UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

Sistema de Gestión de Tutorías: Aplicación Web

Pablo David Llanes Páramo

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito

para la obtención del título de

INGENIERO EN CIENCIAS DE LA COMPUTACIÓN

Quito, 24 de diciembre del 2020

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

Sistema de Gestión de Tutorías: Aplicación Web

Pablo David Llanes Páramo

Nombre del profesor, Título académico Daniel Riofrío, Ph.D.

Quito, 24 de diciembre del 2020

© DERECHOS DE AUTOR

Nombres y apellidos:	Pablo David Llanes Páramo
Código:	00133203
Cédula de identidad:	1717546244
Lugar y fecha:	Quito, 24 de diciembre del 2020

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

En el presente documento se describe el proceso de desarrollo y la implementación de una página web utilizando la metodología de trabajo ágil SCRUM. Este es un caso de estudio independiente relacionado a un sistema de gestión de tutorías para el Colegio de Ciencias e Ingenierías de la Universidad San Francisco de Quito USFQ y surge ante la necesidad de reemplazar el sistema actual, el cual es poco amigable con los usuarios y no mantiene un registro permanente de las actividades realizadas. Por este motivo como objetivo principal se planteó implementar un sistema digital aislado, el cual permita dar seguimiento a los estudiantes, crear reuniones y generar reportes de manera sencilla.

Después de casi cuatro meses de trabajo y cinco fases de desarrollo se logró implementar una página web funcional, la cual recopila los requerimientos más importantes de los clientes y comparte la misma información con una aplicación móvil. Esto permite concluir que la metodología de trabajo utilizada reduce los plazos de entrega de un producto y permite identificar nuevas necesidades que no fueron pensadas en un inicio. Finalmente, en vista a que este es un proyecto en desarrollo, a futuro se necesitan realizar pruebas para detectar y corregir errores con anticipación, así como también implementar nuevas funcionalidades que mejoren la experiencia de los usuarios.

Palabras clave: Metodología Scrum, Gestión de proyectos ágiles, Desarrollo web, JavaScript, HTML, CSS, Bootstrap, Node JS, Handlebars, MySQL, Bases de datos, API, Endpoints, Tutorías, Sistema de Gestión de Tutorías, Acreditación ABET, ABET

ABSTRACT

The following document describes the development process for a web application using the Scrum Methodology. This is an independent case study that is related with a Tutoring Management System for Universidad San Francisco de Quito USFQ, in particular with its school of engineering "Colegio de Ciencias e Ingenierías". The idea started with a request made by some authorities who wanted to replace the current system given that it is not user friendly and hasn't been able to register permanent data since its creation. For these reasons the main goal was to implement an application that allowed users to create meetings, follow up on students, and generate reports easily.

After four months of work and five development sprints, a functional web application was created, and it includes the main requirements from the clients. Likewise, it shares information with a mobile application for the system given that they have the same database. With these results it can be concluded that the working methodology shortened the project's development time and allowed the team to identify new requirements that were not planned at the beginning of the process. Furthermore, given that this application is still in its development phase more testing is necessary to find issues preemptively and fix them as soon as possible. Finally, in the future new enhancements need to be added to improve the overall user experience.

Key words: Scrum methodology, managing agile projects, Web development, JavaScript, HTML, CSS, Bootstrap, Node JS, Handlebars, MySQL, Database Systems, API, Endpoints, Tutoring, Tutoring Management System, ABET accreditation, ABET

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Uno de los objetivos actuales del Colegio de Ciencias e Ingeniería (Politécnico) de la Universidad San Francisco de Quito (USFQ) es llevar un control sobre el rendimiento académico de todos los estudiantes para ir mejorando continuamente el nivel de educación y adicionalmente cumplir con normativas nacionales e internacionales. Este control se lleva a cabo mediante la asignación de un tutor (profesor) a un estudiante para que pueda realizar un seguimiento de su carrera. Por otro lado, los decanos buscan tener reportes sobre las reuniones realizadas a lo largo de cada semestre para en un futuro lograr obtener la certificación ABET para todas las carreras del Politécnico.

Actualmente, el Politécnico cuenta con un sistema de seguimiento poco amigable con los usuarios, el cual no facilita la recolección de información, pues una vez que un profesor agenda una reunión no se genera un registro. Asimismo, el sistema actual es muy poco accesible ya que para encontrar el servicio los usuarios deben navegar por varias páginas y seguir varios pasos para agendar las reuniones respectivas. Otro aspecto importante es que existe un único método de comunicación con los estudiantes mediante el correo electrónico de la USFQ, lo que en muchos casos es un impedimento para realizar las reuniones ya que muchos estudiantes no utilizan el correo institucional.

En este contexto, los estudiantes no tienen conocimiento sobre su tutor asignado y los beneficios de poder contar con dicha persona. De la misma manera, existen estudiantes que no se sienten cómodos con el tutor asignado por lo que al tener un problema no pueden obtener la información para resolverlo. En términos generales, se puede decir que existe muy poca iniciativa tanto de los estudiantes como de los profesores para generar reuniones y hacer el seguimiento deseado.

En este contexto, se puede argumentar que existe la necesidad del desarrollo de una plataforma tecnológica que solucione los problemas encontrados por parte de decanos, profesores y estudiantes que pertenecen al Politécnico. Por este motivo se ha planteado la creación de una página web y una aplicación móvil, las cuales se ajusten a las necesidades de los usuarios.

Para las autoridades del Politécnico, los decanos, se busca realizar reportes de una manera sencilla mediante un registro de las reuniones y sus diferentes estados. En cuanto a los profesores se busca brindar varias formas de comunicación con los estudiantes, además de contar con una lista de los estudiantes asignados y toda la información relevante de los mismos para poder realizar un seguimiento fácil (gpa, periodo de admisión, tipo de admisión, etc.). Finalmente, para los estudiantes se busca facilitar la comunicación con su tutor asignado y dar la opción de pedir un cambio de tutor en caso de ser requerido.

La metodología que se usa en el desarrollo de este proyecto es SCRUM, la cual se basa en entregas regulares y de duración fija (2 semanas). A estas fases cortas de trabajo se les conocen como Sprints y son de gran utilidad para desarrollar el producto final continuamente mediante la retroalimentación de los clientes y la experiencia del equipo de desarrollo. El uso de metodología ágiles en el desarrollo de proyectos de desarrollo de software ayuda a reducir el tiempo de entrega y cumplirlos exitosamente. Asimismo, la satisfacción del cliente siempre será una prioridad, por lo cual se lo involucra a lo largo del desarrollo de todo el proyecto para que el resultado final sea un producto de calidad. Por otro lado, el trabajo en equipo ayuda a que la creatividad e innovación se desarrolle en todos los miembros y se tenga una mejor organización de cada funcionalidad a desarrollar (Abellán, 2020).

Siguiendo esta metodología de trabajo para el desarrollo de este proyecto se decidió realizar reuniones cada dos semanas y bajo los siguientes roles. En primer lugar, como los clientes interesados del proyecto se cuenta con la presencia de César Zambrano, Eduardo Alba,

y Fabricio Yépez, quienes son decanos del Politécnico. Por otro lado, se cuenta con Daniel Riofrío quien ocupó el rol de Scrum Master o mediador entre los clientes y el equipo de desarrollo durante las reuniones realizadas. Finalmente, el equipo de desarrollo quedó conformado por Pablo Llanes y Andrea Porras quienes implementaron las aplicaciones Web y Móvil respectivamente. En el presente documento se describe la información relacionada al desarrollo de la aplicación Web como un caso de estudio independiente.

Objetivos Generales y Específicos

Objetivo General.

1. Implementar una aplicación web independiente al sistema actual de la universidad, la cual permita generar reuniones entre profesores y estudiantes, gestionarlas, y mantener un registro actualizado y permanente de las mismas.

Objetivos Específicos.

1. Mejorar la comunicación entre los diferentes agentes de la universidad, especialmente entre los estudiantes y profesores.
2. Facilitar el acceso a la información del estudiante y su progreso durante la carrera.
3. Facilitar la recolección de información y la generación de reportes para las autoridades del politécnico.
4. Identificar los riesgos presentes en la estructura de la aplicación web y el ingreso de datos.
5. Realizar una discusión sobre la metodología de trabajo, las tecnologías usadas para el desarrollo del proyecto, e indicar los pasos a seguir a futuro.

DESARROLLO

Progreso de la Metodología de Trabajo

En el transcurso del tiempo se ha concluido con los cinco primeros Sprints del proyecto, y se ha logrado crear una aplicación funcional con los requerimientos más importantes de los clientes. El desarrollo del proyecto se ha realizado siguiendo la planificación inicial, sin embargo, se lo ha tenido que adaptar de acuerdo con las circunstancias y retos encontrados. El proyecto inició oficialmente el 7 de septiembre con el Sprint 1. En este periodo se estableció el tema a realizar y se recolectaron los requerimientos principales de los clientes.

El Sprint 2 inició el lunes 21 de septiembre, y en esta fase se documentó las interfaces del proyecto, se identificaron los casos de uso iniciales y se creó el proyecto de la aplicación web con su código inicial. En el Sprint 3, el cual inició el 5 de octubre, se trabajó principalmente en completar la parte funcional y generar las reuniones dentro del sistema, las cuales representan la funcionalidad principal de las aplicaciones.

El Sprint 4 inició el lunes 19 de octubre, y se enfocó en modularizar la información de la base de datos y extraerla dentro de una API (Application Programming Interface en inglés). Este es un componente que permite conectar dos sistemas computacionales entre sí y se utilizó para evitar tener código redundante entre la aplicación web y la aplicación móvil, los detalles sobre esto se detallan más adelante. Gracias a este cambio ambas aplicaciones pueden obtener la misma información en tiempo real de manera independiente. Asimismo, dentro de este ciclo de trabajo se definieron claramente los flujos de las reuniones, los cuales están relacionados a los cambios de estado de estas y se definieron las funcionalidades principales que debe tener el sistema para que pueda ser considerado completo.

