Toc163806012)

[*Figura 3.32 Selección de docentes* 57](#_Toc163806013)

[*Figura 3.33 Preguntas Múltiple* 57](#_Toc163806014)

[*Figura 3.34 Preguntas Abierta* 58](#_Toc163806015)

|  |
| --- |
| **ÍNDICE DE TABLAS** |

[Tabla 3.1 Involucrados y sus funciones 27](#_Toc163806016)

[Tabla 3.2 Definición de etiquetas 45](#_Toc163806017)

|  |
| --- |
| **AGRADECIMIENTOS** |

Texto libre escrito por el estudiante.

|  |
| --- |
| **RESUMEN** |

Dentro del centro de desarrollo de software (CDS) se llevó acabo la implementación de un nuevo módulo para la universidad de ciencias jurídicas del estado de Morelos en el cual se hace referencia a la gestión por completo de la evaluación docente.

La inicialización de este módulo se llevó acabo con reuniones con el cliente, el cual dio a conocer sus necesidades para llevar acabo el módulo, después de ello se realizó un documento formal de requerimientos para validar cada uno de los puntos hablados en la reunión. La gestión de las funcionales que abarcan toda la evaluación docente son; secciones, preguntas y respuestas, consulta de evaluación docente y registro de la evaluación docente

Se inició con una búsqueda de las mejores opciones de tecnologías para realizar el módulo en cuestión. Se inclinó por optar las tecnologías de bootstrapVue junto con typescript para desarrollar el lado del cliente y Java junto a spring para desarrollar el lado del servidor.

Posteriormente se llevó acabo la elaboración de un diagrama de Gantt para poder medir el progreso. Despues de ello se empezó a realizar el proceso planeación junto análisis y diseño

Una vez terminado la fase de planeación y de análisis y diseño se realizó la codificación del lado del servidor utilizando API’s para comunicar el servidor con el cliente, implementado por Spring. Para visualizar y manipular los datos del módulo se utilizó BootstrapVue y TypeScript, es explicado el proceso de como los componentes de BoootstrapVue.

Una vez concluido los objetivos específicos, se obtuvo una sesión remota para mostrar el resultado del módulo, obteniendo la aprobación del cliente.

|  |
| --- |
| **SUMMARY** |

|  |
| --- |
| DESCRIPCIÓN DEL PROYECTO |

## Datos generales de la empresa

El centro de desarrollo de software es un centro de entrenamiento para personas y/o estudiantes de nivel superior enfocados en la carrera de Tecnologías de la Información y Comunicación, donde los participantes no solo adquieran conocimientos teóricos, sino que también obtengan una valiosa experiencia práctica en proyectos reales centrados en el desarrollo de software.



*Figura 1.1 Logotipo de la empresa*

CDS se encuentra dentro de las instalaciones de la UTEZ ubicada en avenida Universidad Tecnológica 1,Palo Escrito, Emiliano Zapata Morelos, C.P 62765 frente a la preparatoria CECyTE, cuenta aproximadamente con 500 empleados los cuales son distribuidos en ciertas áreas o carreras que son distribuidas a lo largo de toda la universidad.  
   
Este proyecto se realizará bajo la tutela de Erick Mireles Merchant, docente de   
la Universidad Tecnológica de Emiliano Zapata (UTEZ) que actualmente desempeña   
el cargo de docente y del proceso de Gestión de Proyectos dentro del CDS-UTEZ.

## Antecedentes del proyecto

Con el constante crecimiento de la Universidad de Ciencias Jurídicas, la gestión de estudiantes, docentes y procesos administrativos ha generado una demanda cada vez más compleja y robusta. El sistema existente, aunque funcional en ciertos aspectos, ha evidenciado limitaciones que no cumplen adecuadamente con las necesidades actuales de la institución.

Se ha decidido buscar el apoyo del Centro de Desarrollo de Software (CDS) para llevar a cabo la creación de una aplicación híbrida que aborde de manera específica y efectiva los desafíos presentes en la gestión académica y administrativa. Aunque actualmente existe un sistema en funcionamiento, su capacidad se ha quedado corta y no logra satisfacer completamente las demandas y expectativas de la universidad.

## Objetivo general

Administrar el proceso de evaluación docente en la Universidad de Ciencias Jurídicas a través de una aplicación híbrida, con el propósito de proporcionar a todos los docentes retroalimentación directa de los alumnos, permitiéndoles identificar y mejorar sus puntos de mejora o fortalezas en el ámbito educativo.

## Objetivos específicos

1. Diseñar los modelos necesarios dentro de la base de datos para almacenar la información de la evaluación.
2. Diseñar e implementar los servicios API para consulta, registro y cambio de estado de aquellos datos relacionados a la evaluación docente.
3. Implementar los privilegios que tendrán los usuarios para poder realizar los procesos que conllevan a la creación y modificación de la evaluación docente.
4. Diseñar una interfaz intuitiva y amigable para el usuario final, garantizando la accesibilidad y facilidad de uso para los estudiantes y los encargados de hacer la evaluación docente.
5. Implementar las funcionalidades para la gestión de secciones, preguntas y consulta de las respuestas de la evaluación docente.
6. Implementar pruebas funcionales y probar la aplicación en un servidor.

## Justificación

El desarrollo e implementación de un Módulo de Evaluación Docente en la Universidad de Ciencias Jurídicas es esencial para optimizar el proceso de evaluación del desempeño docente. Este módulo permitirá a los estudiantes expresar sus opiniones y evaluaciones de manera estructurada, recopilando datos significativos y proporcionando retroalimentación detallada.

La introducción de este sistema contribuirá a la transparencia y rendición de cuentas, promoviendo una cultura de mejora continua en la universidad. Además, alineado con las mejores prácticas educativas, el Módulo de Evaluación Docente no solo cumplirá con estándares de calidad, sino que también fortalecerá la reputación de la institución al demostrar su compromiso con la excelencia académica y la satisfacción estudiantil.

## Alcances

* Generación del manual de usuario para las respectivas funcionalidades de: secciones, preguntas, detalles de la evaluación docente, evaluación docente.
* Pruebas al sistema.
* Redacción del DFR.
* Generación de un plan de pruebas.
* Generación de análisis y diseño del módulo.
* El desarrollo del proyecto no incluirá el desarrollo de una aplicación móvil.

## Restricciones

* El proyecto debe cumplir con reglas de usabilidad para facilitar la experiencia del usuario final.
* Las reuniones son echas de manera remota, por video llamada por medio de plataformas como Google Meet, esto conlleva a ideas no completas o casos donde la información de un receptor a otro no sea lo suficientemente clara para el avance del proyecto.

|  |
| --- |
| MARCO DE REFERENCIA |

## Conceptos, metodologías y herramientas

Para comprender el contexto dentro del proyecto, es necesario conocer los conceptos acerca de las tecnologías, lenguajes, frameworks, entornos de trabajo, entre otros.

### Lenguajes de Programación

**Java**

Teniendo en cuenta que Java es uno de los lenguajes más utilizados en la actualidad, tiene un gran apoyo por parte de la comunidad y el soporte oficial, además de ello es un lenguaje multiplataforma capaz de crear cualquier tipo de aplicación robusta ya sea para un proyecto mediano o grande.

Según Tiobe “Java ha mantenido consistentemente su posición como uno de los lenguajes de programación más populares y ampliamente utilizados en la industria del desarrollo de software durante muchos años” (Tiobe, 2024).

Además de ello, java cuenta con uno de los frameworks más estables y eficaces para el manejo de este último como tecnología para manejo de datos y seguridad.

Tal como lo dice Ivakhnenko Maryna “Java es un buen lenguaje de programación debido a su portabilidad, orientación a objetos, rico ecosistema de bibliotecas y frameworks, características de seguridad robustas y una amplia documentación respaldada por una comunidad activa de desarrolladores.” (Ivakhnenko, 2018)

**JavaScript**

JavaScript es un buen lenguaje de programación para la web debido a su capacidad para crear interactividad dinámica en páginas web. Es ampliamente compatible con todos los navegadores modernos y tiene una gran cantidad de frameworks y bibliotecas desarrolladas por la comunidad que facilitan el desarrollo web rápido y eficiente.

El portal de Internet Mozilla señala que “La compatibilidad universal de JavaScript con los navegadores, combinada con su ecosistema rico y diverso, lo convierte en una herramienta esencial para el desarrollo web moderno. Frameworks como React, Angular, Vue, y Ember facilitan la construcción de aplicaciones web dinámicas e interactivas al proporcionar estructuras robustas y patrones de diseño eficientes. Estos frameworks resuelven problemas comunes, ofrecen consistencia en el desarrollo y mejoran la productividad de los desarrolladores” (Mozilla, 2024).

**TypeScript**

TypeScript es una excelente opción para proyectos porque mejora la seguridad al permitir el tipado estático opcional, facilita la escalabilidad del código con características como interfaces y clases, mejora la productividad con herramientas avanzadas de desarrollo y cuenta con un sólido respaldo de Microsoft y una comunidad activa de desarrolladores.

Según Attardi del portal logRocket “TypeScript es valorado por mejorar la seguridad del código mediante el tipado estático opcional, lo que permite a los desarrolladores identificar y corregir errores antes de la ejecución. Este lenguaje extiende JavaScript, añadiendo funcionalidades como interfaces y clases que facilitan la organización y mantenimiento del código, especialmente útil en proyectos complejos” (Attardi, 2022).

**SQL**

El lenguaje SQL, Structured Query Language, es una elección sólida para proyectos de cualquier escala debido a su capacidad integral para gestionar datos en bases de datos. Este lenguaje permite realizar operaciones fundamentales como el almacenamiento, consulta, modificación y eliminación de datos. SQL es preferido para proyectos de almacenamiento y consulta de datos debido a su facilidad de uso, optimización de consultas, soporte para transacciones y seguridad integrada.

De acuerdo con el equipo de OptimizDBA “la optimización de bases de datos, que abarca la optimización de consultas SQL, es esencial para lograr una ejecución de consultas más rápida, una mejor escalabilidad y una reducción del consumo de recursos. Destaca la importancia de la seguridad de los datos mediante la implementación de medidas para proteger los datos sensibles contra el acceso no autorizado” (TEAM, 2023).

### Herramientas de prototipado

Parte de los prototipos son una pieza fundamental para llevar a cabo dicho proceso,

ya que con esto se tiene una mejor visión de los conceptos principales y básicos del

proyecto, con esto se puede compartir las opiniones con el equipo y ver cuál opción

es la más idónea.

Para Patricia Galina del portal Iebschool “aparte fabricar un prototipo puede aportar varios beneficios como:

* Evitar hacer una gran inversión al principio en un producto que no termina de gustar a los usuarios.
* Reduce el riesgo, ya que, si alguna función del producto no guste, se deja de implementarla antes de la fabricación.
* Se puede mejorar el producto según la aceptación que ha tenido entre los clientes“ (Galina, 2021).

**Figma**

Figma es un programa disponible en la web o en forma de aplicación de escritorio, su funcionalidad principal es ayudar a realizar cualquier tipo de interfaces, ventanas, flujos de aplicación, animaciones, presentaciones de flujos dentro de una ventana o componente, entre muchas otras más. Es de gran utilidad para los diseñadores o programadores, quienes buscan realizar bocetos, mockups o flujos de programa de una aplicación o proyecto.

Según Blandino comenta “Figma bien permite diseñar prototipos web o móvil, presentaciones, es muy útil porque facilita el trabajo de diseñadores o programadores, puedes interactuar con las personas de tu equipo ya que es una herramienta colaborativa” (Blandino, 2023).

**Sketch**

Es una herramienta con funcionalidades de colaboración como Slack y Google Drive promueve una comunicación efectiva en equipo. Su capacidad para exportar fácilmente a formatos estándar y su amplia biblioteca de componentes predefinidos hacen de Vienna Sketch una opción versátil y eficiente para el diseño de productos digitales.

De acuerdo con Reina Weisheim “Sketch es una herramienta de edición fácil de usar que simplifica la creación de maquetas con funciones intuitivas como arrastrar y soltar, Sketch Symbols y guías inteligentes para una alineación precisa. Sus extensiones poderosas ofrecen plugins e integraciones para aumentar la funcionalidad y automatizar el flujo de trabajo, incluyendo Sketch Assistant para garantizar diseños libres de errores. Además, permite la coedición en tiempo real, lo que facilita la colaboración entre varios diseñadores en un mismo archivo” (Weisheim, 2023).

**UXPin**

UXPin es una plataforma confiable para la construcción de esquemas interactivos y flujos de diseño, destacándose por su capacidad para facilitar el diseño basado en código gracias a sus componentes de interfaz de usuario integrados. Esto permite acelerar el desarrollo al proporcionar una biblioteca de elementos listos para usar. Además, su enfoque en la colaboración en equipo y la revisión de prototipos en tiempo real hace que sea una opción sólida para proyectos de diseño web y de aplicaciones.

Para Reina Weisheim “UXPin ofrece una guía de estilo generada automáticamente que mantiene todos los colores, fuentes y activos sincronizados en un solo lugar, garantizando la consistencia del diseño en todo el proceso. Su disponibilidad multiplataforma para macOS, Windows y navegadores permite a los usuarios trabajar desde cualquier dispositivo. Además, sus elementos de formulario interactivos y funciones de accesibilidad web permiten diseñar experiencias de usuario más realistas y accesibles, con la capacidad de agregar lógica condicional y estados animados para mejorar las pruebas de usabilidad” (Weisheim, 2023).

### Herramientas para desarrollo frontend

**Vue JS**

Uno de los frameworks basado en el lenguaje de programación JavaScript para el desarrollo frontend es Vue JS, cuenta con un gran número de usuarios utilizando este framework, soporte y actualizaciones constantes por parte de sus desarrolladores. Cuenta con la compatibilidad de utilizar TypeScript para añadir tipado específico al lenguaje JavaScript por medio de interfaces o clases, también se puede utilizar funciones para manipulación de la información en cualquier punto de su ciclo de vida, además que es bastante ligero y fácil de usar, así como de aprender.

De acuerdo con Jesus Dongee “Permite una interfaz de usuario más reactiva y receptiva. Además de ello, facilita la construcción de interfaces de usuario escalables y mantenibles. El objetivo principal de Vue JS es simplificar el desarrollo web y los sitios web. Gracias a su pequeño tamaño de compilación, su facilidad de uso, aprendizaje, y rendimiento. Vue una opción robusta para aplicaciones complejas” (Jesus, 2022).

**Boostrap Vue**

Vue Bootstrap combina la potencia del framework de desarrollo frontend Vue.js con la versatilidad y estilos predefinidos de Bootstrap. Vue JS proporciona un enfoque reactivo para las interfaces de usuario, mientras que Bootstrap ofrece un conjunto de estilos y componentes que agilizan el diseño y la apariencia de las aplicaciones.

Según Peter Ekene Eze en el portal LogRocket “BootstrapVue se integra de forma sencilla en los proyectos de Vue y ofrece componentes que son responsivos por defecto, lo que elimina la necesidad de escribir código adicional o usar librerías externas para lograr diseños adaptables” (Ekene, 2022).

**Angular JS**

De acuerdo con Basalo en su blog “AngularJS es un framework de JavaScript que fue creado, desarrollado y está mantenido por Google, que se utiliza para la creación y desarrollo de aplicaciones web” (Basalo, 2014).

El objetivo de Google con esta librería es poder construir o desarrollar aplicaciones web por parte del cliente las cuales además de facilitar su codificación sean fáciles de mantener. Angular se enfoca en ofrecer herramientas y estructuras que faciliten el desarrollo de aplicaciones complejas, al tiempo que promueve buenas prácticas de programación y arquitectura. Su objetivo es proporcionar un ecosistema robusto para el desarrollo web que satisfaga las necesidades tanto de los desarrolladores como de los usuarios finales.

