UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

Sistema de Gestión de Tutorías: Aplicación Móvil

Andrea Camila Porras Sampedro

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, 11 de diciembre de 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 Móvil

Andrea Camila Porras Sampedro

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

Quito, 11 de diciembre de 2020

© DERECHOS DE AUTOR

Nombres y apellidos: Andrea Camila Porras Sampedro

Código: 00136217

Cédula de identidad: 1718349382

Lugar y fecha: Quito, 11 de diciembre de 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UNPUBLISHED DOCUMENT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

El Colegio de Ciencias e Ingenierías de la Universidad San Francisco de Quito USFQ, en la actualidad cuenta con un sistema de tutorías difícil de usar, el mismo que no permite tener una comunicación fluida entre profesores tutores y estudiantes. De igual manera, los decanos acusan problemas al no disponer de reportes oportunos del estado de las tutorías para poder presentar reportes obtenidos en las carreras que poseen la certificación ABET.

Por lo antes mencionado, se propone el desarrollo de una aplicación móvil amigable que funcione como una herramienta digital que permita mejorar la comunicación (estudiante-profesor y viceversa). Además, esta herramienta facilita a los decanos de manera sencilla y ágil el acceso a reportes oportunos en base a la información almacenada en el sistema.

Este documento detalla el proceso de desarrollo de esta aplicación móvil. En este proceso de desarrollo se creó una base de datos para el almacenamiento de información y una API Rest para la conexión de la base de datos con la aplicación móvil. Con la creación de esta aplicación se dispone de una mejora en la comunicación entre tutor y tutelado, se mantiene un registro real sobre las reuniones creadas y se generan reportes para los decanos.

Al tratarse de una primera versión de la aplicación móvil es importante realizar una fase de pruebas con usuarios finales para validar su funcionalidad y permitir una evolución natural de la aplicación para que pueda ser utilizada ampliamente por la USFQ.

Palabras clave: Aplicación móvil, base de datos, API, IONIC, Node.js, Tutorías, Comunicación.

ABSTRACT

The Sciences and Engineering School (Colegio de Ciencias e Ingenierías) at Universidad San Francisco de Quito USFQ, has currently a difficult to use tutoring, which does not allow fluid communication among teachers and students. Similarly, authorities denounce trouble trying to access tutoring reports, especially for ABET certified careers.

For the aforementioned, the development of a friendly mobile application is proposed, which works as a digital tool that allows communication improvement (student-teacher and vice versa). In addition, this tool facilitates authorities in a simple and agile way to access reports generated based on the information stored in the system.

This document details the development process of such mobile application. In this development process, a database was created for storing information and a Rest API for connecting the database with the mobile application. With the creation of this application, there is an improvement in the communication between the tutor and the ward, a real record is kept of the meetings created and reports are generated for the deans.

As it is a first version of the mobile application, it is important to test it with a group of final users to validate its functionality and allow a natural evolution of it in order to become broadly used by USFQ members.

Key words: Mobile application, database, API, IONIC, Node.js, Tutorials, Communication.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Uno de los objetivos principales 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. Parte de 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 requieren acceder a reportes sobre las reuniones realizadas a lo largo de cada semestre para conocer mejor del desempeño de los tutores y del programa de tutoría; así como también, para dar soporte a las carreras que tienen certificaciones ABET.

Actualmente, el Politécnico cuenta con un sistema de seguimiento obsoleto y poco amigable con los usuarios que no facilita la recolección de información, pues se tiene un sistema de información básico para el registro del contenido de reuniones. Asimismo, el sistema actual es muy poco accesible ya que para encontrar el servicio los usuarios deben navegar por varias páginas y realizar varios pasos para agendar las reuniones respectivas, proceso que es completamente manual ya que se lo hace por correo electrónico y debido a esto no se tiene un registro de esa interacción ni mucho menos recordatorios para las reuniones agendadas. 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 un seguimiento deseado.