Esta última etapa del cronograma fue la que más se tuvo que prolongar ya que la implementación de la API tomó más tiempo de lo estimado. Por este motivo durante este

periodo no se pudo agendar una reunión con los decanos como se tenía planificado. En este contexto, hay que recalcar que la etapa que se tenía designada para hacer pruebas e implementar la funcionalidad adicional se dedicó para terminar las funcionalidades pendientes y trabajar en el diseño de la página. Esta etapa corresponde al Sprint 5, la cual inició el 2 de noviembre, y corresponde a la última etapa del proyecto.

En el transcurso de la metodología de trabajo se han realizado cuatro reuniones con los decanos. En la primera reunión se pudo obtener toda la información necesaria para poder estructurar el proyecto y plasmar algunas ideas para resolver las necesidades de los clientes. En la segunda reunión se presentó a los clientes algunos casos de uso y un flujograma inicial. En la tercera reunión se presentó un demo funcional de la aplicación, en donde se mostró los procesos de registro e inicio de sesión y en la cuarta reunión se habló sobre los reportes y la posibilidad de filtrar la información por semestre, carrera y profesor, así como también sobre la información que se muestra en los perfiles de los usuarios.

Requerimientos funcionales

La aplicación Web se enfoca principalmente en los usuarios encargados de la parte administrativa y la generación de reuniones y reportes, aunque más allá lo que se busca es crear una experiencia amigable para todos los usuarios. En términos generales, mediante las reuniones llevadas a cabo se ha recolectado una lista de requerimientos que el sistema debería tener y entre ellos el requerimiento principal consiste en crear un sistema independiente del sistema manejado por la universidad, el cual pueda leer datos de fuentes externas en formato Excel.

Estos archivos externos van a contener la información relevante sobre los usuarios de acuerdo con el rol que estos ocupan. En este contexto, se han identificado cuatro roles: decanos, profesores, estudiantes y administradores. Adicionalmente, la fuente de datos debería contener

información sobre los semestres, el gpa de cada alumno por semestre y las carreras del politécnico. Toda esta información debería ser ingresada en hojas de cálculo separadas y siguiendo un formato específico. El formato de ingreso que se debería manejar es el siguiente.

Formato de ingreso de datos.

1. **Administradores:** código, correo institucional, nombres, apellidos, correo personal y teléfono.
2. **Decanos**: código, carrera, correo institucional, nombres, apellidos, correo personal y teléfono.
3. **Profesores:** código, carrera, correo institucional, nombres, apellidos, correo personal y teléfono.
4. **Estudiantes:** código, carrera, correo institucional, nombres, apellidos, correo personal, teléfono, período de admisión, tipo de admisión, status, código profesor, correo profesor, nombres profesor, apellidos profesor.
5. **GPA por semestre:** semestre, código estudiante, correo institucional estudiante, nombres estudiante, apellidos estudiante, gpa.
6. **Carreras**: carrera
7. **Semestres**: semestre

Posteriormente, se espera que todos los usuarios que utilicen la plataforma sean miembros de la comunidad USFQ, por lo que para ingresar deberán hacerlo utilizando su correo de la universidad y una contraseña de elección. En vista a que la información va a ser ingresada mediante archivos externos todos los usuarios deberán restablecer su contraseña por lo menos una vez. Para hacer esto los usuarios deberán seleccionar una opción dentro del sistema, la cual enviará la información pertinente a su correo de la universidad. Una vez hecho esto los usuarios podrán utilizar una contraseña de su elección para ingresar.

Con respecto a los profesores se espera que ellos puedan crear, editar y eliminar reuniones dentro del sistema y ver la lista de estudiantes asignados. En relación con esto los estudiantes deberían ver la información de su tutor asignado y tener la capacidad de aceptar o rechazar las reuniones creadas por su tutor. Asimismo, debería existir un sistema de notificaciones que alerte a los usuarios sobre las reuniones que ellos participan. Los decanos, por otra parte, deberán ser capaces de visualizar todas las reuniones creadas en el sistema y recibir información importante como las reuniones creadas, las reuniones eliminadas, el gpa promedio y el número de estudiantes condicionados. De la misma manera, ellos deberán ser capaces de filtrar esta información por semestre y carrera, y poder exportarla en archivos externos.

Los administradores, deberían ser capaces de ingresar información al sistema, esto se aplica tanto para los datos iniciales, como para cualquier dato adicional que se quiera ingresar a futuro. Asimismo, ellos podrán ingresar nuevos usuarios al sistema, editar su información y eliminar aquellos que se consideren inactivos. En este contexto, se requiere un súper usuario dentro del sistema, el cual no pueda ser ni eliminado, ni editado, y este se encargará de administrar e ingresar toda la información dentro de la plataforma, incluso otros usuarios con el rol de administrador. No obstante, a pesar de que los administradores van a cumplir con el rol de agregar, editar, y eliminar información del sistema, hay que recalcar que todos los usuarios van a ser capaces de editar su información personal y su contraseña.

Finalmente, se debe recalcar que todos los archivos que se ingresan y se exportan del sistema deberían tener un formato csv. Con respecto a los datos exportados, el sistema se va a encargar de hacer esto automáticamente. Sin embargo, para el ingreso de información es necesario que los usuarios generen archivos en este formato, ya que de lo contrario podrían surgir resultados inesperados a futuro. Esto se puede hacer de manera sencilla en la mayoría de

las aplicaciones que manejan hojas de cálculo como Excel y la ventaja de hacer esto es que es fácil de transportar y leer ya que usualmente se crean archivos más ligeros.

Estructura del proyecto

La base de datos.

Con el fin de almacenar toda la información requerida se ha planteado una estructura con catorce entidades y siete vistas (Ver ANEXO A). Como entidad principal se tiene al usuario, el cual recibe la información inicial de los archivos externos ingresados al sistema. Esta tabla contiene toda la información común de todos los roles identificados hasta el momento y el motivo principal de mantener esta estructura es debido a que se asume que todos los usuarios del sistema van a ser miembros de la USFQ.

Posteriormente, se encuentran las tablas de los roles (decano, profesor y estudiante), entre las cuales hay algunas que no tienen más información. El motivo por el cual se decidió crear dichas tablas fue para separar la información de los usuarios conceptualmente y tener una base en el caso de que se quieran hacer cambios a futuro. Dichas entidades mantienen una relación con la tabla de usuarios mediante una llave foránea, la cual tiene el mismo valor que su llave principal o id. Esto se lo hizo para evitar tener usuarios duplicados y manejar una relación uno a uno entre la tabla usuario y los decanos, profesores y estudiantes. Asimismo, en relación con la tabla de usuarios se creó la tabla de imágenes. Esta última entidad surgió por la necesidad de almacenar imágenes de los usuarios y eventualmente mostrar una galería de imágenes a futuro.

Para definir los roles de los usuarios existe una tabla rol, la cual tiene un identificador primario y un nombre. En relación a esto, los nombres registrados en el sistema de manera automática son: Decano, Profesor, Estudiante y Administrador. Esta tabla permite clasificar a los usuarios por su rol en la universidad, pero solo funciona cuando se maneja un único rol por

usuario. Por ese motivo se creó una tabla adicional llamada roles de usuario, la cual, en lugar de asignar un único rol por usuario en el sistema, permite asignar cualquier cantidad de roles deseados para un usuario siempre y cuando no existan valores duplicados. Por esta razón si se lo desea un usuario podría ser un decano, un profesor, un estudiante y un administrador, aunque eso se sabe que no sería factible ni funcional. El motivo principal por el cual se agregó esta tabla fue por el caso particular de los decanos, los cuales también ocupan un rol de profesor. Sin embargo, esta tabla también funciona en otros casos como aquellos en los cuales solo se maneja un rol por usuario.

Aparte de esto, una característica importante que comparten la mayoría de los roles, sin incluir a los administradores, es que todos corresponden a una carrera. Para lograr esto, una tabla adicional que se creó fue la entidad carrera, la cual se compone de un identificador y un nombre. En este punto cabe recalcar que esta entidad facilita la recolección y filtración de información en los reportes, lo cual se va a detallar más adelante.

Dentro de los roles de usuario, la entidad que almacena más información corresponde a los estudiantes y esto es razonable ya que lo que se busca es hacer un seguimiento al progreso de su carrera. En este contexto, la entidad estudiante almacena información como el estatus, el periodo de admisión, el tipo de admisión e información adicional como el horario y el progreso de la carrera, la cual no se ha implementado en las interfaces por el momento, pero se lo podría hacer almacenando la información como archivos en formato pdf o como imágenes.

De esta tabla las características más importantes son que almacenan una llave foránea de la tabla profesor y que se relacionan con la tabla gpa por semestre. Esta última entidad permite ingresar el promedio global del estudiante durante el semestre y es sumamente útil ya que de esta manera se puede identificar si existe algún periodo en el cual el estudiante necesita algún tipo de ayuda por tener bajo rendimiento. De la misma manera, para que la información recolectada sea coherente, la tabla gpa por semestre se relaciona con la tabla semestre, la cual

contiene un identificador, un nombre y un valor que permite identificar si se trata del semestre actual o no.

Posteriormente, se ha definido una tabla para almacenar la información relacionada a las reuniones, y su característica principal es que relaciona a estudiantes y profesores al referenciar sus tablas con una llave foránea. En relación con esto, una reunión puede tener solo un profesor y un estudiante, pero estos usuarios pueden ser parte de varias reuniones. Por otra parte, para poder manejar las reuniones correctamente se han creado ocho estados, con los cuales se asegura el correcto funcionamiento de la aplicación y evitan conflictos. Por este motivo dentro del sistema una reunión puede estar: Creada, Editada, Aceptada, Rechazada, Eliminada, En Progreso, Realizada y No Realizada, y toda esta información se almacena de manera automática en la tabla llamada estado reunión (Ver ANEXO B).

Para manejar el flujo de las reuniones y relacionarlas con los usuarios participantes se implementó la tabla notificación. Esta entidad permite comunicar a los usuarios sobre los cambios en las reuniones con las cuales están relacionados y les permite interactuar con ellas de mejor manera. Asimismo, para complementar su funcionamiento y alertar a los usuarios se creó la tabla estado notificación. Esta permite desplegar notificaciones para los usuarios en la pantalla. Con respecto a esto, los estados que pueden mantener una notificación son: Activas, Vistas y Archivadas, y estas se las creo para que tengan un comportamiento similar a los mensajes recibidos en el correo electrónico.