### Herramientas para backend

**Spring Boot**

Spring Boot es un marco de trabajo que facilita la creación de aplicaciones independientes basadas en Spring utilizando el lenguaje de programación Java, enfocado en la simplicidad y la rapidez.

Es ideal para proyectos grandes que cuenten con robustes y seguridad al momento de desarrollar el proyecto, permite el uso de todos los recursos del lenguaje de programación Java ayudando a reducir el tiempo de desarrollo y aumentando la productividad.

De acuerdo con el portal de Spring “Se destaca por su capacidad de manejar proyectos grandes al ofrecer herramientas listas para producción, como métricas, verificaciones de estado y configuración externa, lo que refuerza la robustez y seguridad. La configuración predeterminada simplifica el proceso de inicio de proyectos, ofreciendo una estructura predeterminada que puede ser fácilmente personalizada según las necesidades específicas del proyecto “ (Spring, 2023).

**PostgreSQL**

PostgreSQL es un sistema de gestión de bases de datos relacional (RDBMS) que destaca por su robustez, flexibilidad y capacidad para manejar cargas de trabajo variadas. Además, es un software multiplataforma, compatible con una amplia gama de sistemas operativos, lo que lo hace accesible para usuarios en diferentes entornos. Su arquitectura avanzada le permite gestionar eficientemente distintos tipos de datos, desde textos hasta imágenes y sonidos, soportando un gran volumen de estos sin comprometer el rendimiento. Esta versatilidad lo convierte en una opción sólida para proyectos de cualquier tamaño, desde aplicaciones web pequeñas hasta grandes sistemas empresariales.

Santiago Borges señala que “Cuenta con un ámbito multiplataforma para funcionar en diversas plataformas, incluyendo sistemas operativos Unix y Windows. Esta característica elimina las preocupaciones relacionadas con la compatibilidad y permite su uso tanto localmente como a través de conexiones remotas, ofreciendo flexibilidad para adaptarse a diferentes entornos. Destaca en el manejo de bases de datos con alto volumen de información gracias a su eficiente método de Control de Concurrencias Multiversión (MVCC). Este enfoque permite lecturas y escrituras simultáneas sin bloquear operaciones, mejorando significativamente el rendimiento en entornos con movimientos intensivos en la base de datos” (Borges, 2015).

### Herramientas de desarrollo

**VSCode**

VSCode cuenta con una interfaz intuitiva y con una capacidad ampliamente compatible para aceptar y ejecutar varios lenguajes de programación, lo que lo convierte en una herramienta valiosa para desarrolladores de diferentes tecnologías. Además, su amplio conjunto de extensiones permite personalizar el entorno de desarrollo según las necesidades específicas del proyecto y del desarrollador.

La integración de Git directamente en el IDE facilita el control de versiones y la colaboración en equipo. También ofrece características como IntelliSense, depuración integrada, resaltado de sintaxis y una poderosa capacidad de búsqueda, lo que mejora la productividad y la eficiencia durante el desarrollo.

Según El portal de Visual Studio Code “Es ampliamente reconocido por su interfaz intuitiva y su capacidad para manejar una amplia gama de lenguajes de programación, lo que lo convierte en una herramienta valiosa para desarrolladores de diversas tecnologías. Su compatibilidad con múltiples plataformas, incluidas macOS, Linux y Windows, asegura que los desarrolladores puedan trabajar cómodamente en cualquier entorno operativo” (VisualStudio, 2024).

**Intellij**

Es un entorno de desarrollo que contiene diferentes herramientas avanzadas para la escritura y revisión de código. Adicionalmente, subraya la flexibilidad y capacidad de personalización del entorno de trabajo, y su compatibilidad con diversos lenguajes de programación y plataformas importantes para el desarrollo de aplicaciones modernas.

Según el portal de internet Trucoteam “Es un IDE que ofrece una amplia variedad de ventajas, Mejora de la productividad ya que ofrece un conjunto de herramientas que te ayudarán a optimizar tu tiempo al escribir código y revisar errores, reduciendo las horas de trabajo necesarias para programar una aplicación. Ofrece un entorno de trabajo flexible se puede personalizar fácilmente para adaptarse a tus necesidades y a la arquitectura de tu aplicación. Es compatible con Java, Kotlin, Groovy y otros lenguajes de programación más recientes, y también ofrece una solución óptima para plataformas como Android, Spring, Maven y Gradle” (Trucoteam, 2024).

**GIT**

Git es un sistema de control de versiones distribuido, forma parte de los equipos de desarrollo donde gestionan y colaboran en el código fuente. Git no solo es usado únicamente por desarrolladores o creadores de código, también es utilizado por profesionales de distintos enfoques como Diseñadores digitales ya que su uso no es únicamente para archivos de programación, si no que involucra cualquier tipo de archivo electrónico.

Permite un seguimiento preciso, facilita la creación y fusión de ramas, optimiza la colaboración entre los integrantes de un equipo o de manera individual, y garantiza un historial detallado de los cambios realizados a los archivos afectados o modificados.

De acuerdo con Astigarraga y Cruz “Git permite ‘rastrear’ el progreso de un proyecto a lo largo del tiempo ya que hace ‘capturas’ del mismo a medida que evoluciona y los cambios se van registrando. Esto permite ver qué cambios se hicieron, quién los hizo y por qué, e incluso volver a versiones anteriores. Además, Git facilita el trabajo en paralelo de varios participantes.” (Julen Astigarraga, 2022)

### Arquitectura

**Clean Arquitecture**

La arquitectura limpia promueve la separación de preocupaciones, mejorando la mantenibilidad y la escalabilidad del código. Al aislar las dependencias y centrarse en la separación de la lógica de negocio de la interfaz de usuario y la base de datos, facilita las pruebas y la depuración, y permite adaptaciones futuras del software sin grandes sobrecostos.

Cambarieri señala que “permiten separar la complejidad del negocio, dejando como pieza central del mismo, el dominio. El enfoque diseño dirigido por el dominio ofrece la posibilidad de contar con principios, patrones y actividades para construir un modelo de dominio, que es el artefacto principal. Además, ayuda a garantizar que la arquitectura de software permanezca centrada en las funcionalidades del negocio” (Cambarieri, 2020).

**Modelo Vista Controlador**

La arquitectura Modelo Vista Controlador (MVC) promueve un desarrollo más estructurado y modular. Esto no solo mejora la claridad y la división de responsabilidades entre los componentes, sino que también permite que equipos diferentes trabajen en partes distintas de la aplicación simultáneamente, aumentando la eficiencia.

Según Camarena Sagredo, Trueba Espinosa, Martínez Reyes, y López García “El patrón Modelo, Vista y Controlador (mvc) es el más extendido para el desarrollo de aplicaciones donde se deben manejar interfaces de usuarios, éste se centra en la separación de los datos o modelo, y la vista, mientras que el controlador es el encargado de relacionar a estos dos. En este sentido en el patrón mvc, el acceso a datos depende directamente del mismo modelo que se mapea por medio de la consulta sql (vista), por lo tanto, los dto y dao corresponden a una estructura muy acoplada a la vista, ya que los objetos de intercambio (dto) dependen directamente de los dao y la generación de estos dependen directamente del modelo.” (Camarena Sagredo, Trueba Espinosa, Martínez Reyes, & López García, 2012)

**Cliente Servidor**

Esta arquitectura separa las responsabilidades entre el cliente, que se encarga de la interfaz de usuario y la interacción con el usuario final, y el servidor, que maneja el procesamiento de datos y la lógica de negocio.

Esta separación facilita la escalabilidad, el mantenimiento y la gestión del sistema, ya que permite realizar cambios en el cliente o en el servidor de manera independiente sin afectar al otro componente, lo que resulta en un desarrollo más organizado y eficiente.

De acuerdo con Alejandra Burin “Este tipo de patrón de arquitecturas de software refiere a la existencia de un servidor que proporciona un servicio a un cliente (persona o empresa). Cuando el cliente solicita determinados datos al servidor, el servidor acepta el proceso y entrega los datos que solicitó el cliente” (Burin, n.d.).

### Metodologías empleadas

**SCRUM**

Es una metodología ágil la cual se acopla de manera idónea al desarrollo de software, cada uno de los interesados o integrantes del proyecto toma un rol dentro del marco de SCRUM y cumple con ciertas funciones, además de ello, contiene varias ceremonias que ayudan al equipo a tener un ciclo de trabajo basado en sprints que constan entre dos a cuatro semanas.

Según Ken Schwaber y Jeff Sutherland “Scrum involucra a grupos de personas que colectivamente tienen todas las habilidades y experiencia para hacer el trabajo y compartir o adquirir tales habilidades según sea necesario” (Schwaber & Sutherland, 2020).

El equipo de Scrum se compromete a lograr sus objetivos y apoyarse mutuamente. Su enfoque principal es el trabajo del Sprint para hacer el mejor progreso posible hacia estos objetivos. El equipo de Scrum y sus partes interesadas están abiertos sobre el trabajo y los desafíos. Los miembros del equipo de Scrum se respetan mutuamente para ser personas capaces e independientes, y son respetados como tales por las personas con las que trabajan. Los miembros del equipo de Scrum tienen el valor de hacer lo correcto y de trabajar en problemas complejos.

**Extreme Programming**

La metodología Extreme Programming (XP) es efectiva en el desarrollo de software por su enfoque en adaptabilidad y colaboración entre el equipo de TI y los stakeholders. Permite ajustar el proyecto según necesidades específicas al definir variables críticas como coste, tiempo y calidad, con una responsabilidad compartida que mejora la toma de decisiones y optimiza resultados. Esto la convierte en una opción robusta y flexible para enfrentar los desafíos del desarrollo tecnológico.

De acuerdo con Jose Canós, Patricio Letelier y Carmen Penadés “Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico” (José H. Canós, Patricio Letelier y Ma Carmen Penadés).

**Método WaterFall o Cascada**

La metodología de cascada es efectiva para el desarrollo de software porque ofrece un enfoque secuencial claro, donde cada etapa debe completarse antes de pasar a la siguiente. Esto reduce los riesgos y facilita la gestión del proyecto, haciendo el proceso más accesible y predecible para todos los involucrados.

De acuerdo con J. Gómez Cervantes “Los modelos de desarrollo de software se basan en un enfoque secuencial que incluye análisis de requerimientos, diseño, codificación, validación, y mantenimiento, asegurando además una documentación detallada del sistema. Variantes del modelo de cascada ofrecen flexibilidad; por ejemplo, permitiendo solapamiento de fases para realizar actividades en paralelo, dividiendo el proyecto en subproyectos que se desarrollan simultáneamente o incorporando estrategias de reducción de riesgos que permiten identificar y mitigar problemas desde temprano, aunque manteniendo siempre el orden secuencial característico” (Gómez, 2020).

## Propuesta de solución

Basándonos en la investigación previa, se determinó que para el desarrollo por parte del cliente para la aplicación hibrida se utilizará la tecnología Vue, en conjunto con el framework BootstrapVue. Esto se debe a las ventajas que ofrecen en la interacción con el usuario a través de sus interfaces gráficas y la facilidad de uso para los programadores en conjunto al IDE VSCode. Además, se optará por PostgreSQL como base de datos para acceder a la información almacenada gracias a su manejo y consulta especializada en datos. Se empleará Spring Boot como soporte para el lado del servidor, utilizando IntelliJ IDEA para su desarrollo. Los protitpos y mockups serán realizados por medio de la herramienta figma contando con una gran variedad de funcionalidades de diseño como de desarrollo cooperativo entre los desarrolladores lo que facilita la comunicación y el flujo de trabajo.

La elección de estas herramientas se fundamenta en su facilidad de uso y en las funcionalidades que ofrecen. Por ejemplo, Spring permite la separación de los microservicios para facilitar su reutilización, así como una rápida ejecución y la capacidad de conectarse a una base de datos. Por otro lado, Vue simplifica el desarrollo al extender el HTML con directivas que crean componentes, clases y atributos con funcionalidades predefinidas.

Estas tecnologías se seleccionaron con el objetivo de generar un producto eficiente y ligero, que aproveche al máximo las distintas funcionalidades que ofrecen. Al tratarse de un proyecto nuevo, se espera obtener una aplicación fácil de usar para el usuario final. Además, se prevé la necesidad de buscar información para implementar algunas tecnologías menos conocidas, lo cual se realizará a través de la documentación oficial proporcionada por los desarrolladores.

|  |
| --- |
| DESARROLLO |

## Inicio

Con el fin de obtener un poco más del conocimiento acerca del funcionamiento y problemática acerca de este módulo, se requirió llevar a cabo una reunión con el responsable de proyecto y el cliente, con el fin de identificar satisfactoriamente las funciones y el objetivo exacto que se necesita para poder implementar adecuadamente el desarrollo de la evaluación docente dentro del sistema ya existente.

En dicha reunión, donde estuvieron presentes el cliente, responsable de desarrollo, administrador de proyecto y el equipo de desarrollo, se llegó al objetivo de que la universidad requiere la funcionalidad de evaluación para que todos los estudiantados puedan ofrecer una completa retroalimentación de sus docentes y opiniones acerca de ellos, junto a que la universidad tendrá el poder consultar aquellas respuestas en cualquier momento. Los diferentes involucrados del proyecto se encuentran definidos como se observa en la Tabla 3.1

Tabla 3.1 Involucrados y sus funciones

|  |  |  |
| --- | --- | --- |
| **INVOLUCRADOS EN EL PROYECTO** | | |
| **Nombre de la persona** | **Función** | |
| José Christian Narváez Figueroa | | Sponsor  *Interno* |
| Mónica Sotelo Rosales | | Cliente  *Externo* |
| Erick Mireles Merchant | | Asesor *Interno /* Administrador del proyecto *Interno* |

## Planeación

Para realizar la planeación correspondiente al proyecto y que este tuviera un control de las actividades a realizar en el desarrollo, se requirió elaborar la documentación necesaria donde se tomó en cuenta las fechas para la entrega del módulo a cumplir.

**Diagrama de Gantt**

Uno de los documentos para planificar el desarrollo es el diagrama de Gantt como se muestra completo en Figura 3.1 Diagrama de Gantt y dentro del ANEXO A. DIAGRAMA DE GANTT cual relata las actividades que se deben hacer en cada una de las semanas para llevar un buen control en el desarrollo y no presentar retrasos en ello. Este abarca las 15 semanas que dura el cuatrimestre en el cual se tuvo el tiempo para completar el proyecto.

A graph with green squares

Description automatically generated

*Figura 3.1 Diagrama de gantt*

**Documento Formal de Requerimientos**

Para lograr comprender e identificar los requerimientos funcionales y no funcionales del módulo de evaluación docente, se llevó a cabo la realización de un análisis de la información obtenida en la reunión con el cliente, dónde se pudo identificar los puntos que permitieron la elaboración del módulo y plasmarlo en el documento formal de requerimientos como se muestra en la Figura 3.2 Documento Formal de requisitos y en el ANEXO B. DOCUMENTO FORMAL DE REQUISITOS.

**A document with text on it

Description automatically generated**

*Figura 3.2 Documento Formal de Requisitos*

El documento Formal de Requerimientos define todas las necesidades del cliente en manera de funcionalidades, entre ellos se encuentra. Secciones, Preguntas, Consulta de Respuestas, Evaluar docente por Alumno y Activar/Desactivar evaluación. En cada una de las funcionalidades se define los datos de entrada, datos de salida (si este aplica), criterios de aceptación, reglas de negocio, requerimientos funcionales y Notas para aclarar posibles dudas o inquietudes por parte de los involucrados.

Esto permite que tanto el desarrollador como el cliente tengan los detalles claros con respecto a lo que podrá hacer la aplicación una vez que se llegue a su etapa de entrega.