Por este motivo, el desarrollo de este proyecto se enfoca en la resolución de los problemas encontrados por parte de decanos, profesores y estudiantes que pertenecen al Politécnico. En este documento se describe la solución propuesta para resolver los problemas antes mencionados, que consiste en una aplicación móvil que brinda la comodidad y fácil accesibilidad tanto a profesores como estudiantes. Para los profesores, esta aplicación es un canal de comunicación alternativo 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 y oportuno. Finalmente, para los estudiantes se brinda la información del perfil del tutor asignado y las reuniones programadas por el tutor.

Objetivos Generales y Específicos

Objetivos Generales.

- Implementar un sistema digital para realizar seguimiento a los estudiantes tutorados del politécnico.
- Establecer un registro de las reuniones generadas y realizadas por cada profesor para la generación de reportes.

Objetivos Específicos.

- Mejorar la comunicación entre el profesor y el estudiante tutelado.
- Mejorar la experiencia tanto del estudiante como del profesor al agendar reuniones.
- Facilitar el acceso a la información del estudiante y si progreso durante la carrera.

DESARROLLO DEL TEMA

Casos de Uso

Para el desarrollo de la aplicación se consideraron 5 casos de uso generales:

Inicio de sesión por primera vez

Tal como se muestra en la figura 1, este caso de uso tiene un actor. Este actor que es un usuario nuevo de la plataforma tiene que realizar un registro por primera vez para poder crear su contraseña personal. El usuario ingresa en la opción de “Registro” y a continuación ingresa el correo electrónico de la institución. El sistema envía un email al correo ingresado el cual cuenta con un botón que redirige a la aplicación en donde el usuario ingresa el correo electrónico, contraseña y confirmación de la contraseña. El sistema almacena la información en la base de datos y la aplicación muestra la pantalla de “Inicio de Sesión”.

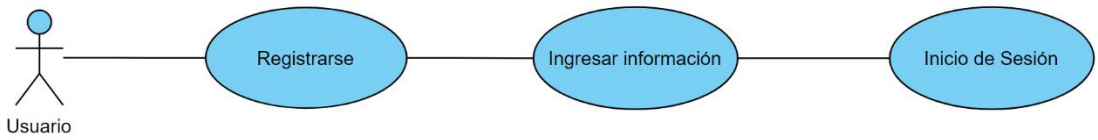

Figura 1. Inicio de Sesión por primera vez

Inicio de sesión

Si el usuario ya posee una contraseña en el sistema debe realizar el inicio de sesión como se muestra en la figura 2. El usuario ingresa su correo electrónico y contraseña. El sistema verifica y valida los datos ingresados. Si la contraseña o correo electrónico no son los iguales al del sistema, se muestra un mensaje de error. Si la validación de los datos es exitosa la aplicación permite el ingreso y se muestra el perfil del usuario en pantalla.

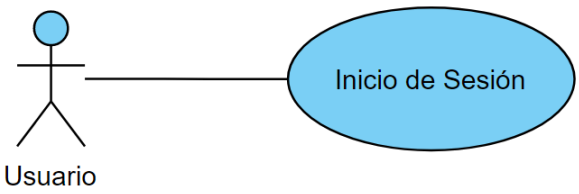

Figura 2. Inicio de Sesión

Aceptar o rechazar una reunión

Como se muestra en la figura 3, este caso de uso tiene un solo actor. El actor es un estudiante el cual debe aceptar o rechazar una reunión que su tutor le ha asignado. Para este proceso el estudiante ingresa a las notificaciones en donde se muestra todas las nuevas reuniones. El usuario tiene dos opciones rechazar o aceptar la reunión. Para aceptar la reunión presionará el botón con un visto, por el contrario, para rechazar una reunión el usuario presionará el botón con una "x". Una vez aceptado o rechazado la reunión, la notificación desaparecerá de esa vista y la reunión podrá observarse en la vista de reuniones.