Finalmente, para recuperar la información de manera fácil dentro de la aplicación se han creado siete vistas, las cuales son: gpa, reunión, usuario, profesor, estudiante, notificación y roles de usuario. Estas fueron creadas para contener toda la información de las reuniones, sus estados y los roles identificados, y resolver el problema para recuperar información al tener una tabla con datos en común, como es el caso de la tabla de usuarios, o recuperar información

de las reuniones en las notificaciones. En este contexto se puede destacar que consumir los datos de las vistas ha funcionado bien y ha simplificado el proceso de desarrollo.

La API.

Como se lo mencionó anteriormente, para evitar tener código repetitivo entre las aplicaciones Web y Móvil se ha desarrollado una API (de sus siglas en inglés, Application Programming Interface), la cual va a mantener una conexión con la base de datos. Nuevamente, una API es una pieza de código que permite conectar dos sistemas computacionales entre sí, y permite realizar un intercambio de información. Por este motivo se ha pensado que el proyecto debe estar dividido de una manera similar al sistema Modelo Vista Controlador (MVC), tal como se muestra en la Figura 1.

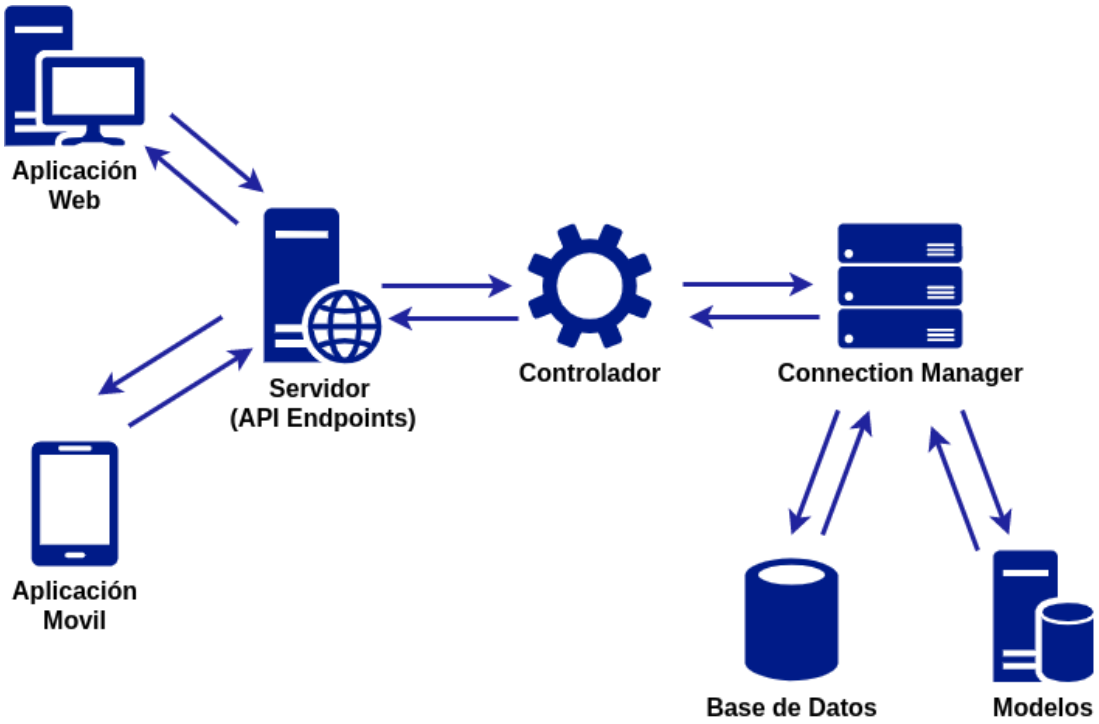


Figura 1. Arquitectura del Sistema

Primeramente, las aplicaciones Web y Móvil establecerán una conexión con la API para solicitar los recursos o información. Una vez recibidas las solicitudes la API se contactará con la base datos mediante un controlador que se encargará de indicar cual es la información correcta que se debe devolver. Este controlador está conectado con un objeto que se ha denominado Connection Manager, el cual mantiene una conexión directa con la base de datos y con los modelos. Una vez que el controlador se conecta con el Connection Manager este obtiene la información de la base de datos y la transmite como un objeto. El controlador distribuye esa información por medio de una función y la API se encarga de distribuir esa información por medio de sus endpoints.

Los endpoints.

Los endpoints o rutas de la aplicación son URLs (Uniform Resource Locators en inglés) designadas por las cuales se pueden acceder a los recursos de un sistema. Estas funcionan mediante una interacción de petición y respuesta en la cual un sistema computacional conocido como "el cliente" accede a la ruta para obtener información particular y en el otro lado el sistema conocido como "el servidor" envía una respuesta. Para que una petición pueda ser procesada correctamente el cliente debe proveer una URL, un método, una lista de cabeceras y un cuerpo o parámetros (Rouse, 2020).

En este caso como se están utilizando tecnologías que simplifican la escritura de código solo se describen los métodos de la petición y el cuerpo o los parámetros. Para recuperar y actualizar información se van a usar los métodos HTTP GET, de los cuales algunos requieren parámetros y el método POST, en el cual se va a pasar un bloque con información a través de formularios. En general, el uso de endpoints ayudan a tener un mayor control sobre la información que se va a devolver para que los usuarios puedan acceder al contenido habilitado para cada uno. Un ejemplo de endpoint en el proyecto es obtener la lista de los estudiantes

asociados a un tutor para que este pueda tomar otras acciones más adelante. Todas las rutas implementadas en el sistema se muestran en los anexos: ANEXO I y ANEXO J.

Interfaces del sistema.

Para este proyecto se han identificado veintidós interfaces, las cuales están relacionadas con los requerimientos establecidos por los clientes. Estas se detallan a continuación:

1. **Inicio de sesión: Si no han ingresado a la plataforma previamente, la página inicial que podrán ver los usuarios es la página de inicio de sesión. Para hacer esto** los usuarios deberán ingresar su correo institucional de la USFQ y su contraseña de elección.
2. **Restablecer contraseña:** En esta página los usuarios deberán ingresar su correo electrónico institucional, con lo cual van a recibir un correo con información para restablecer su contraseña.
3. **Crear contraseña:** Esta vista fue creada principalmente para cuando los usuarios se olviden de su contraseña. Sin embargo, esta página tiene una importancia significativa la primera vez que los usuarios ingresan al sistema, ya que todos deberán crear su contraseña la primera vez que ingresan. Esta vista la podrán acceder mediante el correo enviado a través de la página para restablecer la contraseña.
4. **Cambiar contraseña:** Esta vista permite a los usuarios cambiar su contraseña actual. Esto solo lo van a poder hacer una vez que hayan iniciado sesión y estará disponible para todos los usuarios sin importar su rol.
5. **Página de inicio:** En esta página la información presentada va a variar dependiendo del rol de usuario. Actualmente el sistema muestra la información personal de los usuarios como su vista inicial, pero esto puede cambiar a futuro.

6. **Reuniones:** En esta vista se van a ver todas las reuniones relacionadas a un estudiante o profesor. Una funcionalidad importante que se ha determinado es que los profesores van a ser los únicos capaces de crear una reunión. Los decanos, por otra parte, van a poder visualizar el total de reuniones generadas hasta la fecha, y el sistema va a poder agrupar esa información por semestre.
7. **Creación y modificación de reuniones:** Esta vista estará disponible para los usuarios que cumplan un rol de profesor para que puedan crear, modificar o eliminar una reunión. Se contará con la siguiente información: nombre del profesor, nombre del estudiante, correo del profesor, correo del estudiante, descripción de la reunión.
8. **Detalles de reunión:** Esta vista estará disponible una vez que una reunión haya concluido, esto puede pasar si la reunión se realizó o no. En este caso los usuarios pueden ver los detalles de la reunión, así como también los comentarios ingresados.
9. **Reunión en progreso:** Esta vista se mostrará una vez que un profesor y un estudiante tengan una reunión un día particular. En esta vista el profesor podrá indicar si se realizó o no la reunión, y los usuarios pueden incluir sus comentarios.
10. **Reagendar reunión:** Esta reunión funciona similarmente a la vista que es para editar una reunión y su propósito es cambiar los datos en el caso de que la reunión haya sido rechazada por el estudiante.
11. **Notificaciones:** Vista para notificar a los estudiantes o profesores sobre algún cambio en las reuniones generadas. Esta vista podría ser una buena candidata para incluir en la página principal de los estudiantes y profesores, pero esta decisión no se ha tomado hasta el momento.
12. **Perfil de usuario:** Esta vista estará disponible para todos y aquí los usuarios podrán visualizar toda su información ingresada en el sistema.

13. **Lista de estudiantes:** Esta es una vista que contiene la lista de estudiantes asociadas a un profesor. En esta lista se muestran los nombres, apellidos, código del banner y el gpa, como información rápida.
14. **Información Tutor/Profesor:** Esta vista estará disponible para los usuarios que cumplan un rol de estudiante o profesor. Los estudiantes podrán ver la información de su tutor y los profesores podrán ver información relevante de los estudiantes como: nombre, correo, teléfono, Kárdex y horario
15. **Reportes:** Esta es una vista exclusiva para los decanos, quienes van a poder ver el historial de las reuniones creadas hasta el momento y ver los KPIs más importantes. También van a tener las opciones para exportar los reportes en formato csv. En vista a como ha ido avanzando el proyecto se podría decir que esta página podría ser la página inicial para los decanos. Sin embargo, esta decisión no se la ha tomado hasta el momento.
16. **Editar Perfil:** Esta vista va a permitir a los usuarios modificar su información personal y su imagen de usuario.
17. **Admin - Inicio:** Esta vista está pensada para los usuarios con rol de administrador. En esta los usuarios podrán ver los usuarios registrados, las carreras y podrán elegir el semestre actual.
18. **Admin - Datos iniciales:** Probablemente una de las interfaces más importantes del sistema. Accesible sólo para el súper usuario, esta permitirá ingresar los usuarios por cada rol, las carreras, los semestres y los gpas por semestre. Aquí cabe recalcar que se necesitan subir todos los archivos para poder ingresar los datos.
19. **Admin - Subir imagen:** Esta interfaz permite a los usuarios ingresar imágenes al sistema. Actualmente no existe mayor funcionalidad para esto, pero puede ser utilizada para almacenar y mostrar una galería de imágenes para los usuarios.