Una vez finalizado el Documento Formal de Requerimientos, se iniciaron diversos procesos de acuerdo con el marco DMS que regula las operaciones dentro de los proyectos en el Centro de Desarrollo de Software. Estos procedimientos comprendieron la revisión y validación de los requisitos por parte de los interesados, así como la planificación detallada de las etapas de desarrollo y las actividades de prueba. Además, se definieron los roles y responsabilidades de los miembros del equipo de desarrollo, lo que garantizó una comprensión clara de las tareas asignadas y los objetivos del proyecto por parte de todos los involucrados. Este enfoque metodológico contribuyó a establecer una base sólida para el desarrollo del software y a mantener una alineación adecuada entre las expectativas del cliente y la ejecución del proyecto.

### Diagramas de Casos de Uso

El diagrama fue elaborado con el propósito de establecer un modelo de representación que abarque los distintos requerimientos del módulo en cuestión. Este modelo consta de la identificación de los diversos roles que pueden interactuar con la aplicación, así como de las funciones asociadas a cada uno de estos roles, incluyendo las que se desglosan de estas funciones principales.

Inicialmente, se creó un diagrama general que ilustra estos casos de uso, tal como se muestra en la Figura 3.3 Diagrama de casos de uso. Posteriormente, se detallaron estos casos de uso en él. El propósito fundamental de este enfoque es proporcionar una comprensión clara y completa de las interacciones que pueden ocurrir dentro del sistema, tanto desde la perspectiva de los usuarios como de las funcionalidades que estos pueden llevar a cabo. Esto facilita el diseño y desarrollo del módulo, al establecer una base sólida para la implementación de las diversas características y su integración en la aplicación final.

A diagram of a process

Description automatically generated

*Figura 3.3 Diagrama de casos de uso*

### Plan de Reporte de Pruebas

En este documento se comienza a estructurar el correcto funcionamiento deseado para la aplicación. Se abordan aspectos tanto del diseño estructurado como del funcional, incluyendo la realización de pruebas pequeñas destinadas a validar el comportamiento y la lógica del sistema.

Este proceso implica la identificación y descripción detallada de los diferentes comportamientos esperados de la aplicación. Estas descripciones proporcionan una visión clara de las funcionalidades y escenarios de uso previstos, lo que constituye una base sólida para el diseño y desarrollo subsiguientes. La realización de estas pruebas es esencial para asegurar que la aplicación cumpla con los requerimientos establecidos y funcione de manera coherente y efectiva una vez implementada.

**Pruebas de Sistema**

En pruebas de sistema se describe todo lo relacionado en comprobar el buen funcionamiento de la aplicación por requisito. Se trata de relatar el número del requisito antes puesto en el documento formal de requisitos, la sesión, que quiere decir, qué roles lo pueden ejecutar, el nombre del componente, una pequeña descripción de lo que trata, los prerrequisitos si llegara a aplicar, que estos son una especie de aprobación, por su contenido o por especiales razones administrativas, los datos de entrada que necesita para esta prueba, el procedimiento que se debe hacer para llegar a la respuesta ideal, la acción que hace esta prueba y el resultado esperado, que en este caso pueden ser varios resultados esperados, contando los correctos e incorrectos.

**Pruebas No Funcionales**

Dentro de estas pruebas se describen la verificación de los requisitos no funcionales basados en la operación de un software, es decir, no se basa en la funcionalidad, puede ayudar a determinar la carga que soporta el producto, o para ver si su rendimiento es el correcto con el servidor.

En esta también se describen los requerimientos de interfaz de usuario que son las expectativas que esta tiene por cumplir con las necesidades del sistema. También se describen los requerimientos de confiabilidad, los de eficiencia que esto es el acto y consecuencia de pedir o necesitar algo, los de mantenimiento, los de portabilidad, los de interoperatividad, los de reusabilidad entre otros requerimientos no funcionales.

**Pruebas de Integración**

Estas pruebas se hacen una vez que ya se aprobaron las pruebas unitarias y lo que esto hace es probar todos los elementos unitarios que componen el software, que funcionan juntos correctamente probándolos en grupo, así mismo para verificar el funcionamiento correcto de las interfaces entre los distintos subsistemas que lo componen. Esto con el fin de verificar que interactúan correctamente a través de sus interfaces, tanto como las internas como externas y que cubran su funcionalidad establecida.

### Análisis y Diseño

**Diagrama de despliegue**

En este diagrama se muestra la arquitectura de cómo está el sistema, del despliegue de los artefactos, para este diagrama se separó el BackEnd y el FontEnd para tener una mejor visualización de las tecnologías que le pertenecen a cada uno, quedando así de esta manera como se muestra en la siguiente Figura 3.4 DIAGRAMA DE DESPLIEGUE

A screenshot of a computer

Description automatically generated

*Figura 3.4 Diagrama de Despliegue*

La primera capa llamada como “Dispositivos Estándar de Entrada y Salida” son los dispositivos y/o medios en los cuales se podrá utilizar la aplicación hibrida. Dentro del cual se pueden encontrar dispositivos electrónicos como:

* Computadoras
* Tabletas
* Teléfonos Celulares – Smartphones

La segunda capa llamada “Capa de Presentación” son aquellas tecnologías empleadas en el esquema del frontend o en palabras resumidas, lo que verá el usuario final. Se estará utilizando un supe complemento de Javascript como un framework de este mismo lenguaje de programación para el manejo de las vistas conocido como BootstrapVue. A la vez que se estará utilizando una librería de NPM llamada axios quien será la encargada de realizar todas las peticiones a los diferentes endpoints que surjan del backend.

La tercera capa llamada “Capa de controlador” tiene como objetivo mostrar aquellas tecnologías utilizadas para el uso del manejo de lógica y manejo de datos por parte del backend, utiliza el lenguaje de programación Java junto a su framework Springboot el cual utilizará Spring Security para el manejo de seguridad dentro del proyecto el cual funcionará por controladores que a su vez serán esenciales para e llamar a los servicios API’s los cuales serán encargados de brindar los respectivos datos solicitados para una acción.

La cuarta capa llamada “Capa de persistencia de datos” tiene el manejo de del control del almacenamiento de datos utilizando JPA que facilita el mapeo objeto-relacional la gestión de transacciones y la realización de consultas para el fututo almacenamiento dentro de la base de datos.

**Diagrama de componentes**

Este diagrama consiste en separar como su nombre todos los componentes que hay y adecuarlos en su módulo correspondiente, describiendo dentro de los componentes todos los requerimientos que este puede hacer, basándose siempre del documento formal de requisitos. Este mismo muestras las relaciones que tienen entre ellos, así como sus dependencias entre estos componentes, como se muestra en la siguiente Figura 3.5 Diagrama de componentes.

A screenshot of a computer screen

Description automatically generated

*Figura 3.5 Diagrama de Componentes*

**Diagrama de Clases**

Este diagrama fue creado de acuerdo al diagrama de componentes, donde, del sistema existente se han agregado nuevos elementos para llevar acabo la evaluación docente, anexando en ellos todos los atributos que se describieron en el documento formal de requisitos para llevar a cabo su registro, de igual forma llevando una relación entre ellos para tener mejor visualización entre las clases al momento de crearse. A continuación, se muestra la Figura 3.6 Diagrama de Clases y dentro del ANEXO C. DIAGRAMA DE CLASES.

**A computer screen shot of a diagram

Description automatically generated**

*Figura 3.6 Diagrama de Clases*

**Diagrama de Actividades**

Este es un diagrama de flujo que ayuda a representar de manera gráfica el algoritmo o proceso de cada uno de los requerimientos con el fin de que, cuando este pase a código, se tenga una clara comprensión de las acciones que se ejecutarán según las distintas opciones que se presentan. Este enfoque visual facilita la identificación de posibles errores o ineficiencias en las etapas tempranas del desarrollo, permitiendo ajustes antes de la implementación.

Básicamente se modela el comportamiento que tiene el sistema, mostrando así una secuencia de acciones, como se muestra en el Figura 3.7 Diagrama de Actividades ANEXO D. DIAGRAMA DE ACTIVIDADES

**A diagram of a flowchart

Description automatically generated**

*Figura 3.7 Diagrama de Actividades*

**Diagrama de Secuencia**

Este diagrama se desarrolló con el fin de tener un acercamiento a como se iban a llamar las clases y los métodos para las funcionalidades, lo que se pedirá de valor inicial en cada uno de ellos, lo que va a devolver cada una de las peticiones solicitadas, todo esto para tener claro la interacción entre los objetos de un sistema, esto se hace por cada caso de uso, para determinar qué objetos son necesarios para la implementación del escenario, como se muestra en la Figura 3.8 Diagrama de secuencia y en el ANEXO E. DIAGRAMA DE SECUENCIA.

**A screenshot of a computer

Description automatically generated**

*Figura 3.8 Diagrama de Secuencia*

**Arquitectura del Producto**

Este diagrama se realizó con el fin de conocer el comportamiento e interacción de los componentes una vez estructurados en micro servicios, es muy similar al diagrama de despliegue, con la diferencia que incluye los servicios, controladores y los repositorios estructurados, así como el uso de API’s que ayudan persistencia de datos almacenados en PostgreSQL, además del añadido del uso de JWT para la generación de tokens de autenticación para el proyecto. La arquitectura del producto quedó de la siguiente forma como se muestra en la Figura 3.9 Arquitectura del Proyecto

A diagram of a computer network

Description automatically generated with medium confidence

*Figura 3.9 Arquitectura del Proyecto*

**Árbol de Navegabilidad**

Este documento se elaboró con el fin de proporcionar una visión más clara y detallada sobre la navegabilidad de la aplicación durante su desarrollo, ofreciendo un mapa exhaustivo de cómo se interconectan las diversas pantallas y funciones. Se tuvo en cuenta la jerarquía de la información y las acciones prioritarias, facilitando así una navegación intuitiva y eficiente que contribuye al diseño de una interfaz amigable y accesible para el usuario final.

Esta representación se muestra de manera gráfica en la Figura 3.10 Árbol de navegabilidad y dentro del Anexo F. ÁRBOL DE NAVEGABILIDAD. El objetivo principal fue asegurar que el diseño de la aplicación permitiera una navegación intuitiva y eficiente para los usuarios, facilitando así su experiencia de uso.

**A diagram of a company

Description automatically generated**

*Figura 3.10 Árbol de navegabilidad*

**Mockups**

Dada a las necesidades escritas dentro del Documento Formal de Requerimientos (DFR) se hicieron varias pantallas denominadas Mockups, los cuales son un bosquejo de la idea por cada una de las funcionalidades implícitamente agregadas en el DFR. La concepción de los mockups sirvió para ayudar al equipo de Frontend y al cliente en conocer mejor el flujo de los datos y el flujo que un usuario tendría que seguir para llevar a cabo cada una de las funcionalidades que cuenta todo el Módulo de evaluación Docente.

Los mockups fueron enviados y posteriormente revisados por el cliente, el cual presento su respectiva retroalimentación para poder añadir o eliminar elementos que en ese momento consideraba necesarios.

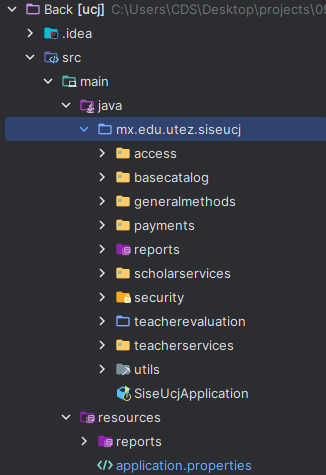
## Ejecución

### Construcción y Codificación

Para la inicialización correcta del proyecto se necesitó clonar el repositorio del proyecto a una carpeta, dicho proceso fue realizado con el control de versiones Git. Actualmente, hay diferentes ramas las cuales contienen los cambios del repositorio remoto dentro de nuestro dispositivo de manera local. Para términos de este proyecto se creó una rama local que posteriormente será unida a la rama principal.

**Construcción del Backend**

Para la construcción del proyecto por parte del backend se utilizó la tecnología antes elegida, la cual fue spring, una vez creado el proyecto en spring se le dio el orden de las carpetas antes acomodadas en el documento de registro de rastreo, cada uno con sus módulos y carpetas correspondientes como lo muestra la Figura 3.11 Encarpetado Backend.



*Figura 3.11 Encarpetado Backend*

Para todos las clases, esta representa la estructura de los datos que deseas almacenar en tu base de datos. En Spring, se anota con @Entity. Contiene atributos que representan las columnas de la tabla en la base de datos. Esto se puede apreciar en el Anexo G ENTIDADES.

Se continuó con la programación de los archivos Data Transfer Object (DTO). El DTO es una clase que se utiliza para transferir datos entre el frontend y el backend. Se usa para definir la estructura de datos que se envía o recibe en las solicitudes HTTP. En el DTO, se pueden aplicar validaciones como @NotBlank para asegurar que los datos sean válidos antes de ser procesados.

Además de la implementación de las clases DTO, se dedicó tiempo a optimizar la gestión de errores en las solicitudes HTTP. Se incorporaron manejo de excepciones específicas para diferentes tipos de errores, lo que permite una respuesta más clara y precisa a los usuarios o clientes del sistema. Esta mejora en la gestión de errores no solo incrementa la robustez del sistema, sino que también mejora la experiencia del usuario al proporcionar mensajes de error más descriptivos y útiles.

Una vez hecho eso, se continuó creando un repositorio, el repositorio es una interfaz y esta proporciona métodos para realizar operaciones CRUD en la base de datos, como guardar, eliminar, buscar, etc.

Una vez finalizada la implementación del repositorio, se procedió a la creación de la clase servicio. El servicio constituye el núcleo de la lógica de negocio de la aplicación. En esta capa se desarrollan los métodos encargados de llevar a cabo las operaciones de creación, lectura, actualización y eliminación de registros en la base de datos. Utilizando el servicio, este importa el repositorio correspondiente a su función designada y utilizará sus métodos para interactuar con la base de datos de manera eficiente y organizada.

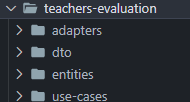
Además de la creación de los métodos básicos de CRUD (Crear, Leer, Actualizar, Eliminar), se puso énfasis en la implementación de lógica adicional para validar los datos antes de ser almacenados o modificados en la base de datos. Se aprovechó este punto para integrar la validación de datos en línea con las mejores prácticas de seguridad y consistencia de datos.

Ya teniendo terminado el servicio, repositorio, DTO y el modelo, finalmente se desarrolló el controlador. El controlador es responsable de manejar las solicitudes HTTP entrantes y enviar respuestas de vuelta al cliente. Se anota con @RestController y define métodos que corresponden a diferentes endpoints de la API. El controlador utiliza el DTO para recibir datos del cliente y el servicio para procesar y manipular esos datos como se muestra en la Anexo H ENDPOINTS.

**Construcción del Frontend**

Dentro del proyecto clonado se encuentra la carpeta 094-SISE-UCJ/Front/sise-ucj/src/modules. Dentro de la carpeta de módulos es donde todos los módulos junto a sus respectivas funcionalidades que tiene el proyecto. Para este caso se creó una carpeta con el nombre teachers-evaluation para desarrollar todas las funcionalidades de la evaluación docente.