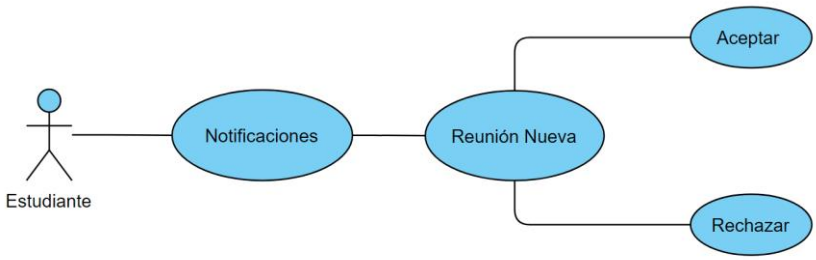

Figura 3. Aceptar o Rechazar reunión

Arquitectura de la Aplicación

Para el funcionamiento de la aplicación móvil se utiliza una base de datos que es la encargada de almacenar todos los datos, una API que es la encargada de conectar la base de datos con la aplicación móvil para mostrar y guardar información en la base de datos, y finalmente la estructura de la aplicación móvil que contiene todas las vistas que se van a utilizar para profesores y estudiantes.

Base de datos

La base de datos cuenta con 8 tablas (estudiante, profesor, decano, usuario, rol, reunión, estado reunión y notificación) y cuatro vistas (estudiante, profesor, reunión y notificación) como se muestra en la Figura 4.

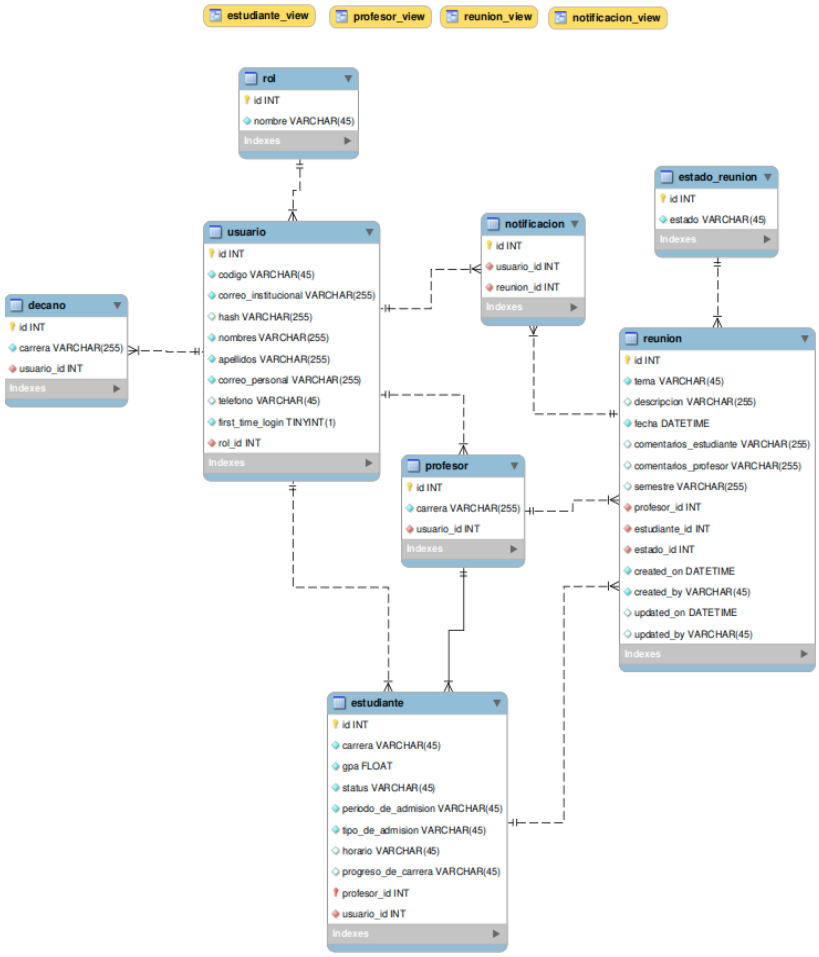

Figura 4. Estructura de la base de datos.