20. **Admin - Datos adicionales:** Esta interfaz tiene un propósito a largo plazo, ya que permite agregar información adicional al sistema. Esto es de gran utilidad ya que se pueden ingresar más promedios y nuevos semestres a medida que transcurra el tiempo. Asimismo, si de pronto en un futuro se desean incorporar nuevas carreras, se lo podría hacer.
21. **Admin - Usuarios:** Esta interfaz permite a los administradores ingresar nuevos usuarios y ver aquellos que han sido registrados.
22. **Admin - Editar usuario:** Esta interfaz permite modificar los datos de los usuarios ingresados al sistema, funciona similarmente a la interfaz para editar el perfil del usuario, pero aparte se pueden editar los campos como el correo y el código de la universidad.

El acceso a la plataforma

Para acceder a la plataforma los usuarios deberán ingresar a la página principal de la aplicación web, la cual les mostrará la página de inicio sesión cuando los usuarios no hayan ingresado sus credenciales. En el caso de que el usuario ya cuente con su contraseña, este deberá ingresarla en conjunto con su correo institucional de la universidad. De lo contrario, tal como se mencionó anteriormente, el usuario deberá restablecer su contraseña seleccionando la opción que dice "¿Primera vez que ingresas?" e ingresando su correo electrónico en el formulario que se muestra después de dar clic. Este proceso tiene que seguir todos los usuarios independientemente de su rol.

Cuando se haya seguido este proceso el sistema enviará un correo electrónico al usuario con su enlace personal para crear una contraseña. En este punto el usuario deberá aplastar el botón que se muestra en el correo, y este lo llevará a una vista que le permita generar una nueva contraseña. En este formulario hay dos campos en donde se deben ingresar los mismos valores

y esto se lo hace para evitar errores, y asegurar al usuario que la contraseña ingresada es la que se desea.

Una vez creada la contraseña los usuarios pueden ingresar sus datos, y al hacer esto el sistema les mostrará páginas personalizadas que se ajustan a su rol y a sus datos personales. En este caso como página inicial se tiene el perfil del usuario, en el cual se muestran los datos como el nombre, el rol y su código. En el caso de los estudiantes se incluyen también su gpa global y una tabla que muestra los promedios obtenidos por semestre.

Asimismo, todos los usuarios tienen acceso a las páginas para editar su perfil y cambiar su contraseña. En estas vistas los usuarios no solo serán capaces de cambiar su información textual, sino que también si se desea se puede cambiar su foto de perfil. Esto corresponde a las vistas que comparten todos los roles, pero aparte de esto cada uno va a tener acceso a vistas exclusivas que están diseñadas de acuerdo a sus funcionalidades como usuario.

En el caso de los decanos, una vez que ingresan al sistema, una vista que estará disponible solo para ellos es la página de reportes, la cual se va a detallar más adelante. Por otra parte, los profesores van a tener acceso a tres páginas adicionales. Las primeras consisten en las páginas de reuniones y notificaciones, las cuales también se van a discutir más adelante. Por otra parte, está la vista de estudiantes, la cual permite a los profesores ver cuales estudiantes le fueron asignados y ver la información completa de sus alumnos. Gracias a esto los profesores van a poder gestionar el progreso de la carrera de sus alumnos de mejor manera.

Ahora con respecto a los estudiantes, ellos van a tener prácticamente las mismas vistas que los profesores, con excepción a que, en lugar de tener una vista de estudiantes, ellos tienen una vista que les da acceso a la información de su tutor. Esto puede ser de gran utilidad ya que ahí se incluyen los datos de contacto y estos pueden ser utilizados por los usuarios para mejorar su acercamiento con su tutor de ser el caso.

Las reuniones y las notificaciones

Las reuniones son el componente principal de la aplicación, ya que son un medio de comunicación entre el estudiante y el profesor, y además permiten establecer un registro que luego puede ser usado para generar reportes. Para su correcto funcionamiento y en base a los requerimientos establecidos por los clientes se han establecido algunas características importantes. Primeramente, se asume que solo los profesores van a tener control sobre las reuniones, en el sentido que solo ellos van a ser capaces de ver, crear, editar o eliminar reuniones dentro del sistema.

Para crear una reunión el profesor debe ir a la página de reuniones. Una vez hecho esto tendrá acceso a una tabla con todas las reuniones asociadas. Para crear una nueva reunión el profesor debe aplastar en el botón que dice “Crear Reunión”. Una vez hecho esto el profesor verá una ventana con toda la información correspondiente. Para crear una reunión debe llenar todos los campos y aplastar el botón que dice “Crear Reunión”. Una vez hecho esto, recibirá un mensaje para informarle que la reunión fue creada y en el caso de que se haya seleccionado la opción para enviar notificaciones, el sistema también enviará un correo a los usuarios cada vez que la reunión cambie de estado a futuro (Ver ANEXO D). En este proceso la reunión pasa a un estado de **aceptada.**

En el caso de que el profesor desee cambiar la reunión creada este podrá hacerlo mediante un enlace de acceso rápido, el cual permite ver la información actual de la reunión seleccionada y permite al profesor cambiar cualquier campo de la reunión con excepción al estudiante seleccionado. Nuevamente esta funcionalidad es para evitar conflictos de información a futuro. Una vez realizados los cambios el sistema genera una notificación automática que informa al estudiante sobre los cambios efectuados a la reunión original, y estos también le llegan al correo en caso de que así se lo haya deseado al momento de crear la reunión (Ver ANEXO E). En este proceso la reunión pasa al estado de **editada.**

Ahora, con respecto a la funcionalidad de eliminar reuniones se ha pensado que las reuniones van a ser almacenadas permanentemente dentro del sistema. Esto significa que, si un profesor decide "eliminar una reunión", esta simplemente va a cambiar al estado de **eliminado**, y no va a ser visible por los usuarios que la crearon, pero va a permanecer dentro de la base de datos (Ver ANEXO F). Este almacenamiento permanente va a ser útil en términos de auditoría a futuro.

Para los estudiantes, las reuniones van a ser visibles, pero únicamente aquellas que están relacionadas con su usuario. En este sentido el estudiante también va a ser capaz de ver su progreso durante la carrera. Asimismo, como ocupa un rol importante, este va a tener la capacidad de **aceptar o rechazar** una reunión a la cual fue invitado. Si decide rechazar una reunión, el estudiante deberá ingresar un comentario de por qué está rechazando la reunión para informar al profesor cuando este podría agendar una nueva reunión, o explicando el motivo por el cual no puede asistir. Si este es el caso el profesor va a poder reagendar la reunión en base a la información original. Por otra parte, cuando una reunión haya sido aceptada por el estudiante esta no podrá ser ni eliminada ni actualizada, esto evita que se generen conflictos de información y confusiones por parte de los usuarios. Asimismo, esto ayuda a simplificar el flujo de la aplicación para no tener que preocuparse por eso más adelante.

Una característica que tiene el sistema es que dentro del mismo existe un proceso que corre todos los días a la media noche y automáticamente cambia el estado de aquellas reuniones cuya fecha de realización concuerda con el día. Cuando esto pase se manda una notificación para los usuarios participantes con el fin de recordarles que la reunión fue fijada para esa fecha, la cual nuevamente puede darse por correo de ser el caso. Asimismo, se ha contemplado el caso en el que la fecha de la reunión corresponde al día en el que se la creó. De darse este caso, el sistema también cambia el estado de la reunión, pero solo si el estudiante la acepta en el mismo día.

Una vez que los usuarios hayan visto la notificación del sistema, estos podrán ver una opción llamada “ir a reunión” la cual les permite acceder a los detalles de la reunión e incluir sus comentarios. Del mismo modo los profesores van a poder indicar si la reunión fue realizada, y evidentemente esta información podrá ser verificada con los comentarios del estudiante. Una vez hecho esto el proceso concluye y la reunión pasa al estado **realizada** o **no realizada,** los cuales corresponden con su estado final (Ver ANEXO G).

Este es el flujo de las reuniones relacionado con los estudiantes y profesores, ahora con respecto a los decanos hay que recalcar que ellos van a tener acceso a las reuniones a través de la página de reportes, pero estos no van a poder modificarlas. En el caso de los administradores hasta el momento no se ha pensado los privilegios que estos podrían tener, pero ellos sí podrían ver y modificar las reuniones dependiendo del criterio de los clientes a futuro.

Finalmente, un componente importante que se va a mantener para las reuniones es la creación de logs de auditoría, los cuales van a poder responder las siguientes preguntas: **¿quién creó una reunión?, ¿quién la modificó?, y ¿cuándo ocurrieron estos cambios?** Estos logs van a ser generados automáticamente y van a ser creados a medida que una reunión cambia de estado. Por ejemplo, cuando una reunión pasa de ser Creada a Aceptada, o de Aceptada a Finalizada.

Los reportes

Los reportes son un mecanismo que permite realizar el seguimiento de las reuniones y generar un registro permanente que puede ser utilizado en procesos como la acreditación ABET. Dentro la aplicación web la vista de reportes es accesible únicamente para aquellos usuarios que tienen el rol de decano. Si este es el caso, dicha página podrá ser accedida mediante el menú principal ubicado en la parte superior de la pantalla.

La página de reportes está compuesta por tres secciones. La primera consiste en mostrar indicadores clave o KPIs (de sus siglas en inglés, Key Performance Indicators), que muestran de manera resumida datos importantes como: el número de reuniones creadas, el número de reuniones eliminadas, el gpa promedio del politécnico y el número de estudiantes condicionados, el cual al momento solo considera el gpa de los alumnos. Asimismo, por debajo de la sección de KPIs se ha colocado un botón central que permite a los decanos exportar la información de las reuniones en un archivo con formato csv. Nuevamente, el formato de exportación se eligió debido a que presenta beneficios de portabilidad y lectura de datos, y pueden ser leídos utilizando la aplicación Excel sin mayor inconveniente.