Como se puede observar en la Figura 3.12 Encarpetado, se muestra el contenido que la carpeta teachers-evaluation contiene, esto gracias a la metodología arquitectura limpia



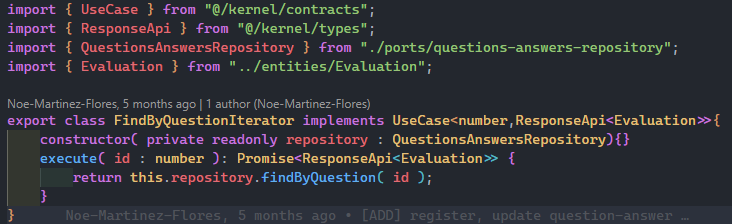
*Figura 3.12 Encarpetado*

Como primer punto a destacar se necesita crear un archivo dentro de use-cases/ports para poder definir todos aquellos métodos que posteriormente serán utilizados con los servicios provistos por él backend. Esto utilizando TypeScript, Se utiliza interfaces ya que proporciona beneficios en términos de abstracción, desacoplamiento, flexibilidad, pruebas unitarias, legibilidad y tipado estático, lo que contribuye a la creación de sistemas más robustos y mantenibles. Esto se puede ver en el Anexo G Repositorio

Posteriormente se creó un archivo por cada interfaz definida, dentro del cual se le conocerá como un iterador. Para esto, se necesita declarar una nueva Clase en TypeScript el cual implementará una interfaz previamente definida en el proyecto conocido como *UseCase* o Caso de Uso, el cual recibe dos tipos de parámetros, estos parámetros son esenciales para el correcto funcionamiento de esta implementación. El primer parámetro funciona como dato de entrada, y el segundo parámetro recibido es lo que espera ser regresado. Esto es completamente dependiente del tipo de dato se espera tanto por recibir como por devolver.

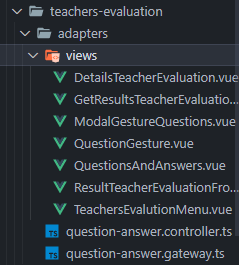
Posteriormente se define un constructor, los constructores son utilizados para realizar una función en específico cuando se llama una instancia de la misma clase antes creada.

El cual se indica que recibirá solamente un atributo el cual solo puede ser solo lectura y usable solo en este punto del código, para ello se utilizaron las palabras reservadas de private y readonly, este atributo se debe especificar el tipo de dato que corresponde para posteriormente definir la ejecución del código, el cual recibe como parámetro un dato con el tipo de dato definido previamente en el useCase, para posteriormente indicar que regresará una promesa. Se define un tipo de promesa el cual regresará uno o varios datos, esto dependiendo del tipo de dato, haciendo referencia la segundo parámetro que recibió el UseCase. Esto se puede mostrar en la Figura 3.13 Iteratdor.



*Figura 3.13 Iterador*

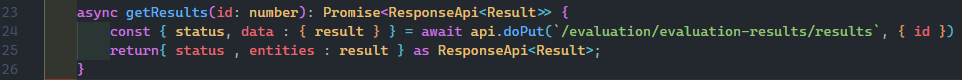
Posteriormente se dentro de la carpeta adapters se crearon dos archivos, el primero de ellos hace referencia a tener la funcionalidad de un Gateway, el segundo de un Controlador. Esto se puede mostrar en la Figura 3.14 Adaptadores



*Figura 3.14 Iterador*

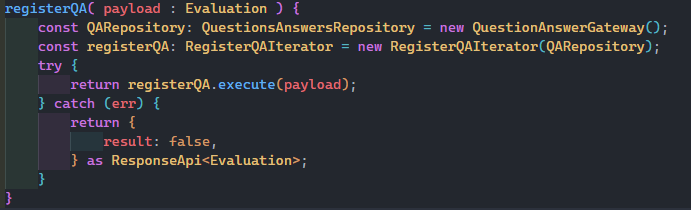
Dentro del archivo con terminación gateway.ts, fue creada una clase que implementa todos las interfaces definidas anteriormente, estas interfaces pueden ser vistas en el Anexo G Repositorio, la implementación de estas interfaz van acorde a cada uno de los endpoints definidos por parte del backend, su estructura cuenta primeramente la definición de la palabra reservada async, el cual indica que es un evento asíncrono posteriormente, el o los parámetros necesarios para el uso del futuro endpoint, justo después se deja explicito que tipo de dato regresará la función, en todos estos casos son promesas con diferentes tipos de datos. Seguidamente se utiliza la palabra reservada await, que indica que el tiempo de ejecución del código en ese preciso momento del tiempo no será síncrono si no que podrá tomar más tiempo. Utilizando una configuración previa del llamado a los endpoints, se utiliza el método correspondiente para ejecutar la acción, justo después de ello se especifica cada uno de los endpoints junto al tipo de dato que recibe. Para recolectar los datos obtenidos y posibles errores se utiliza la desestructuración propia de JavaScript, el cual retorna un status y una data, dentro de data cuenta con otra propiedad el cual es accedido por el nombre de result.

Posteriormente se indica el retorno, está constituido por el status y el result, si son varios datos o posibles arreglos, retorna entities pero si solo es un elemento retorna entitty, si retorna una respuesta bolena entonces solo retorna result, finalmente se indica el tipo de retorno, que es fácilmente identificable por el tipo de dato que retorna toda la función en general. Todo ello es visible en la Figura 3.15 Gateway



*Figura 3.15 Gateway*

Una vez realizado los respectivos gateways, se comenzó de inmediato a crear la clase del archivo con terminación controller.ts, dentro de ellos se define cada uno de las interfaces creadas anteriormente, estas ahora servirán como métodos de clases para poder acceder directamente a su configuración y ejecución del código. Cada una de las funciones recibirá un tipo de dato específico, se llamara una nueva instancia de los respectivos iteradores junto a una nueva instancia del archivo con terminación gateway.ts. Se realizó un bloque de código con las palabras reservadas try y catch para el manejo de errores, si todo el proceso de lógica y llamado a la API es correcto, nunca saldrá del try, pero, si llega a ocurrir un error entrará al catch t ejecutará la instrucción la cual indica un resultado falso, indicando que la operación fallo. El ejemplo antes descripto puede verse en la Figura 3.16 Controlador



*Figura 3.16 Controlador*

Dentro del framework BootstrapVue se inicializa la estructura de un archivo de Vue. Dentro de todos los casos, se utilizan etiquetas especiales provenientes de BoostrapVue como lo pueden ser:

Tabla 3.2 Definición de etiquetas

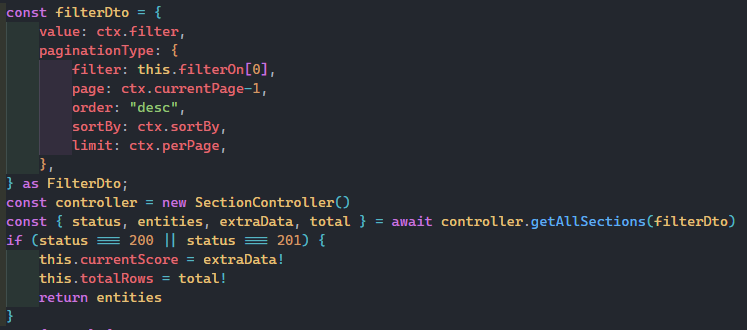
|  |  |  |
| --- | --- | --- |
| **DEFINICION DE ETIQUETAS** | | |
| **Nombre de la etiqueta** | **Función** | |
| b-row | | Se usa para crear 12 filas dentro de un sistema de rejilla, organizando y distribuyendo contenido de manera receptiva y flexible en una página web. |
| b-col | | Se usa para delimitar zonas en específico creadas a partir de b-row |
| b-card | | Se utiliza para crear tarjetas que contienen contenido estructurado, como texto, imágenes o botones. |
| b-form-group | | Se utiliza para agrupar elementos de formulario relacionados |
| b-form-input | | se utiliza para crear campos de entrada de formulario, como campos de texto, áreas de texto, o campos de entrada numérica |
| b-table | | Se utiliza para crear tablas interactivas y responsivas, que muestran datos de manera estructurada |

Además de ello se utiliza principalmente el uso de tablas, siendo utilizados por la etiqueta b-table, esta tabla contiene varios atributos que fueron de ayuda para la correcta visualización de los datos. Algunos de estos atributos son que ayudan a tener un paginador dentro de la tabla, muy útil para una gran cantidad de datos, seleccionar un tipo de filtrado, mostrar condicionalmente un texto si el atributo ítems es nulo o vacío, ordenar de manera ascendente o descendente. Esto puede observarse en la Figura 3.17 Tabla BootstrapVue.



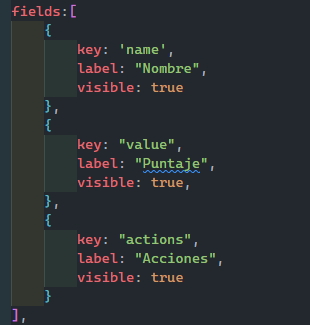
*Figura 3.17 Tabla BoostrapVue*

Dentro de la primera vista principal se iterará una tabla que toma como elementos la llamada a la petición, esta utiliza el controlador previamente creado para llamar a la ejecución a la API correspondiente, en este caso, la respuesta de la API se desestructura en variables constantes que serán almacenadas en variables reactivas para su posterior manipulación esto servirá para obtener cada uno de los elementos requeridos que están registrados dentro de la bases de datos la Figura 3.18 Llamada API muestra el proceso antes mencionado.



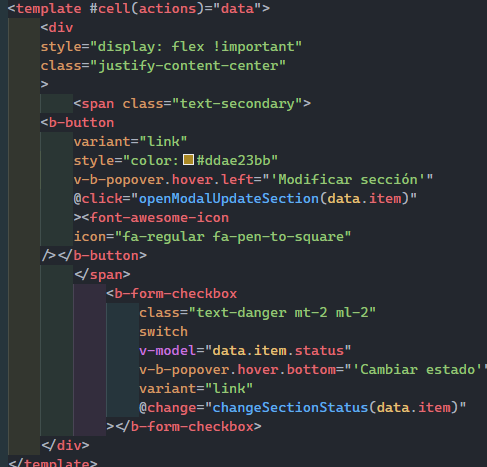
*Figura 3.18 Llamada API*

Más adelante, la tabla requiere de ciertas columnas que actuarán como cabeceras para que los usuarios finales puedan identificar fácilmente la información proporcionada. La estructura de los campos dentro de la tabla se detalla en la Figura 3.19 Cabeceras para la tabla.



*Figura 3.19 Cabeceras para la tabla*

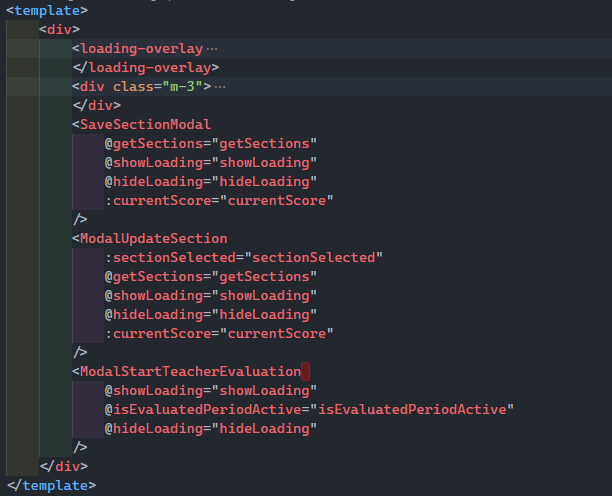
Una vez estos campos estén completados, la etiqueta b-table, empezará a iterar cada uno de los elementos presentes en el arreglo, la respuesta obtenida por medio del llamado a la API. Dentro de esta iteración, dentro de la columna actions, se colocaron funciones respectivas, estas realizan el llamado de abrir un modal para registro o para editar dicha información. Lo antes mencionado está presente en la Figura 3.20 Acciones iteradas.



*Figura 3.20 Acciones iteradas*

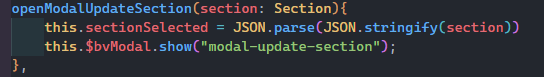
Su funcionamiento consta en la condición si el usuario da un click en el elemento indicado, si este lo hace, disparará el funcionamiento indicando. Estas funciones reciben un parámetro el cual el objeto al cual se está haciendo referencia. Ya que como es una iteración de elementos que forman parte de un arreglo, estos son fácilmente accesibles por la propiedad data.item definida en la llamada de las cabeceras de la columna.

Para el correcto funcionamiento y llamado de lo modals dentro de la vista estos serán llamados por componentes individuales, esto será de gran utilidad al momento de reutilizar código, el cual deberán ser llamados dentro de la vista y esperarán hasta ser activados por medio de su identificador. Estos modals tienen como peculiaridad el recibimiento de parámetros y funciones. Lo antes mencionado puede ser encontrado en la Figura 3.21 Modals en vista.



*Figura 3.21 Modals en vista*

Al hacer clic dentro del elemento iterado y ejecutar la función indicada este acciona una instrucción el cual indica que se muestre el Modal respectivo junto a una variable que es usada como parámetro para el correcto funcionamiento de la visualización de la información dentro del modal. Este es pasado por un parseo a tipo de cadena en JSON y posteriormente parseado a un JSON, Gracias a esto la referencia de memoria original de este dato es perdido haciendo que en este objeto pueda ser libremente modificado o consultado evitando la mutación de variables que es muy ocasional en JavaScript. Esto puede ser visto en la Figura 3.22 Función llamado de un Modal.



*Figura 3.22 Función llamado de un Modal*

Dentro de cada modal se presentará la información indicada según sea el caso, ya sea mostrar información, formularios para realizar un registro o formularios con información cargada para una actualización. Para cada una de estas acciones es necesario colocar una función que las ejecute, es por ello que dentro de cada modal existe un botón para poder cancelar la acción o realizarla, si este último caso, al hacer uso del atributo clave @click se mandará a ejecutar una función.

Para la mayoría de las funciones se especificó el uso de la palabra reservada async indicando que esta función tomará algo de tiempo para su ejecución, para ello antes de realizar una acción se le notificará con una ventana emergente si está seguro de realizar la acción, en caso de ser una respuesta positiva realizará el llamado de la API, de este llamado se desestructurará de la respuesta el estado HTTP de esta misma, si es un estado igual a doscientos emitirá un mensaje indicando que la operación ha sido realizada de manera exitosa. En caso contrario, si la respuesta es diferente a doscientos la operación será cancelada.

Para el uso correcto de las siguientes vistas es necesario contar con un usuario con privilegios de evaluación docente.

La primera funcionalidad en realizarse son las Secciones, dentro de estas secciones se realizaron los repositorios, los casos de uso, gateway y controlador respectivos, así como su vista necesaria dentro de la carpeta adapters/views. Donde dentro de este archivo se coloca una tabla que manda a traer la información de cada una de las secciones registradas en la base de datos, en dado caso, si el arreglo regresado es nulo o contiene una longitud igual a cero, quiere decir que este no cuenta con registros, es por ello que en la tabla aparecerá una leyenda indicando que no hay registros actualmente. La imagen completa de las secciones puede ser vista en la Figura 3.23 Secciones

A screenshot of a computer

Description automatically generated

*Figura 3.23 Secciones*

En la parte superior se colocó un botón el cual ejecutará un modal indicando al usuario que inicie el periodo de evaluación docente, junto a la selección del periodo. Esto dará inicio a una serie de validación que dejará activo la evaluación docente hasta la fecha fin programada. Esto puede observarse en la Figura 3.24 Inicio Evaluación Docente.

A screenshot of a computer

Description automatically generated

*Figura 3.24 Inicio Evaluación Docente*

Dentro de la vista principal, se encuentra la integración de un botón con un icono del símbolo de más. Este botón hace referencia al registro de una nueva sección, al dar clic abrirá un modal. El cual contendrá un formulario con los datos necesarios para que el registro de la aplicación pueda ser llevado sin errores. El formulario cuenta con validaciones indicando si los datos son erróneos o si hace falta llenar algún campo de carácter obligatorio. Mientras todas las condiciones del formulario sean erróneas o los datos no estén completos los botones de registro no se habilitaran hasta que los campos estén correctamente llenados.

Una vez completada toda la información requerida en el formulario se enviara los datos atreves de la llamada de la API por medio del controlador antes creado en TypeScript, esto cuando el usuario haga clic en el botón de registrar accionando toda la lógica para poder colocar la información en uno o varios parámetros que finalmente almacenarán la información dentro de la base de datos como lo es mostrado en la Figura 3.18 Llamada API. La vista antes descrita puede ser vista en la Figura 3.25 Registro de Sección.