A continuación, se encuentra la sección de los filtros. Para que estos datos tengan mayor relevancia, especialmente cuando se tenga almacenado una mayor cantidad de información, se han creado dos filtros importantes que fueron solicitados por los decanos. Estos son los filtros por semestre y los filtros por carrera. Estos funcionan de manera independiente, en el sentido de que solo se puede aplicar uno a la vez. En otras palabras, si los usuarios deciden filtrar los datos por semestre no podrán hacerlo por carrera y viceversa. Este comportamiento se lo eligió para simplificar funcionalidad y evitar conflictos al momento de aplicar los filtros. De todas formas, se ha tomado en consideración el hecho de que aplicar ambos filtros podría ser de mayor utilidad para los decanos y por ese motivo se recomienda implementar esto como funcionalidad adicional a futuro. Por otra parte, dentro del sistema se ha colocado un botón para eliminar los filtros por si se requiere visualizar los datos agregados.

Finalmente, en la última sección se encuentra el historial de reuniones, el cual contiene todas las reuniones que han sido creadas dentro del sistema dentro de una tabla. La información se ordena de manera determinada utilizando el número de la reunión de manera ascendente. Sin embargo, dentro de esta sección los decanos también podrán ordenar los datos de acuerdo con el resto de las columnas que pueden ser ya sea por el tema, la fecha o el nombre de los

participantes. Asimismo, existe un buscador en la parte superior derecha de la tabla que permite a los usuarios buscar reuniones de forma más específica. Nuevamente esta información se va a ajustar de acuerdo con los filtros utilizados por el usuario.

El rol del administrador

Probablemente el rol más importante dentro del sistema, al menos cuando se trata de la información a ingresar, es el rol del administrador. En este contexto, durante las etapas iniciales de desarrollo se pensó que los administradores ocuparían un rol similar al resto considerando que estos serían ingresados al sistema mediante archivos externos. Sin embargo, esto presentaba un problema ya que de esta forma la única forma de ingresar los datos al sistema era de manera programática o directamente en la base de datos, lo cual no estaba acorde con los requerimientos de los clientes. Por este motivo surgió la necesidad de crear un súper usuario dentro de la base de datos que se encargue de la configuración inicial del sistema.

Con respecto a esto, dentro de la aplicación web existen dos tipos de administradores. El primero es un súper usuario que, como se mencionó anteriormente, va a ser utilizado para agregar los datos iniciales del sistema. Luego están los administradores "comunes" o "regulares", los cuales se encargarán de gestionar y agregar información a medida que transcurra el tiempo. Para ingresar a la plataforma los administradores lo podrán hacer de igual manera que el resto de los roles al acceder a la página de inicio de sesión e ingresar su correo de la universidad y su contraseña de elección.

Una vez hecho esto, los usuarios tendrán acceso a una interfaz distinta al resto, en donde se podrá gestionar la información almacenada. Para esto se han creado vistas únicamente disponibles para este rol, y estas son: la página de inicio, las imágenes, los usuarios registrados y los datos adicionales tales como las carreras, los semestres y los gpas por semestre de los estudiantes. Antes de continuar es necesario recalcar que el diseño actual de la plataforma

asume que los usuarios designados como administradores no van a tener otros roles en el sistema, y por eso la interfaz de administradores tiene un diseño distinto. Por este motivo se recomienda que los administradores sean usuarios dedicados únicamente para esto, ya que de lo contrario podrían darse comportamientos inesperados en la plataforma.

En la página de inicio, los administradores podrán ver los usuarios registrados, las carreras y el semestre actual que se ha fijado en el sistema. En este caso, si se requiere modificar el semestre actual se lo puede hacer al elegir una opción dentro de un menú y aplastar el botón que dice "Actualizar Semestre", con esto el sistema automáticamente modifica este valor de acuerdo con la selección del usuario.

Posteriormente, se encuentra la página para ingresar los datos iniciales al sistema, la cual solo estará disponible para el súper administrador, ya que en el caso de que se vuelvan a ingresar nuevos datos toda la información anterior se perderá y esto incluye a los usuarios registrados. Por este motivo, al ser el único usuario que siempre va a estar disponible en el sistema, el súper usuario es clave para manejar este tipo de operaciones. Asimismo, un detalle importante a considerar es que el proceso de ingresar datos iniciales sólo podrá terminar cuando se ingresen los siete archivos que se presentan en el formulario, los cuales contienen información de carreras, semestres, decanos, profesores, estudiantes, gpas y administradores.

Esta corresponde a la funcionalidad principal de los administradores, sin embargo, como funcionalidades adicionales también se encuentran la opción de ingresar imágenes dentro del sistema, la opción ingresar nuevos usuarios y administrar aquellos registrados, y finalmente ingresar datos adicionales como las carreras, los semestres y los gpas de los alumnos. Nuevamente, hay que recalcar que los archivos ingresados deben tener formato CSV y deben tener la misma estructura de los archivos iniciales.

Lenguajes y tecnologías usadas

Para la creación de la aplicación web se utilizó tecnologías open-source tales como el Framework de desarrollo Web Express JS, el motor de plantillas Handlebars, la librería Bootstrap y el gestor de base de datos MySQL. Asimismo, se utilizaron los lenguajes HTML, CSS y JavaScript, los cuales permiten crear la estructura, el diseño y la lógica que conllevan a la creación de una página web. En este contexto, la terminología open-source se refiere a cualquier tipo de contenido, o en este caso software, cuyo diseño permite a cualquier individuo inspeccionar, modificar y mejorar el código ya existente; todo esto de manera gratuita y sin incurrir en ningún tipo de licencias (Opensource.com, 2020).

De todas las herramientas utilizadas, el componente central que abarca al resto es el Framework Express JS. Este básicamente es una colección de código que se basa en el ambiente de desarrollo Node JS, el cual utiliza JavaScript como lenguaje de programación y npm como su manejador de paquetes o librerías. Lo interesante de estas tecnologías es que permiten utilizar JavaScript desde el lado del servidor, cuando en sus orígenes este lenguaje fue diseñado únicamente para programar la parte del cliente. Para más información sobre estas tecnologías se recomienda revisar su documentación, la cual se encuentra disponible en línea.

Un aspecto que hay que recalcar es que Node JS y Express son simplemente una base que permiten el desarrollo de una aplicación web, pero no incluyen por defecto todas las funcionalidades que se necesitan. Por este motivo cuando se usan estas tecnologías existe una gran cantidad de paquetes o librerías disponibles, las cuales se pueden instalar utilizando el manejador de paquetes npm de acuerdo con la discreción del equipo de desarrollo. En este caso para lograr implementar los requerimientos funcionales de los clientes se tuvo que instalar varios módulos tanto en la versión Web del proyecto como en la API, la cual se la trabajó como un proyecto en conjunto (Ver ANEXO G y ANEXO H).

Consideraciones

Debido a que la plataforma Web es un proyecto en continuo desarrollo existen varias limitaciones que se deben tomar en cuenta para evitar cometer errores y generar comportamientos inesperados en la plataforma. Primeramente, con respecto a las reuniones, cuando llega el día en el que una reunión se debe realizar y pasa al estado en progreso se recomienda que el estudiante incluya sus comentarios antes de que una reunión sea marcada como realizada o no realizada por el profesor.

Asimismo, se debe considerar que, al crear, editar o reagendar una reunión el sistema no valida la fecha de esta. Esto es importante ya que si no se tiene cuidado se podrían crear fechas conflictivas como por ejemplo en el mes anterior, lo cual podría ocasionar comportamientos inesperados por parte de la aplicación. En general hay formularios adicionales en los cuales no se validan los datos ingresados por el usuario, así que esto también debería ser considerado por las personas que se hagan cargo del proyecto a futuro.

Otros temas adicionales corresponden al rol del administrador y a las funcionalidades relacionadas con este rol tan importante. Como primer detalle está el hecho que el administrador debería ser un rol separado, ya que su interfaz genera conflictos con las vistas del resto de usuarios. En este mismo contexto, dentro de la página inicial del administrador se puede seleccionar el semestre actual de la plataforma. Esta es información importante que permite el correcto funcionamiento de las reuniones y la generación de reportes. Por defecto, el semestre que se fija como actual en la configuración inicial del sistema es el primer semestre ingresado. En este sentido todas las reuniones creadas tendrán ese semestre como el semestre en el que fueron realizadas de manera automática. Por este motivo se recomienda que al inicio de cada semestre el administrador cambie este valor para evitar generar información errónea.

Ahora, con respecto a los datos de ingreso se debe tomar en consideración que todos los archivos ingresados al sistema deberían estar en formato csv, con excepción a las imágenes. Anteriormente, se describieron los motivos principales por los cuales es necesario utilizar este formato y también que estos se pueden generar de manera sencilla cuando se utilizan aplicaciones de hojas de calculo como Excel.

Posteriormente, dado a que el sistema permite al súper usuario ingresar los archivos iniciales al sistema hay que recalcar que al hacer esto toda la información ingresada con anterioridad se pierde. Por este motivo se recomienda utilizar esta funcionalidad únicamente la primera vez que se quieran ingresar los datos al sistema o por lo menos cuando ya se haya almacenado la información previa en otra fuente de datos.

Con relación a las imágenes, existe una interfaz que permite a los usuarios ingresar nuevas imágenes al sistema y almacenarlas permanente. Por el momento, esta es una funcionalidad adicional que no está implementada completamente por el momento, pero se la dejó activada debido a que los decanos solicitaron implementar una galería de imágenes que permita a los usuarios escoger imágenes para su foto de perfil. Sin embargo, como ya se lo mencionó esto debería ser implementado a futuro y al hacerlo se debería evaluar cuales serían los mejores criterios para esto. Finalmente, con respecto a los roles de usuarios y a los estados de las reuniones y notificaciones se tiene que recalcar que no se pueden ingresar nuevos valores, ya que estos se generan de manera automática. No obstante, si los usuarios requieren de esta funcionalidad a futuro se lo debería implementar por el equipo que se haga cargo del proyecto.