A screenshot of a computer

Description automatically generated

*Figura 3.25 Registro de Sección*

Para la modificación de las secciones es utilizado un modal similar al mostrado en la etapa de registro de secciones, con la gran diferencia de que este modal ya cuenta con la información precargada para poder manipularla, este formulario de modificación cuenta con todas las validaciones que contiene el registro. El funcionamiento de cómo es llamado este modal se encurta en la Figura 3.20 Acciones iteradas. Al tener todo el formulario correctamente llenado se podrá llamar a la función de modificar el dato utilizando la llamada al endpoint correspondiente para poder realizar esta acción. Finalmente se tiene la opción de inhabilitar una sección, esto sirve como manera booleana de eliminar un registro, al hacer clic en el icono saldrá una ventana emergente el cual indicará y refirmará la acción que está a punto de hacerse. Si la respuesta es afirmativa se hará la manipulación del estado de esa sección en específico. Para poder habilitarlo de nueva cuenta se tendrá que repetir el mismo procedimiento.

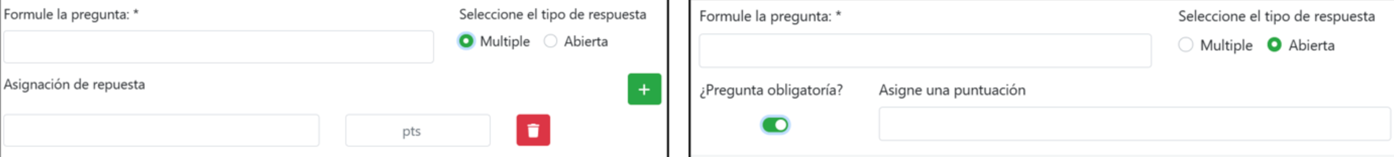
La segunda funcionalidad hecha fueron las preguntas, se realizan los repositorios, los casos de uso, gateway y controlador respectivos, así como su vista necesaria dentro de la carpeta adapters/views. Donde dentro de este archivo se coloca una tabla que manda a traer la información de cada una de las preguntas registradas en la base de datos, esta misma tabla cuenta con las validaciones antes mencionadas en las anteriores tablas. La imagen completa de las secciones puede ser vista en la Figura 3.26 Preguntas

A screenshot of a computer

Description automatically generated

*Figura 3.26 Preguntas*

Para registrar una pregunta se deberá seleccionar el botón verde con el icono de más, cuando se le de clic al botón un modal aparecerá con un formulario. Este formulario indica elegir una sección, sin secciones registrados de manera previa este flujo de registro de preguntas no sería posible. Seguido de formular la pregunta que quiere hacer, en este momento se le indica al usuario que tipo de pregunta será si de opción múltiple o abierta, dependiendo de esta elección el formulario cambiara un poco optando por asignar respuestas con puntos asignados o seleccionar si la pregunta es obligatoria con puntos asignados. Esta demostración puede encontrarse en la figura 3.27 Preguntas Abiertas o Múltiples.



*Figura 3.27 Preguntas Abiertas o Múltiples*

Si el usuario elige las preguntas de opción múltiple este último puede agregar cuantas respuestas le sea posible. En caso contrario si el usuario indica que es una pregunta de opción abierta, tendrá que elegir si esta pregunta es obligatoria, por defecto esta opción está marcada como falsa. Cualquiera de las opciones elegidas tendrá un comportamiento totalmente diferente al momento de que un alumno conteste la evaluación docente. Finalmente, al completar el formulario con las preguntas y respuestas correspondientes, dicho sea, el caso, al hacer clic en el botón de registrar, disparará la función para guardar los datos dentro de la base de datos haciendo persistente la información almacenada. La visualización por completo del formulario de registro de una pregunta se encurta en la Figura 3.28 Registro pregunta.

A screenshot of a computer

Description automatically generated

*Figura 3.28 Registro pregunta*

Para la modificación de las preguntas, estas siguen el mismo flujo que el formulario de registro, solo que ahora conlleva la información precargada gracias a la iteración de los elementos dentro de la tabla y la llamada del modal como lo puede apreciar en la figura 3.14 Acciones iteradas Y 3.16 Función llamado de un Modal. Este contará con el mismo modelo que el registro, cambiado entre pregunta Múltiple a Abierta sin problema, el inconveniente está si el tipo de pregunta es cambiado, ya que la información previa al cambio de tipo de pregunta no se guardará en ese preciso momento de ejecución, para ello se encuentra el botón de cancelar quien ocultará el modal y reinicializara las variables utilizadas para la visualización de la información y gracias a la iteración de los elementos dentro de la tabla al volver abrir el modal de modificación la información aparecerá sin ningún tipo de alteración.

Al momento de modificar una pregunta de opción múltiple, se puede eliminar las respuestas dentro de la modificación, está eliminación es de manera física. Lo que quiere decir que todo rastro de la respuesta en específico será borrado por completo dentro de la base de datos.

El cambio de estado también está presente, como anteriormente se mencionó, es una manera lógica o booleana de eliminar un registro, lo que significa que dentro de la base de datos este registro aún existirá, pero diferentes endpoints que consulten todas las preguntas con estado activo no mostrarán las preguntas que cuenten con el estado como desactivado.

Finalmente se tiene una eliminación física, a diferencia de la anterior y como se explicó anteriormente, es una manera para borrar este registro de manera permanente dentro de la base de datos sin opción a poder recuperarla.

Tanto el cambio de estado como la eliminación por completo de las preguntas serán precedidas por una alerta el cual confirmará que el usuario este de acuerdo con la eliminación lógica o física de dicho dato.

La siguiente funcionalidad a desarrollador es la consulta de los resultados de la evaluación docente. Para ello se creó una vista dentro de la carpeta adapters/views. Al inicio pedirá seleccionar el periodo de evaluación docente que querrá consultar los resultados.

Después de ello se consultará a la base de datos para obtener a cada uno de los docentes su respectiva puntuación obtenida junto a sus detalles. Cabe aclarar que los detalles de la evaluación de cualquier docente no se podrá saber cuál estudiantado realizo dicha evaluación, estas respuestas solo podrán provenir de un tipo de pregunta abierta. La figura 3.29 Resultados evaluación docente muestra lo antes descrito

A screenshot of a computer

Description automatically generated

*Figura 3.29 Resultados evaluación docente*

Al ingresar a los detalles, se redirigirá a una nueva vista, la cual contendrá cada una de las secciones registradas previamente, pero solo serán visibles si estas secciones cuentan con un tipo de pregunta abierta, en dado caso dicha sección no aparecerá. Por lo contrario, si la pregunta es de tipo abierto junto a no es obligatorio, solo aparecerá los datos que se hayan registrados, todos aquellos alumnos que no hayan considerado contestarla, sus respuestas no serán mostradas para mayor legibilidad al usuario final. Al realizar un clic en la sección se desplegará una lista de elementos haciendo referencia a cada una de las preguntas, al hacer clic en cualquiera de esas preguntas mostrará por todas las respuestas que ese docente tenga registradas finalizando la consulta de la evaluación docente. La Figura 3.30 Detalles Evaluación docente muestra lo antes mencionado.

A screenshot of a computer

Description automatically generated

*Figura 3.30 Detalles Evaluación docente*

Para la siguiente funcionalidad, se necesitará realizar con un Privilegio de Estudiante, este puede ser otorgado únicamente realizando una inscripción.

Para resolver la evaluación docente, una estudiante tendrá que iniciar sesión dentro de la aplicación hibrida, dentro de sidebar se verá la opción de ‘Evaluación Docente’, Si existe una evaluación docente activa, el sistema lo hará saber dando la bienvenida a la evaluación docente, en caso contrario se le comunicará al estudiantado que no se cuenta actualmente con una evaluación docente activa. Lo antes mencionado puede ser visto el Figura 2.31 Inicio Evaluación docente alumno.

A screenshot of a computer

Description automatically generated

*Figura 3.31 Inicio Evaluación Docente alumno*

El botón siguiente mostrará una tabla con los docentes que actualmente tenga asignados y podrá elegir si evaluar a todos sus docentes en una sola evaluación o por lo contrario si prefiere realizarlo de manera individual, el sistema está diseñado para cualquier opción elegida por el estudiantado. Esto puede verse en la Figura 3.32 Selección de docentes.

A screenshot of a computer

Description automatically generated

*Figura 3.32 Selección de docentes*

Una vez seleccionado el o los docentes iniciará la evaluación docente. Este está conformado y dividido por todas las secciones previamente registradas, así como de las posibles preguntas con sus respectivas respuestas relacionadas a esa sección. Si en la creación de la pregunta fue de opción múltiple el alumno tendrá que contestar de manera obligatoria una de las opciones mostradas. Cuando haga esto una barra de progreso llevará el conteo total de evaluación respondida hasta ese momento, indicando cuanto porcentaje ha contestado de toda la evaluación. Esto puede verse en la figura 3.33 Preguntas Múltiple.

A screenshot of a computer

Description automatically generated

*Figura 3.33 Preguntas Múltiple*

Una vez contestado todos los rubros de la sección actual, podrá ir a la siguiente sección, dentro de la cual se puede ver como el comportamiento de lo antes visto cambio al incluir un tipo de pregunta de opción abierta y marcarla como obligatoria.

Este contará con un área de texto en la cual un estudiantado podrá escribir sin limitaciones. Si este rubro permanece sin llenar la barra de progreso no subirá, por ende la evaluación no podrá ser concluida. Una vez se haya contestado este rubro, la barra de progreso llegará a cien por ciento. Tanto las secciones como las diferentes preguntas con sus posibles respuestas están sujetas al usuario final, por lo que el sistema está diseñado y pensado para poder mantener cada una de las respuestas previamente almacenadas por si algún estudiantado prefiere cambiar de opinión en una respuesta sin perder su progreso. La respuesta de type abierta puede verse en la figura 3.34 Pregunta Abierta

A screenshot of a computer

Description automatically generated

*Figura 3.34 Preguntas Abierta*

Al tener toda la evaluación docente terminada en su totalidad, el botón de finalizar se habilitará, mostrando un mensaje de alerta indicando que el resultado no podrá ser modificado dando oportunidad al usuario de reiterar si las respuestas que quiere enviar son las apropiadas, terminando así la evaluación docente.

## Control

**Reuniones con el cliente**

Periódicamente se llevó a cabo reuniones con el cliente para poder reiterar si los visto con los mockups es correcto y el flujo que este presenta es útil para todo el marco de trabajo que ellos consideran útil. Dentro de estas reuniones el cliente pudo dar su punto de vista y dar una retroalimentación activa de la evaluación docente. Surgieron pequeños cambios en una primera instancia, pero, todos los cambios mencionados fueron arreglados e incluso el responsable de proyecto también pudo aportar su opinión la cual fue del agrado del cliente.

**Cumplimiento con el Diagrama de Gantt**

Con el fin de dar seguimiento a lo estipulado en el diagrama de Gantt de acuerdo con las actividades de las distintas fases mencionadas, se monitoreaba que las tareas fueran realizadas en las fechas estimadas que se tenían, con la finalidad de conocer si el proyecto mantenía un buen avance o que este tuviera un defecto de retraso.

El resultado fue el esperado, no se detectó ninguna incidencia potencial que retrasará la entrega del módulo por completo.

## Cierre

Al finalizar el módulo que se tenía esperado se tuvo una reunión con el asesor empresarial, en dicha reunión se mostró el flujo de la aplicación de acuerdo con cada rol al que pertenece, se aseguró que las funcionalidades estuvieran definidas correctamente y estuvieran presentes como en lo dicho con el documento formal de requisitos. También se obtuvo una reunión con el cliente el cual dio su punto de vista y acordó estar de acuerdo con el resultado mostrado.

Al ser un proyecto que cuenta con más módulo y con el cliente probando el sistema de manera activa para un uso casi de inmediato, se dio por terminado con el módulo.

|  |
| --- |
| CONCLUSIONES |

## Cumplimiento de objetivos

De acuerdo con los objetivos específicos y al diagrama de Gantt se pudo llevar a cabo cada una de las metas establecidas. Se cumplió en tiempo y forma cada una de las fases acordadas dentro del diagrama. Cada Objetivo específico se desarrollo por completo.

* Se adjunto nuevas tablas en la base de datos existente, se rediseñaron relaciones para que el módulo de evaluación docente funcionara y no tuviera conflictos.
* Se desarrollo cada una de las funcionalidades indispensables para que todo el módulo funcionara, se crearon sus respectivas API’s junto a la documentación del propio código
* Dentro de la aplicación existente se crearon y se asignaron nuevos privilegios a los distintos roles que los usuarios utilizan para poder hacer uso de toda la funcionalidad del módulo.
* La implementación de las pantallas fue basada a prototipos que se le fueron entregados al cliente, posteriormente su flujo de datos fue validado por el cliente y gracias al responsable del proyecto las vistas correspondientes fueron mejoradas para facilitar la experiencia de usuario.
* Se implemento cada una de las funcionalidades acordes al módulo, estas fueron gestión de secciones, preguntas y consulta de las respuestas de la evaluación docente.
* El módulo por completo fue sometido a pruebas por el equipo de desarrollo y por agentes de QA, además la funcionalidad se encuentra disponible para ser probada en su totalidad dentro del servidor del cliente.

## Resultados

Como resultado del desarrollo, se obtuvieron las funcionalidades acordadas de la evaluación docente por el cliente dentro del sistema actual de la Universidad de Ciencias Jurídicas del Estado de Morelos.

Todos los objetivos específicos y junto al objetivo general fueron cumplidos acorde a lo acordado y solicitado inicialmente por el cliente.

## Contribuciones

Dentro del proyecto se utilizaron nuevas tecnologías, antes de que iniciara el proyecto, el sistema de como evaluar a sus docentes era totalmente obsoleta y difícil de manejar, por lo que la implementación de las tecnologías de BootstrapVue y la optimizada carga de los servicios de Spring son de gran ayuda para mejorar el rendimiento de la aplicación junto a una fácil implementación de cambios.

El proyecto y en especial el módulo desarrollado ocupo de conocimientos nuevos y de un entendimiento mayor a cómo funciona los ciclos de vida dentro del lenguaje de programación y de las herramientas utilizadas. Junto a ello se manejó ampliamente cada uno de los conflictos desarrollados por las distintas versiones del proyecto, teniendo un mejor control y entendimiento del control de versiones Git.

En general, el desarrollo del módulo de evaluación docente contribuyo a mejorar la lógica de programación y al cómo solucionar un problema de distintas formas.

|  |
| --- |
| **REFERENCIAS** |

Attardi, J. (04 de 10 de 2022). *blog.logrocket.com*. Obtenido de BlogRocket: https://blog.logrocket.com/understanding-typescripts-benefits-pitfalls/

Basalo, A. (14 de 08 de 2014). *desarrolloweb.com*. Obtenido de desarrolloweb.com: https://desarrolloweb.com/articulos/que-es- angularjsdescripcion-framework-javascript-conceptos.html

Blandino, G. (1 de 4 de 2023). *Pixarprinting*. Obtenido de Pixarprinting: https://www.pixartprinting.es/blog/figma-que-es/

Borges, S. (26 de 09 de 2015). *infranetworking*. Obtenido de infranetworking: https://blog.infranetworking.com/5-razones-por-las-cuales-debes-usar-postgresql/

Burin, A. (s.f.). *teclab.edu*. Obtenido de teclab.edu: https://www.teclab.edu.ar/tipos-de-arquitecturas-de-software-cuales-hay-y-en-que-se-diferencian/

Camarena Sagredo, J. G., Trueba Espinosa, A., Martínez Reyes, M., & López García, M. d. (2012). *Automatización de la codificación del patrón modelo vista controlador (mvc) en proyectos orientados a la Web.* Toluca, México.