Riesgos identificados

El riesgo principal del proyecto son las fuentes de datos ya que tanto la aplicación web como la móvil serán independientes al sistema Banner de la USFQ, por lo tanto, no se conoce la cantidad exacta de datos con los que se va a trabajar. Debido a esto existe un problema con la fuente de datos ya se debe tener en cuenta que toda la información ingresada debe estar con archivos Excel, los cuales deben ser exportados en un formato csv. Asimismo, al ser información ingresada por los usuarios existen errores inherentes asociados al factor humano, y esto podría ser problemático especialmente al ingresar los semestres y las carreras y al relacionarlos con el resto de los datos. Esto es debido a que estos necesitan ser ingresados exactamente igual a como lo fueron inicialmente o de lo contrario pueden ocurrir errores inesperados.

Estrategias para mitigar los riesgos

Como estrategia principal para mitigar los riesgos y corregir a tiempo posibles errores se necesita hacer un seguimiento continuo y exhaustivo al proyecto. Por este motivo se recomienda seguir utilizando metodologías ágiles, pues estas permiten mantener comunicación continua con los clientes y detectar cualquier tipo de anomalía. Asimismo, es necesario documentar absolutamente todos los errores que se encuentran al igual que las soluciones implementadas, ya que esto permite reducir trabajo en el largo plazo y ayudar a nuevos miembros de trabajo, los cuales se podrían familiarizar con el sistema y sus errores de manera más rápida. En este sentido, el uso de herramientas de oficinal como la suite de Microsoft o Google son esenciales para recaudar información. Finalmente, se recomienda utilizar la aplicación de administración de proyecto Asana, ya que de esta manera se puede evaluar el progreso del trabajo realizado.

Discusión

En base al trabajo realizado hasta la fecha se ha logrado obtener información relevante no sólo sobre los procesos internos que se llevan en el Politécnico, sino que también sobre la metodología SCRUM y la importancia de mantener contacto con los clientes. En este contexto los resultados indican que la interacción intuitiva y fácil con el sistema son aspectos importantes que se deben mantener en cuenta. Asimismo, se ha logrado identificar que no necesariamente un sistema completamente automatizado es lo más conveniente. Con respecto a esto es importante recalcar que hay que adaptarse a la forma de trabajo de los usuarios y crear un sistema que se ajuste a sus necesidades.

Naumann, et ál. (2007) menciona que un sistema se puede considerar intuitivo cuando el usuario es capaz de interactuar efectivamente con este, sin emplear mayor esfuerzo cognitivo y utilizando un modelo mental en base a conocimientos previos. Estos pueden darse en su experiencia con aplicaciones comunes como los son las hojas de cálculo, editores de texto o páginas web. Asimismo, al momento de diseñar un sistema se debe considerar que este debe estar acorde con las tareas a usar, cumplir con las expectativas de los usuarios y no requerir ningún manual de instrucciones para su uso. Una vez que se toma esto en cuenta el usuario puede dedicar su esfuerzo a realizar sus tareas en lugar de perder tiempo y energía en aprender cómo funciona la tecnología utilizada.

El sistema actual, tiene varias complicaciones relacionadas a los procesos para generar reuniones, las cuales dificultan el registro de información y el seguimiento a los estudiantes. Como se mencionó anteriormente, el sistema actual es poco amigable y la información no se almacena permanentemente, por lo que los usuarios no tienen iniciativa para usarlo y por ende no se mantiene un registro activo. Esto también complica los procesos de acreditación de las carreras, ya que para la certificación ABET se necesita tener la mayor documentación posible para respaldar las actividades que se llevan a cabo en la universidad.

En relación con la metodología usada, la documentación del proyecto indica que mantener reuniones periódicas ayuda a mantener comunicación entre el equipo de trabajo y los clientes, y presenta varios beneficios que aportan al desarrollo del producto. Uno de los principales beneficios es que se pueden descubrir nuevas funcionalidades y nuevos requerimientos que agregan valor al cliente. Asimismo, el hecho de mantener reuniones periódicas permite documentar información con mayor facilidad y con dicha información se pueden hacer cambios, encontrar mejoras, eliminar fallas e identificar riesgos asociados. Lei et ál. (2017) indica que Scrum es una metodología ágil diseñada para manejar requerimientos cambiantes de manera rápida al mejorar la comunicación entre los miembros de trabajo y el resto de los agentes involucrados. Asimismo, describe que esta se considera como la principal metodología a usar en proyectos de Software, y los factores que más influyen son la transparencia, la inspección y la adaptación.

CONCLUSIONES

En el presente trabajo se documentó información relacionada a la Aplicación Web para el Sistema de Gestión de Tutorías planteado y el uso de la metodología SCRUM para su desarrollo. En este contexto, se mencionó que esta metodología de trabajo ayuda a reducir los plazos de entrega y genera mayor valor al cliente. El desarrollo del proyecto se lo hizo en un plazo de cuatro meses y cinco etapas o fases de trabajo. En este periodo de tiempo se tuvieron cuatro reuniones con los clientes, se estableció la fuente de datos del sistema y los roles de usuario. Asimismo, se definieron las interfaces iniciales, los casos de uso, y se creado una base de datos para almacenar toda la información. De la misma manera, se han definido los flujogramas principales de la aplicación y se ha logrado crear una versión funcional de la misma.

Los resultados indican que mantener un contacto continuo con el cliente permite generar mayores ideas y el proceso de trabajo ha permitido estructurar el proyecto de mejor manera, y documentar la mayor información posible. Asimismo, el trabajo iterativo y continuo ha permitido identificar nuevas necesidades y funcionalidades que no se pensaron en un inicio. Con respecto a las reuniones, se han implementado las funcionalidades descritas en un inicio y se ha podido crear las vistas del administrador. Este es el estado actual del proyecto y a futuro lo que se plantea hacer es realizar pruebas para la detección de errores y si hay tiempo disponible implementar nuevas funcionalidades como la opción para cambiar de tutor, integrar funcionalidad de WhatsApp, y agregar información adicional para los usuarios. De la misma forma, lo que se buscaría hacer en el tiempo restante es mejorar el diseño de la página y la experiencia del usuario. Finalmente, se espera poder agendar una presentación para los decanos y entregar el producto al Politécnico para que se lo pueda utilizar el próximo semestre.

REFERENCIAS BIBLIOGRÁFICAS

Abellán, E. (05/03/2020). *Metodología Scrum: qué es y cómo funciona.* We are marketing (wam). Recuperado el 25 de septiembre del 2020 de https://www.wearemarketing.com/es/blog/metodologia-scrum-que-es-y-como-funciona.html

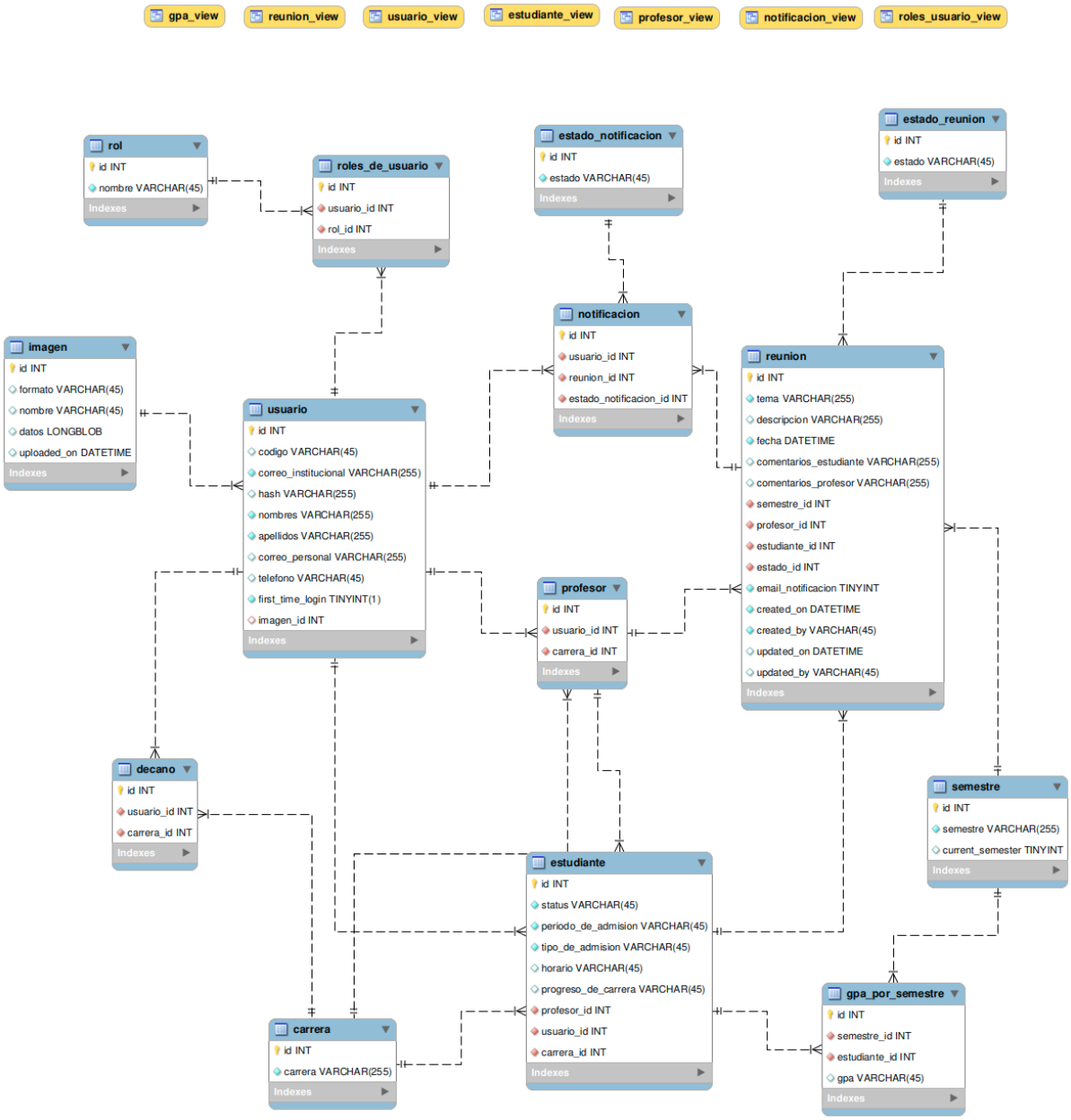
Lei, H., Ganjeizadeh, F., Jayachandran, P. & Ozcan, P. (02/2017). *A statistical analysis of the effects of Scrum and Kanban on software development projects.* Robotics and Computer-Integrated Manufacturing. Elsevier. Recuperado el 25 de septiembre del 2020 de https://www-sciencedirect-com.ezbiblio.usfq.edu.ec/science/article/pii/S0736584515301599/pdfft?md5=5002dc12c5b37f54b003e497b413047b&pid=1-s2.0-S0736584515301599-main.pdf

Naumann, Anja & Hurtienne, Jörn & Israel, Johann & Mohs, Carsten & Kindsmüller, Martin & Meyer, Herbert & Husslein, Steffi. (2007). *Intuitive Use of User Interfaces: Defining a Vague Concept.* 128-136. 10.1007/978-3-540-73331-7_14. Recuperado el 25 de septiembre del 2020 de https://www.researchgate.net/publication/221096159_Intuitive_Use_of_User_Interfaces_Defining_a_Vague_Concept

Opensource.com. (2020). *What is open source?* Recuperado el 10 de diciembre del 2020 de https://opensource.com/resources/what-open-source

Rouse, M. (08/2020). *API endpoint*. Recuperado el 6 de noviembre del 2020 de https://searchapparchitecture.techtarget.com/definition/API-endpoint#:~:text=An%20API%20endpoint%20is%20a,server%20and%20receiving%20a%20response.

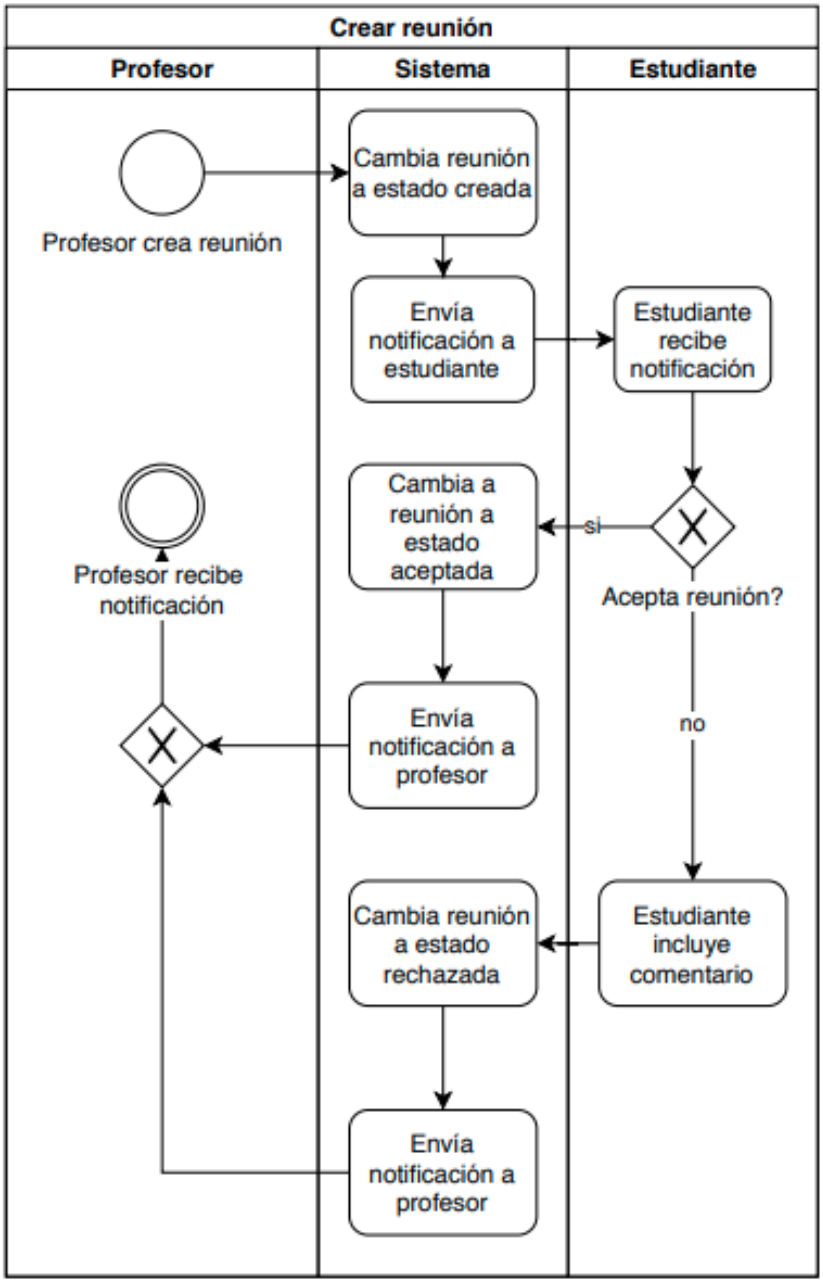
ANEXO A: DIAGRAMA ENTIDAD RELACIÓN BASE DE DATOS



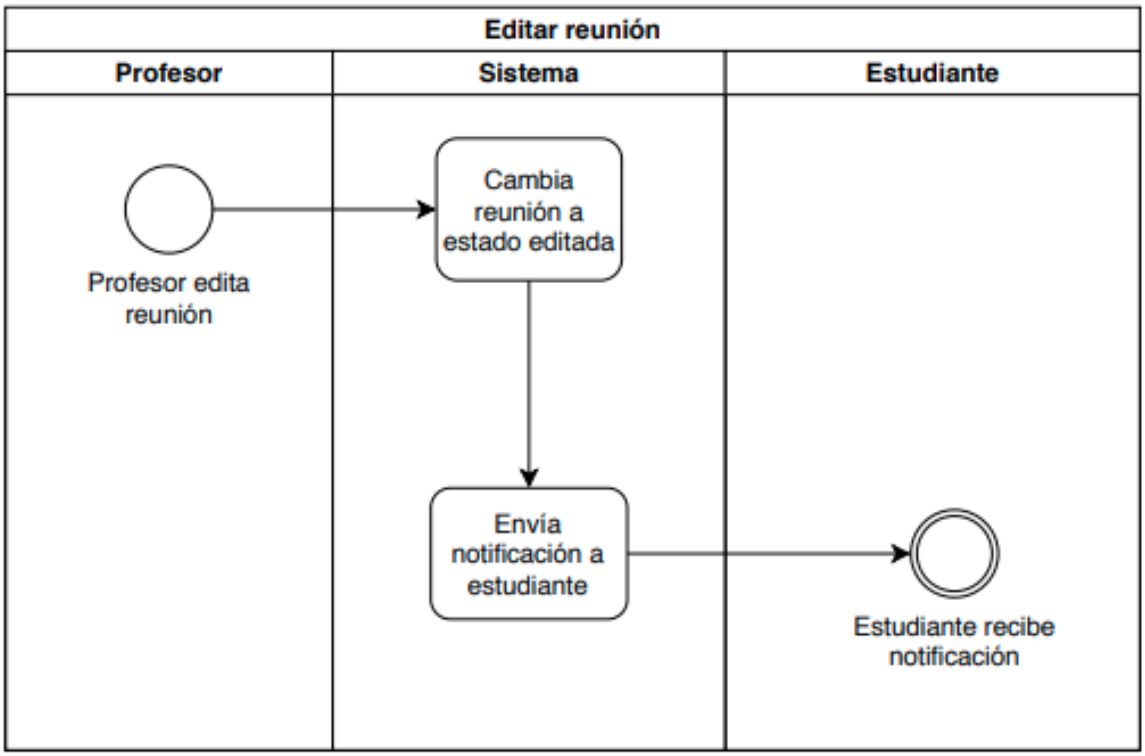
ANEXO B: ESTADOS DE LAS REUNIONES

Estados de Reunión	
Creada	Creada
Editada	Editada
Aceptada	Aceptada
Rechazada	Rechazada
Eliminada	Eliminada
En Progreso	En Progreso
Realizada	Realizada