Cambarieri, M. D. (2020). *Implementación de una arquitectura de software guiada por el dominio.* Sociedad Argentina de Informática e Investigación Operativa.

Eze, P. E. (2 de 09 de 2022). *logRocket*. Obtenido de logRocket: https://blog.logrocket.com/getting-started-bootstrapvue/

Fernández, E. C. (25 de 10 de 2022). *tokioSchool*. Obtenido de tokioSchool: https://www.tokioschool.com/noticias/spring-boot/#:~:text=Spring%20Boot%20cuenta%20con%20una,back%2Dend%20de%20forma%20independiente.

Galina, P. (21 de 05 de 2021). *iebschool*. Obtenido de iebschool: https://www.iebschool.com/blog/herramientas-prototipado-analitica-usabilidad/

Gómez, J. C. (2020). *Taxonomía de los modelos y metodologías de desarrollo de software más utilizados.* México.

Ivakhnenko, M. (18 de 08 de 2018). *Dzone*. Obtenido de Dzone: https://dzone.com/articles/the-good-and-the-bad-of-java-programming

Jesus. (11 de 10 de 2022). *Dongee*. Obtenido de https://www.dongee.com/tutoriales/que-es-vue-js-para-que-sirve/

José H. Canós, Patricio Letelier y Ma Carmen Penadés. (s.f.). *Métodologías Ágiles en el Desarrollo de Software.* Valencia : DSIC -Universidad Politécnica de Valencia.

Julen Astigarraga, V. C.-A. (2022). *¡Se puede entender cómo funcionan Git y GitHub!* Madrid, España: Departamento de Ciencias de la Vida, Universidad de Alcalá. 28805 Alcalá de Henares,.

Martín, E. (2023). *grupocibernos*. Obtenido de grupocibernos: https://www.grupocibernos.com/blog/7-aplicaciones-de-comunicacion-y-trabajo-en-equipo

Mozilla. (7 de 3 de 2024). *Mozilla*. Obtenido de Mozilla: https://developer.mozilla.org/en-US/docs/Learn/Tools\_and\_testing/Client-side\_JavaScript\_frameworks/Introduction

Schwaber, K., & Sutherland, J. (2020). *La Guía Scrum.*

Spring. (2023). *Spring*. Obtenido de https://spring.io/projects/spring-boot

TEAM, O. (28 de 10 de 2023). Obtenido de https://optimizdba.com/the-ultimate-guide-to-database-optimization-tips-and-best-practices/

Tiobe. (03 de 2024). *tiobe*. Obtenido de tiobe: https://www.tiobe.com/tiobe-index/

Trucoteam. (2024). *trucoteca*. Obtenido de trucoteca: https://trucoteca.com/en/que-ventajas-ofrece-intellij-idea/

VisualStudio. (28 de 02 de 2024). *Code Visual Studio*. Obtenido de https://code.visualstudio.com/Docs/editor/whyvscode

Weisheim, R. (03 de 12 de 2023). *Hostinger*. Obtenido de Hostinger: https://www.hostinger.mx/tutoriales/boceto-pagina-web

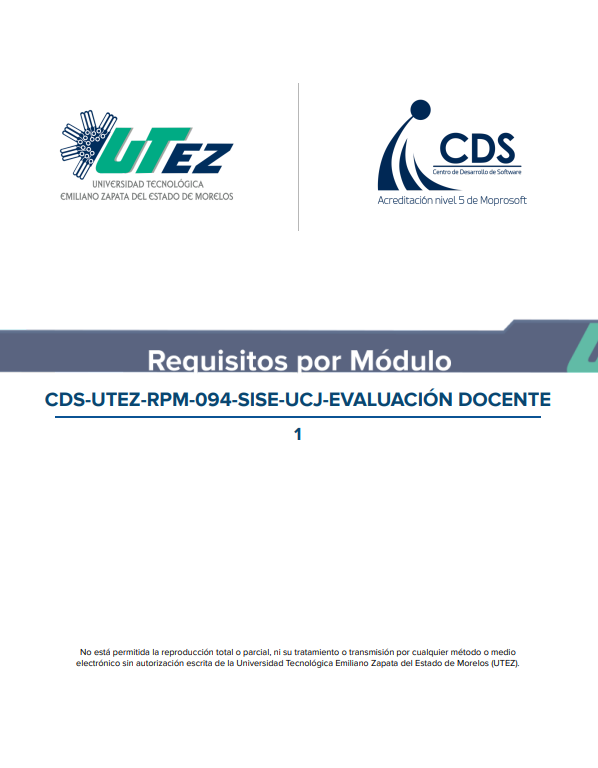
|  |
| --- |
| **ANEXOS** |

**ANEXO A. NOMBRE DEL ANEXO**

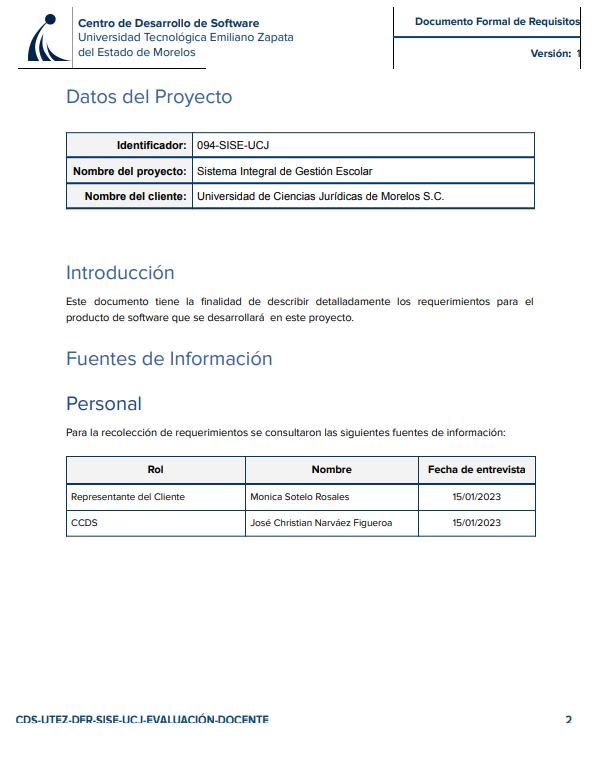
A graph with green squares

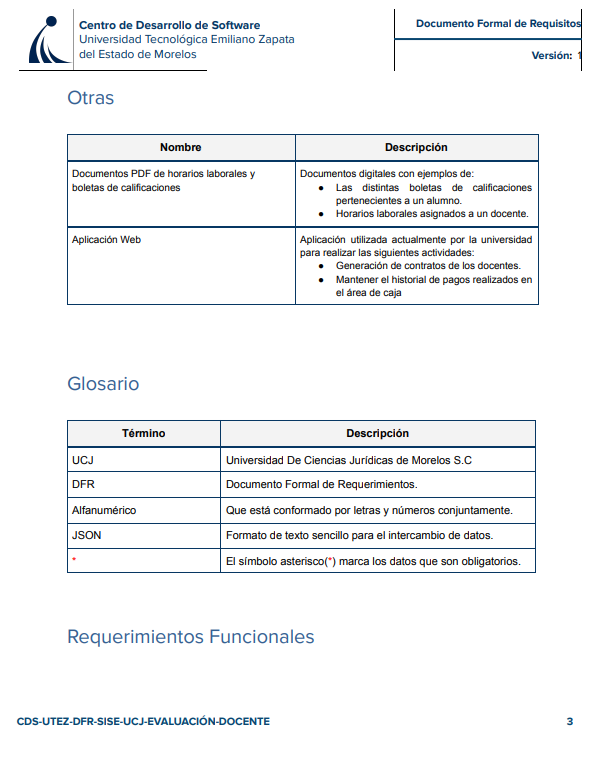
Description automatically generated

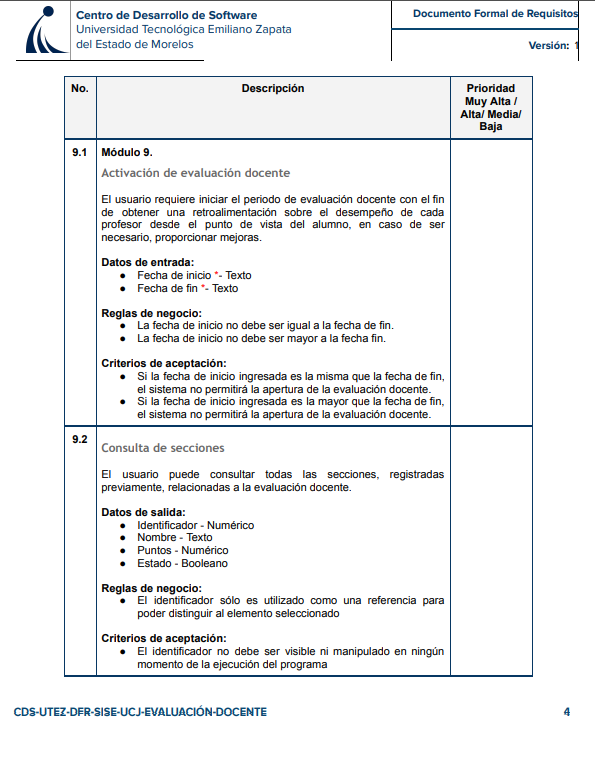
**ANEXO B. DOCUMENTO FORMAL DE REQUERIMIENTOS**

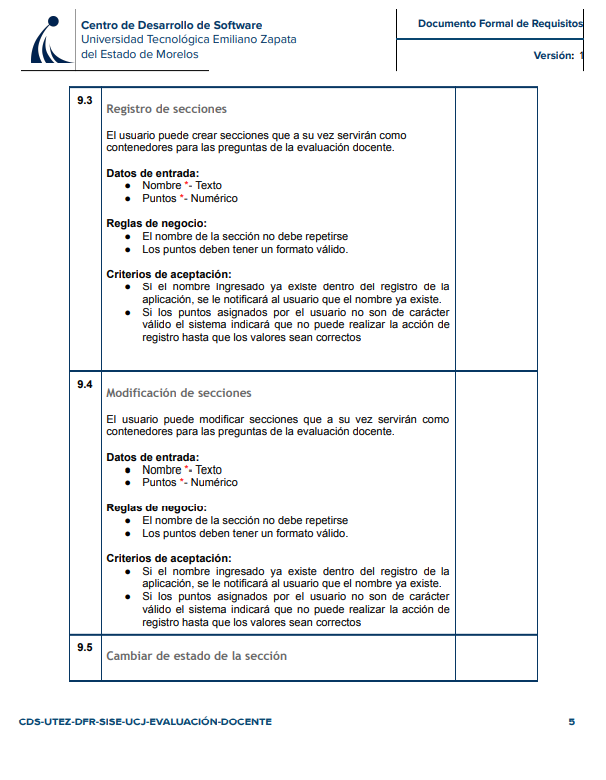
****

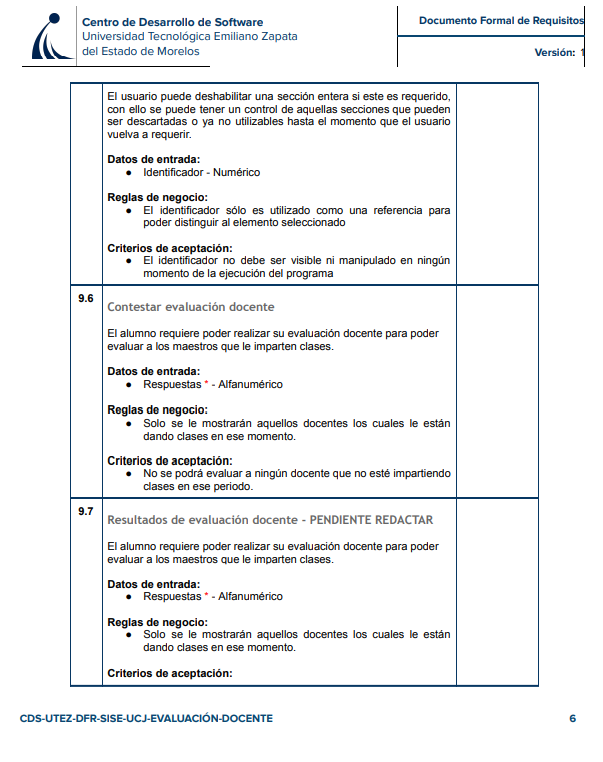
****

****

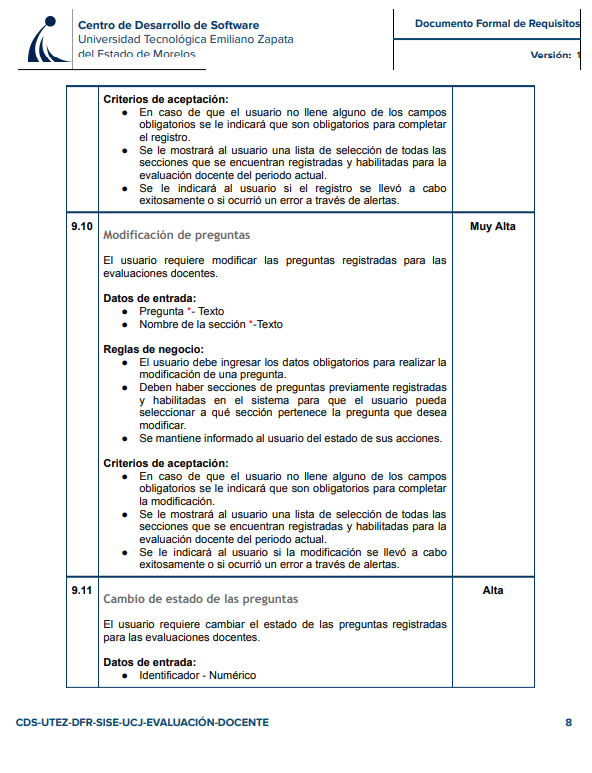
****

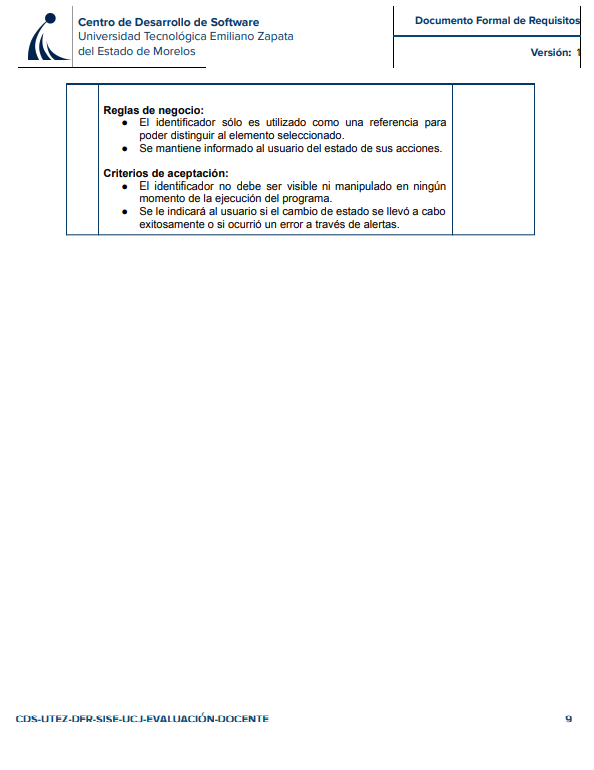
****

****

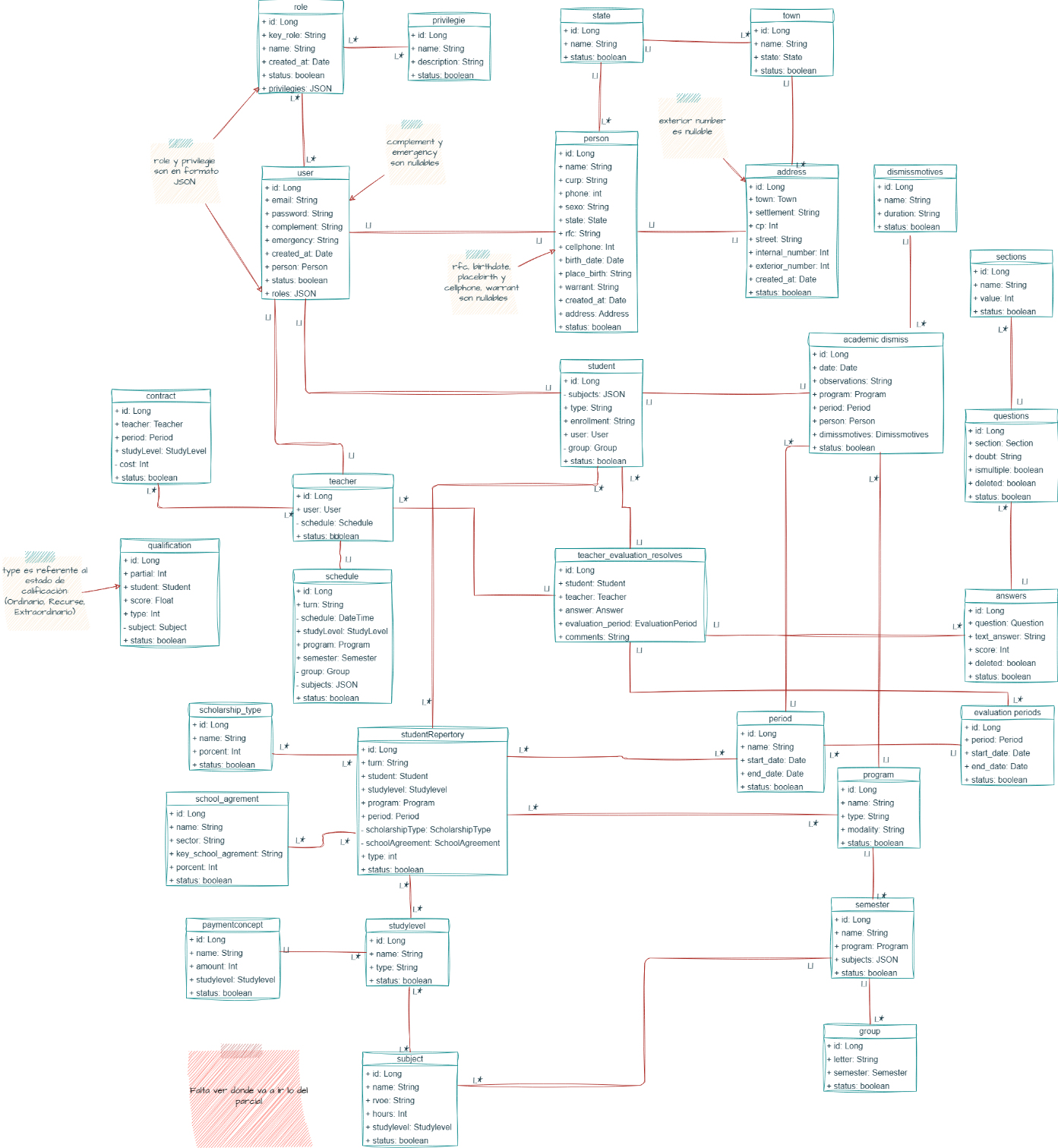
****

****

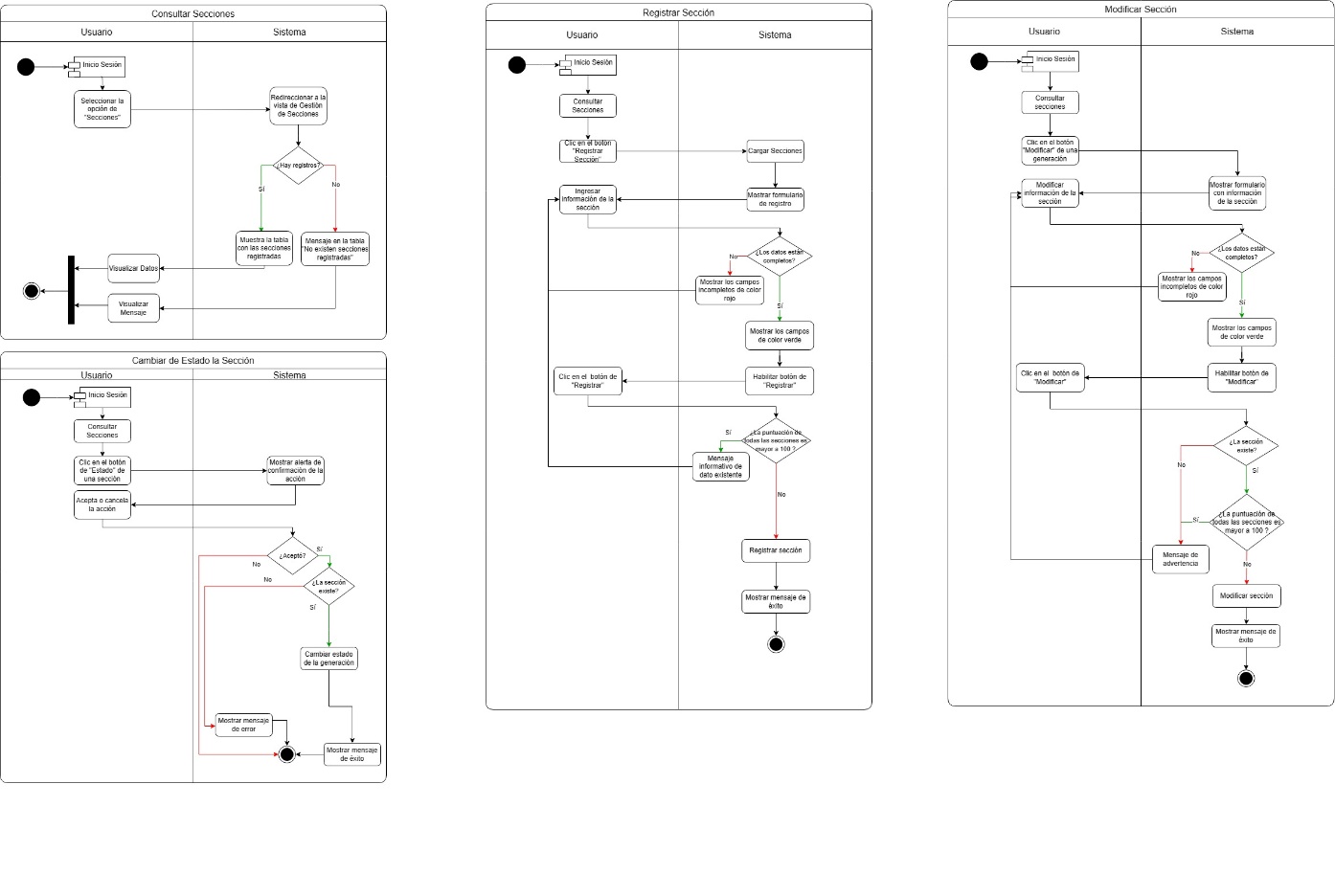
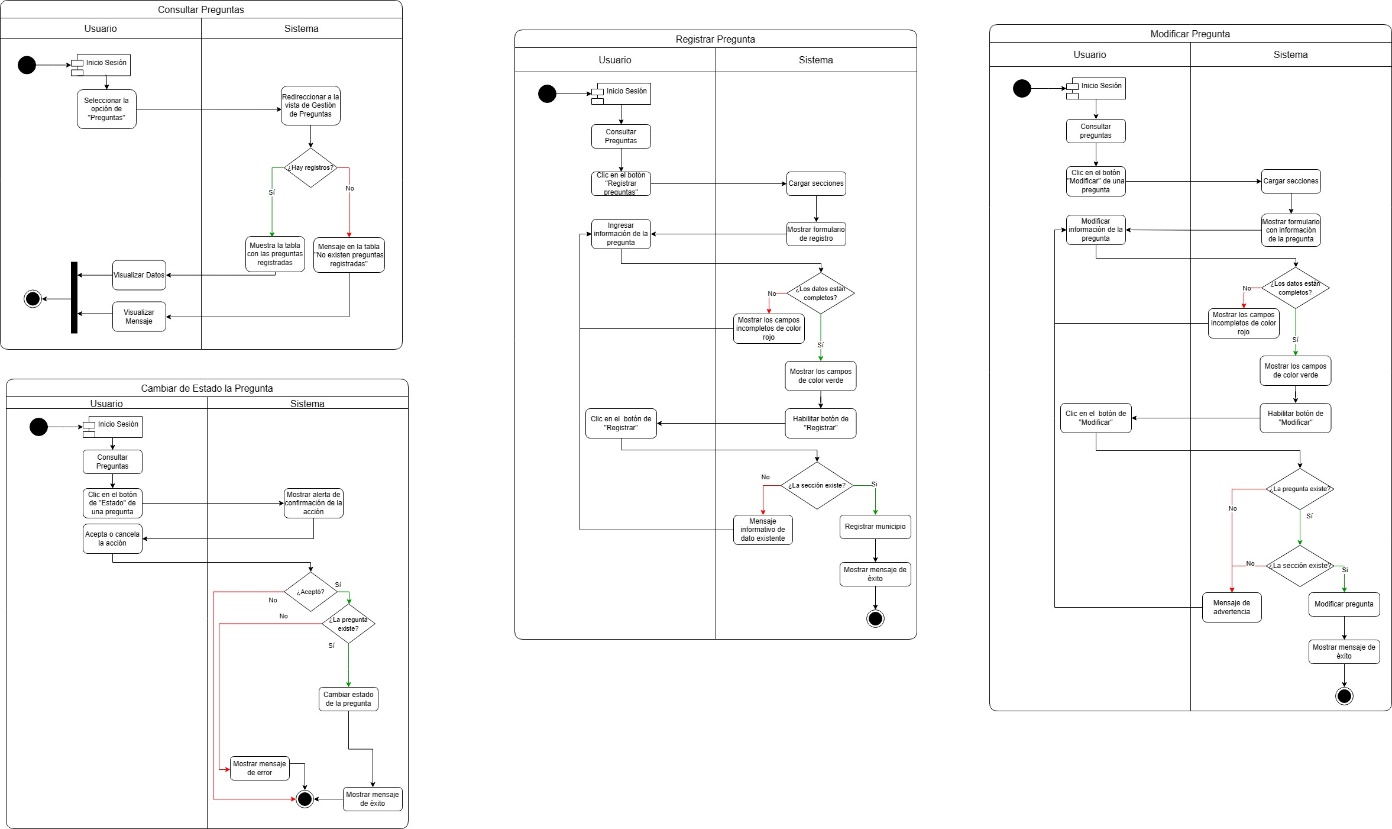
****