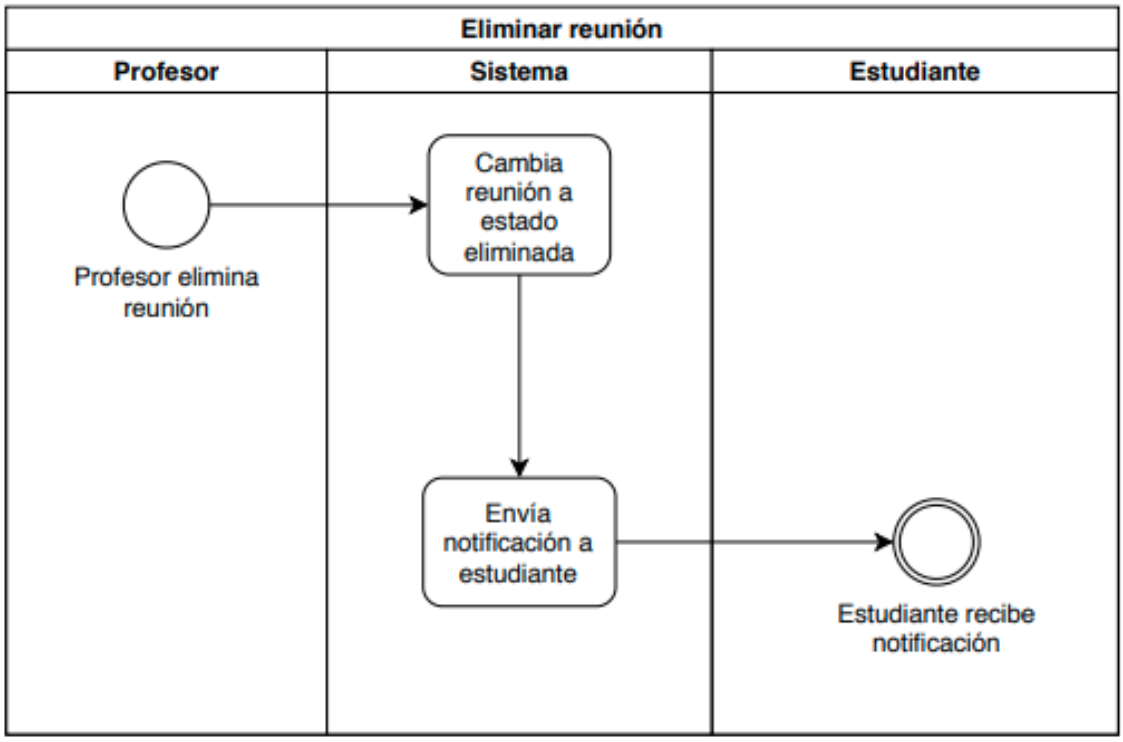
ANEXO C: PROCESO PARA CREAR UNA REUNIÓN



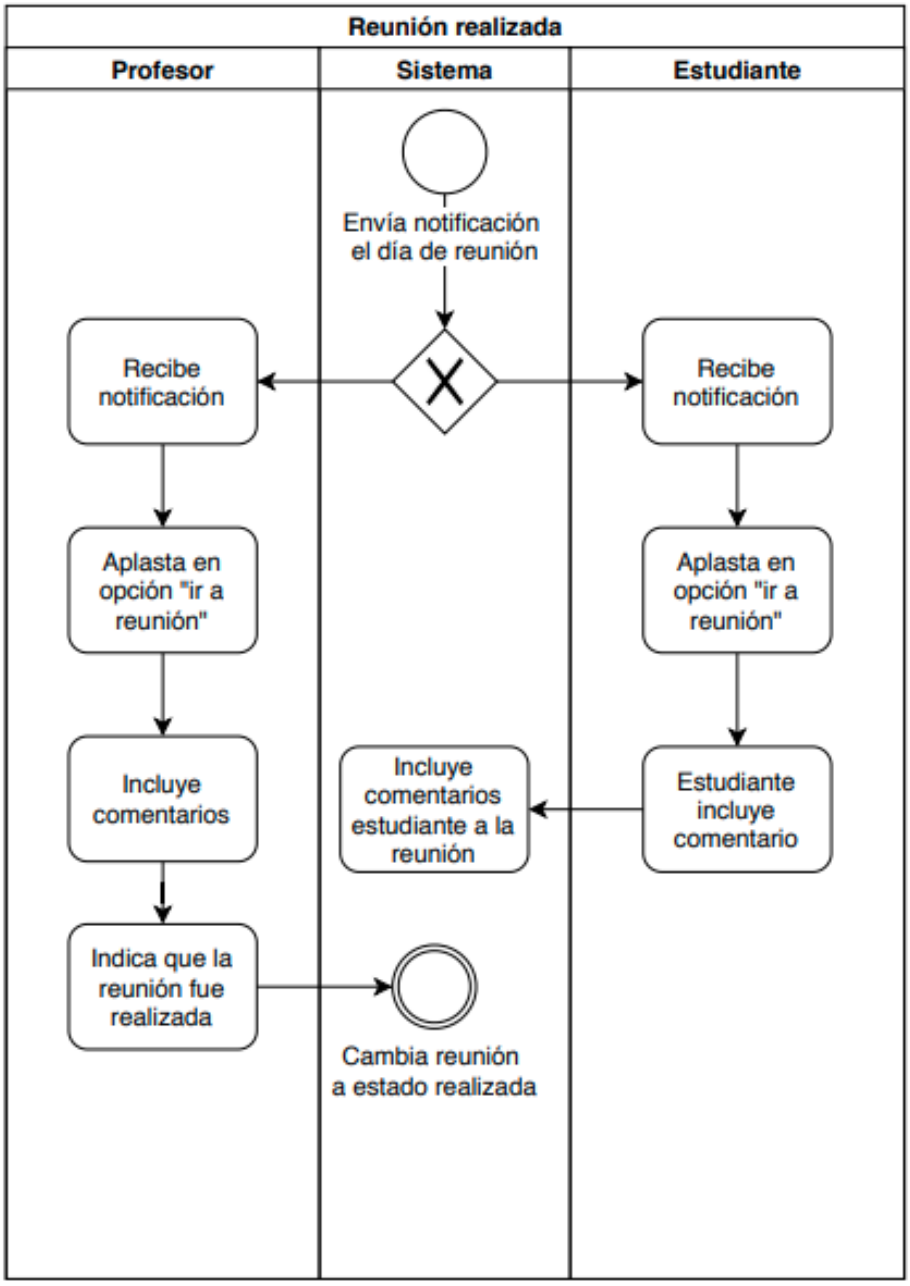
ANEXO D: PROCESO PARA EDITAR UNA REUNIÓN



ANEXO E: PROCESO PARA ELIMINAR UNA REUNIÓN



ANEXO F: PROCESO PARA REUNIÓN REALIZADA



ANEXO G: MÓDULOS UTILIZADOS EN LA APLICACIÓN WEB

Módulo	**Versión**	**Funcionalidad**
axios	0.20.0	Cliente http que permite hacer peticiones utilizando promesas.
bcrypt	2.4.3	Librería que permite encriptar contraseñas. Utiliza el algoritmo bcrypt.
connect-flash	0.1.1	Permite mostrar mensajes de comunicación al cliente. Estos pueden ser mensajes de éxito o error.
express	4.17.1	Módulo que instala express en el ambiente de NodeJS.
express-handlebars	5.1.0	Permite instalar el motor handlebars en el ambiente de NodeJS
express-session	1.17.1	Permite almacenar sesiones en la aplicación web.
express-validator	6.6.1	Permite realizar validaciones en los formularios de la página.
morgan	1.10.0	Es un módulo que permite enviar mensajes de logs en la terminal.
multer	1.4.2	Módulo utilizado para subir imágenes al sistema.
passport	0.4.1	Permite crear un sistema de autenticación.
passport-local	1.0.0	Implementa el mecanismo de autenticación para el sistema. Utilizado en conjunto con el módulo passport.

path	0.12.7	Provee de utilidades que permiten el acceso y la manipulación de directorios y archivos
prettier	2.1.2	Permite dar formato al código.
uuid	8.3.2	Permite crear identificadores para los usuarios a través de cookies (actualmente no se utiliza).
eslint	7.10.0	Herramienta de análisis para corregir el formato de código.
eslint-config-airbnb-base	14.2.0	Implementa el estilo de código de Airbnb.
eslint-config-prettier	6.15.0	Trabaja en conjunto con el módulo prettier para dar formato al código.
eslint-plugin-import	2.22.1	Se encarga de ver la sintaxis al importar y exportar archivos.
eslint-plugin-prettier	3.1.4	Trabaja en conjunto con el módulo prettier para dar formato al código.
nodemon	2.0.4	Módulo usado en desarrollo que permite recargar el servidor cuando se hace algún cambio en el código.
reload	3.1.1	Automáticamente recarga la página en el navegador cuando se hace un cambio en el código. Se utiliza únicamente en un ambiente de desarrollo.

ANEXO H: MÓDULOS UTILIZADAS EN LA API

Módulo	**Versión**	**Funcionalidad**
axios	0.21.0	Cliente http que permite hacer peticiones utilizando promesas.
csvtojson	2.0.10	Permite manipular archivos csv y convertirlos a objetos JSON.
dotenv	8.2.0	
express	4.17.1	Módulo que instala express en el ambiente de NodeJS.
express-handlebars	5.2.0	Permite instalar el motor handlebars en el ambiente de NodeJS
morgan	1.10.0	Es un módulo que permite enviar mensajes de logs en la terminal.
multer	1.4.2	Módulo utilizado para subir imágenes al sistema.
mysql2	2.2.5	Cliente de MySQL para NodeJS
node-schedule	1.3.2	Es un agendador de trabajos que utiliza el modelo cron utilizado en sistemas Unix.
nodemailer	6.4.14	Permite mandar correos electrónicos desde el servidor.
nodemailer-express-handlebars	4.0.0	Permite comprimir los archivos de las vistas en formato hbs y muestra su contenido en el correo.

path	0.12.7	Provee de utilidades que permiten el acceso y la manipulación de directorios y archivos
prettier	2.1.2	Permite dar formato al código.
sequelize	6.3.5	Es un ORM que devuelve la información como una promesa. Esto permite representar a las tablas de la base de datos como objetos.
winston	3.3.3	Permite imprimir logs en la terminal de manera personalizada.
eslint	7.11.0	Herramienta de análisis para corregir el formato de código.
eslint-config-airbnb-base	14.2.0	Implementa el estilo de código de Airbnb.
eslint-config-prettier	6.13.0	Trabaja en conjunto con el módulo prettier para dar formato al código.
eslint-plugin-import	2.22.1	Se encarga de ver la sintaxis al importar y exportar archivos.
eslint-plugin-prettier	3.1.4	Trabaja en conjunto con el módulo prettier para dar formato al código.
nodemon	2.0.6	Módulo usado en desarrollo que permite recargar el servidor cuando se hace algún cambio en el código.

ANEXO I: MÉTODOS GET IMPLEMENTADOS EN LA API

Número	Ruta
1	/
2	/home
3	/signin
4	/user-by-email
5	/user-by-id
6	/user-roles
7	/image-by-id
8	/signup
9	/forgot-password
10	/first-time-login
11	/create-password/:userId
12	/logout
13	/meetings
14	/meetings/create
15	/meetings/meeting-by-id
16	/tutor
17	/students
18	/student/:usedId
19	/notifications
20	/active-notifications
21	/reports
22	/semesters
23	/carreras
24	/reports-by-semester
25	/reports-by-carrera
26	/edit-profile
27	/change-password
28	/current-semester
29	/admin
30	/admin/users

ANEXO J: MÉTODOS POST IMPLEMENTADOS EN LA API

Número	Ruta
1	/signin
2	/create-password
3	/meetings
4	/meetings/create
5	/meetings/edit
6	/meetings/accept
7	/meetings/reject
8	/meetings/done
9	/meetings/reschedule
10	/notifications
11	/delete-notification
12	/archive-notification
13	/viewed-notification
14	/edit-profile
15	/change-password
16	/send-pwd-reset
17	/send-meeting-notification
18	/admin/upload
19	/admin/current-semester
20	/admin/files-upload
21	/admin/edit-user
22	/admin/delete-user
23	/admin/add-users
24	/admin/additional-files