****

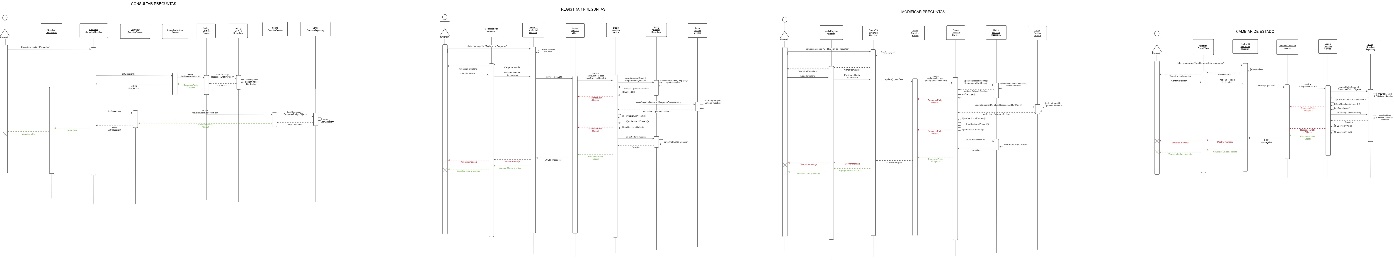
**ANEXO C. DIAGRAMA DE CLASES**

****

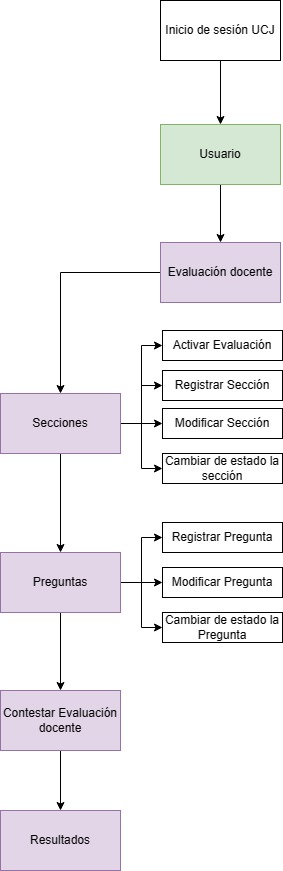
**ANEXO D. DIAGRAMA DE ACTIVIDADES**

****

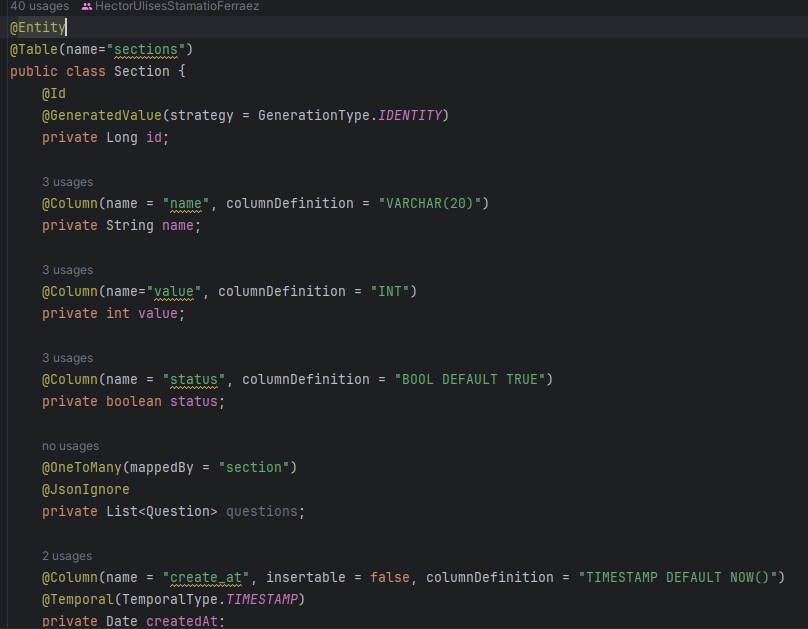
**ANEXO E. DIAGRAMA DE SECUENCIA**

****

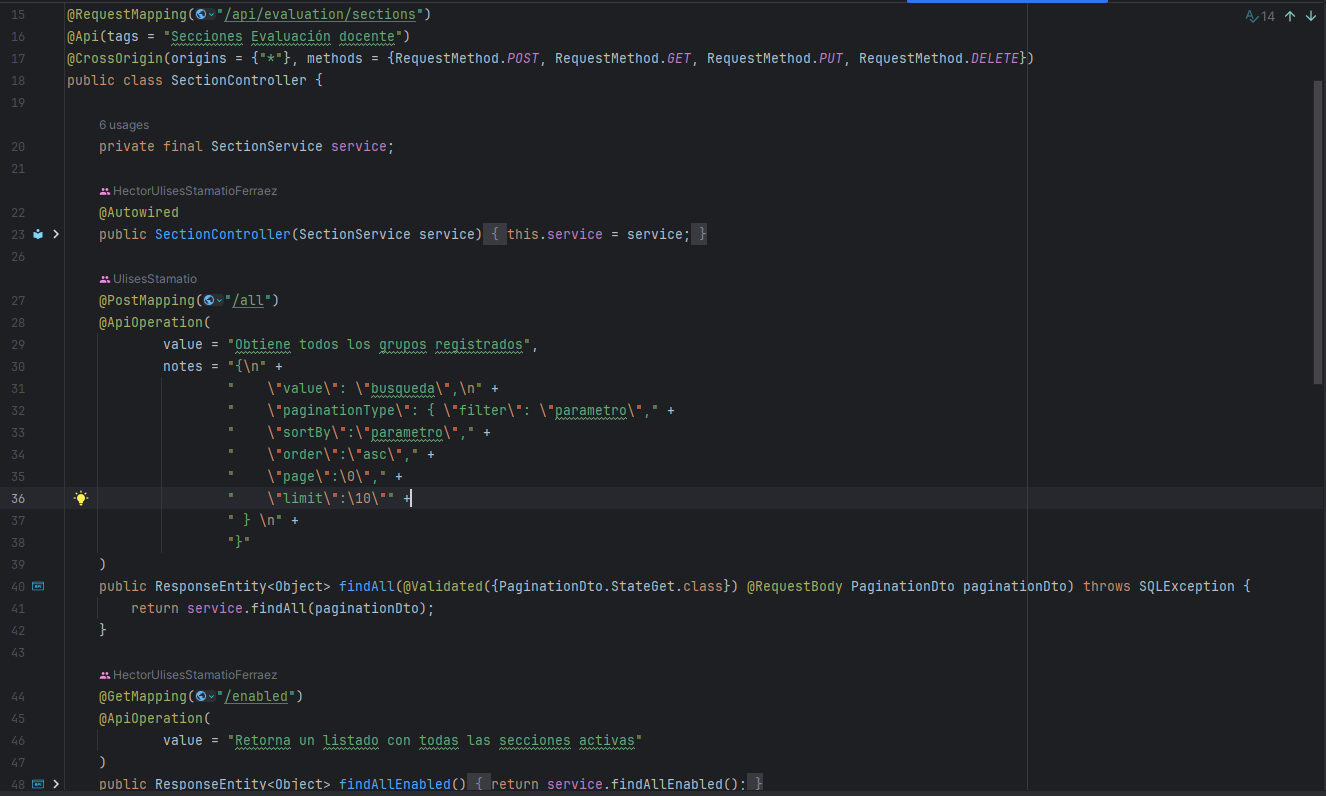
**ANEXO F. ÁRBOL DE NAVEGABILIDAD**

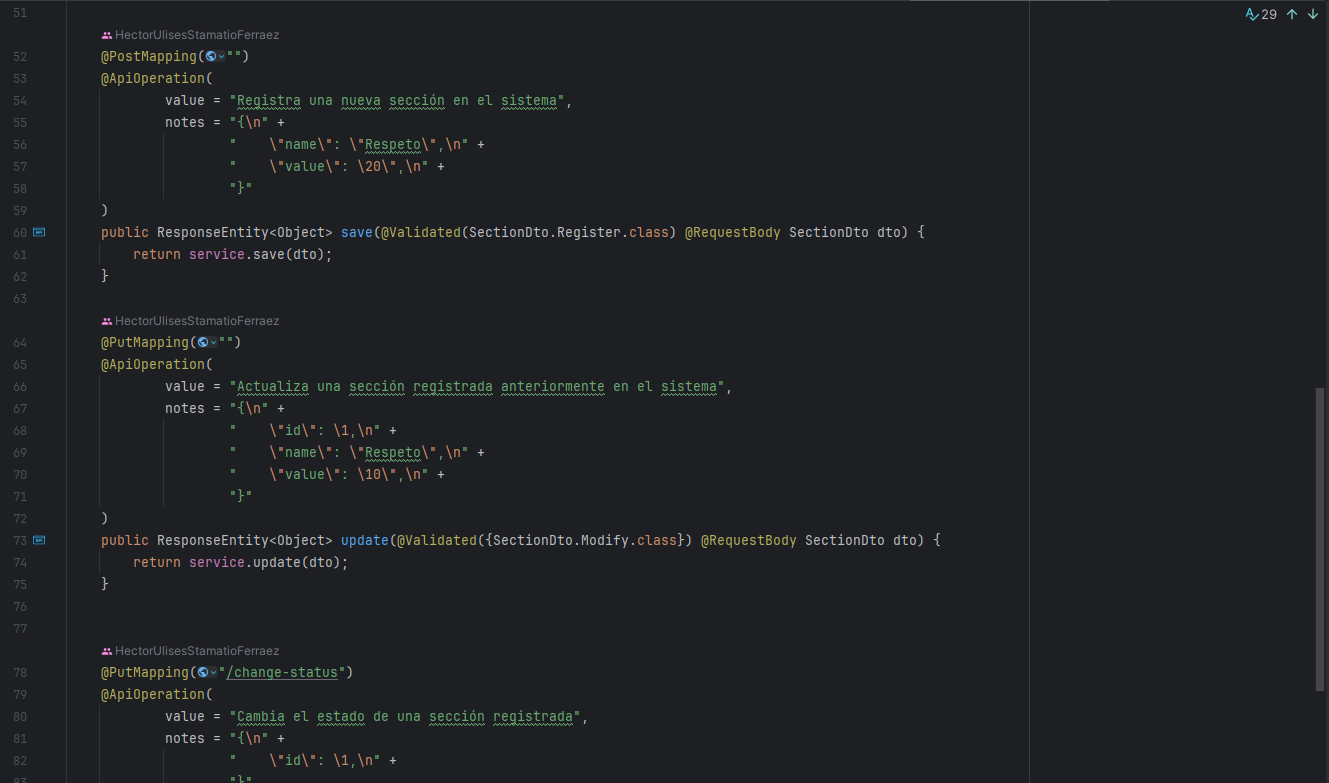
****

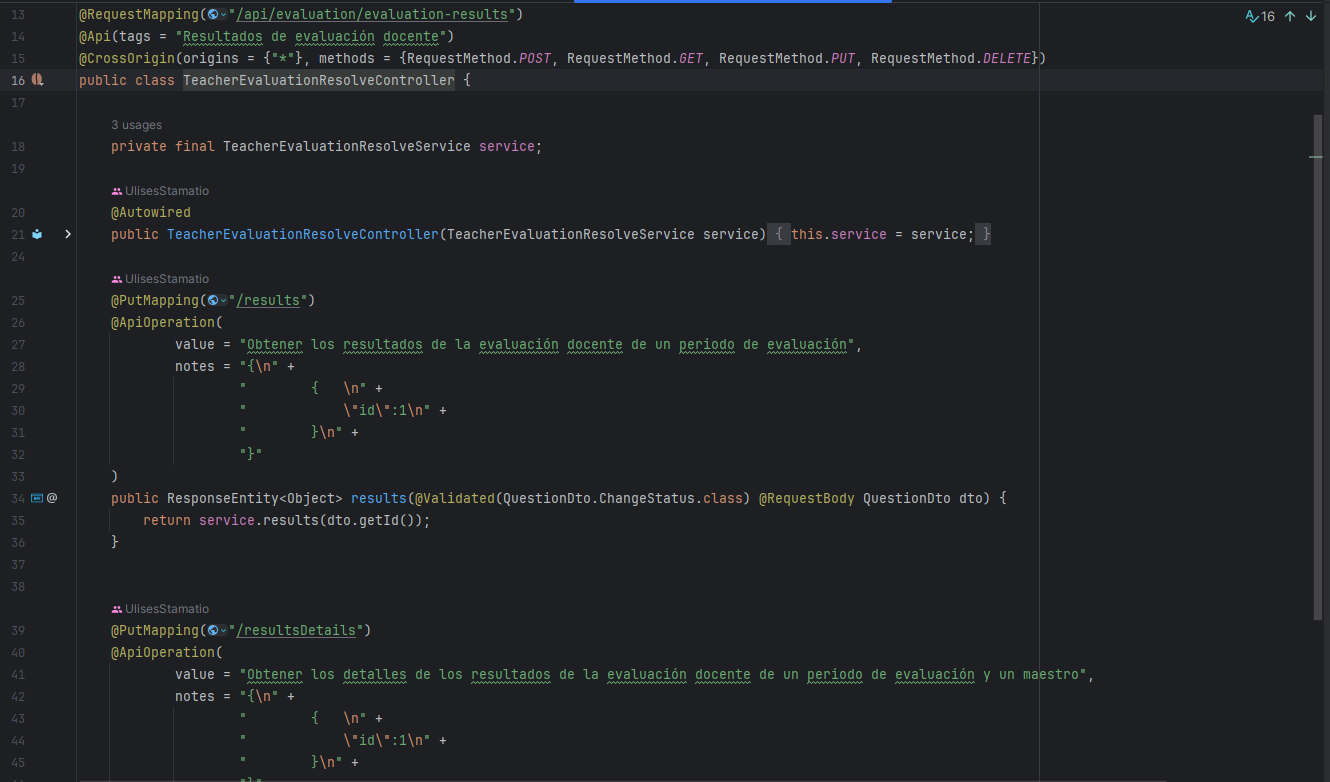
**ANEXO G. ENTIDADES**

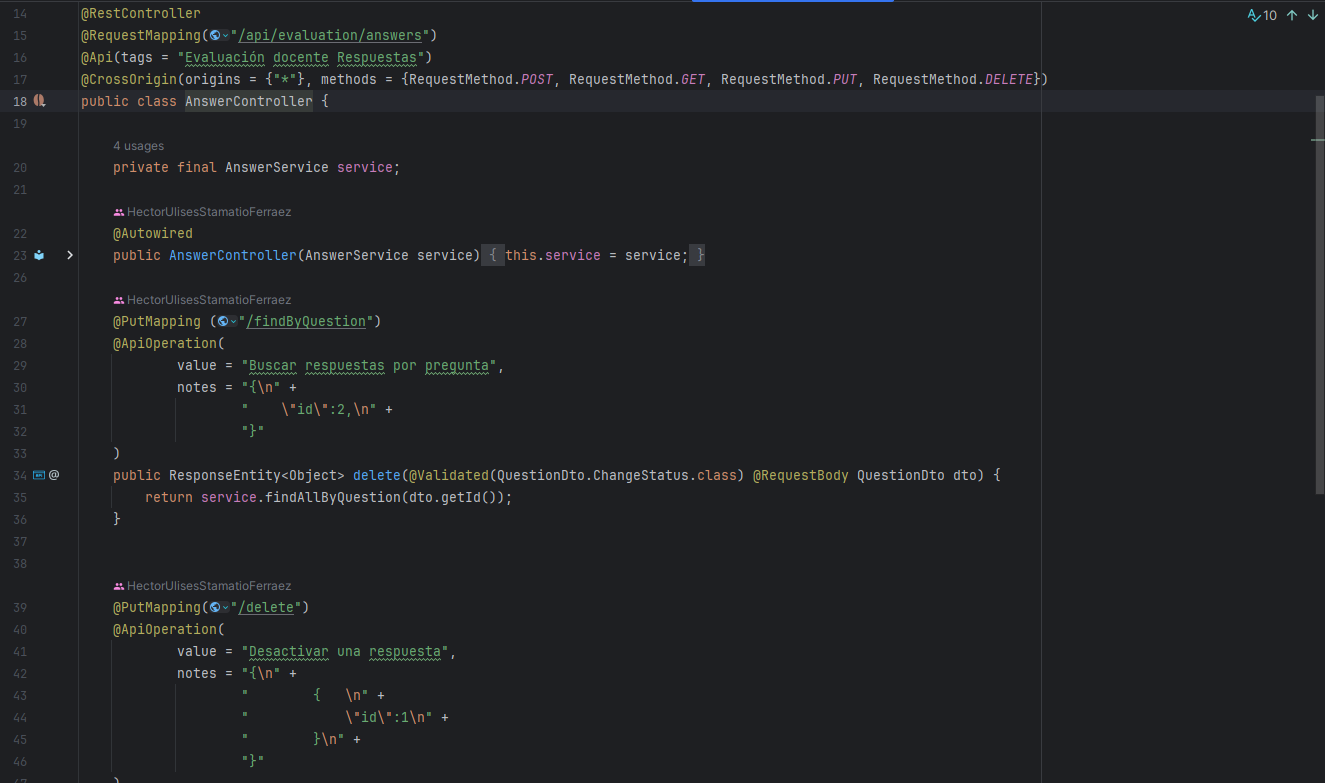
****

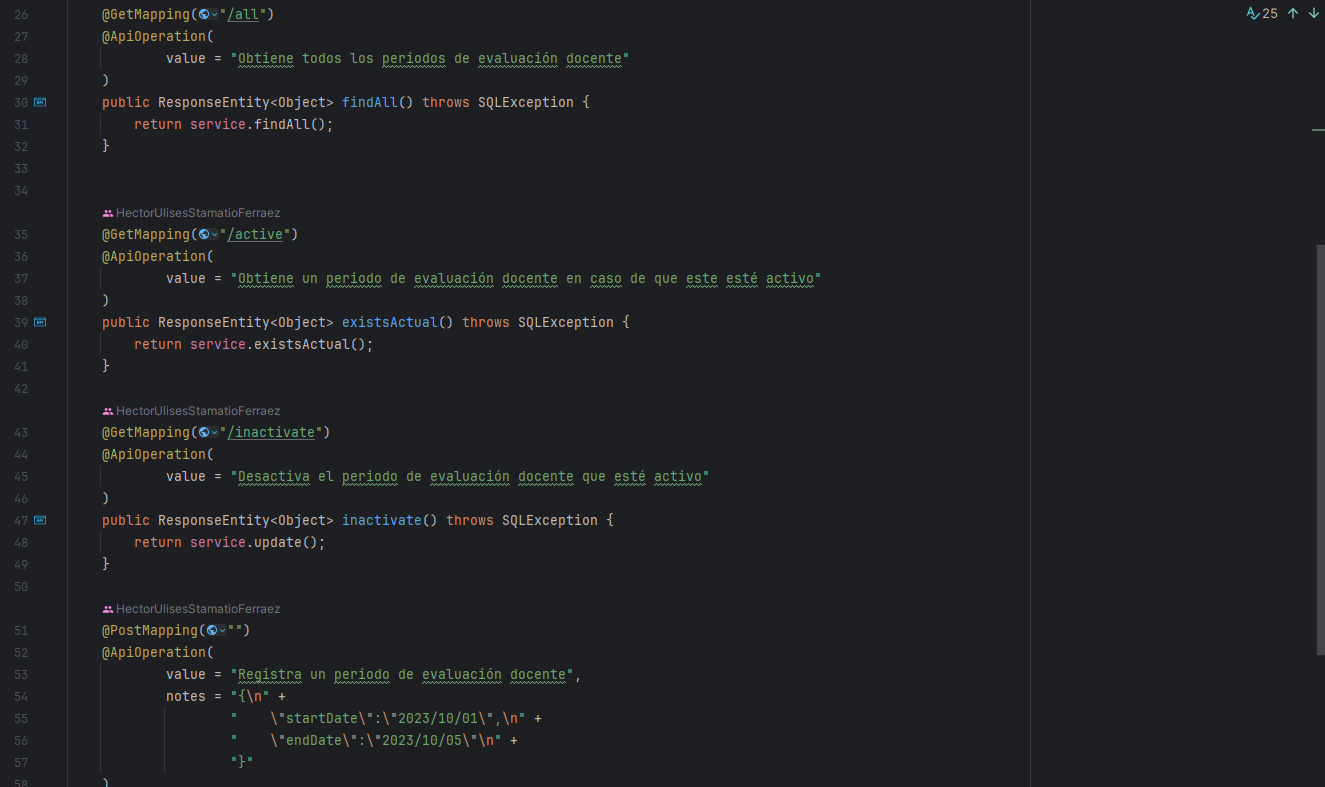
**ANEXO H. ENDPOINTS**

****

****

****

****

****

**ANEXO I. REPOSITORIO**