Descripción de las tablas

1. Tabla usuario: esta tabla es la encargada de almacenar información compartida sobre todos los usuarios existentes.

2. Tabla rol: esta tabla contiene los roles que un usuario puede tener: (1) Decano, (2) Profesor, (3) Estudiante y (4) Administrador.
3. Tabla decano y profesor: estas tablas contienen información (carrera, id, usuarioId) sobre los decanos y profesores.
4. Tabla estudiante: esta tabla contiene información (id, carrera, gpa, status, periodo de admisión, tipo de admisión, horario, progreso de carrera, profesorId, usuarioId) sobre un estudiante.
5. Tabla estado reunión: esta tabla contiene los diferentes estados que puede tener una reunión: (1) Creada, (2) Editada, (3) Aceptada, (4) Rechazada, (5) Eliminada, (6) Realizada y (7) Finalizada.
6. Tabla reunión: esta tabla es la encargada de almacenar toda la información (id, tema, descripción, fecha, comentarios estudiante, comentarios profesor, semestre, profesorId, estudianteId y estadoId) sobre las reuniones creadas entre profesores y estudiantes.
7. Tabla notificación: esta tabla contiene información (id, usuarioId, reuniónId) para crear una notificación de cuando una reunión ha sido creada.

Descripción de las vistas

1. Vista estudiante: esta vista contiene toda la información que se necesita sobre un estudiante. Esta información se obtiene de la unión de la tabla de usuario y la tabla de estudiante.
2. Vista profesor: esta vista contiene toda la información que se necesita sobre un profesor. Esta información se obtiene de la unión de la tabla usuario y la tabla profesor.

3. Vista reunión: esta vista contiene toda la información necesaria sobre las reuniones creadas. La información se obtiene de la unión de la tabla reunión, estudiante, profesor y estado reunión.
4. Vista notificación: esta vista contiene toda la información al momento de crear una reunión para poder notificar a los usuarios. La información se obtiene de la unión de la tabla notificación, usuario y la vista de reunión.

Información sobre la base de datos

La entidad principal es la tabla de los usuarios ya que en ella se almacena toda la información común de los roles que se han identificado anteriormente. Esta entidad mantiene una relación con las tablas de decano, profesor, estudiante y roles por medio de una llave primaria que nos ayudará a tener una relación entre tablas para abarcar toda la información necesaria por cada uno de los usuarios.

Otra entidad que es importante en la base de datos es la tabla de reuniones ya que gracias a esta tabla se puede tener una relación por medio de una llave foránea entre profesores y sus tutelados (estudiantes). Se debe tener en claro que una reunión solo puede contener a un profesor y a un estudiante para que pueda ser generada, sin embargo, un profesor podrá generar varias reuniones con un mismo estudiante. Para tener un control sobre el desarrollo de las reuniones esta tabla se relaciona con la tabla estado reunión ya que se podrá conocer los diferentes estados en los que una reunión se encuentra, lo que ayudará a los decanos a poder tener un mayor control sobre las reuniones que se han generado en cada semestre.

Finalmente, la tabla de notificaciones es la que nos va a ayudar a comunicar entre profesores y estudiantes los cambios que se vayan generando en las reuniones para lograr una mejor comunicación entre estos dos usuarios.

Como se mencionó anteriormente, la base de datos cuenta además con 4 vistas que nos ayudan a poder recuperar de manera fácil y eficiente varia información que se necesita para cada una de las vistas dentro de la aplicación móvil.

Application Programming Interface (API)

Para poder recuperar e ingresar información a nuestra base de datos por medio de la aplicación móvil se diseñó una API que nos ayuda a conectar las vistas de la aplicación con la base de datos creada. La estructura de la API se basa en modelos, controladores, rutas y conexión a la base de datos.

1. Modelos: Existen 12 modelos que equivalen a cada una de las entidades creadas en la base de datos (8 tablas y 4 vistas). Cada uno de los modelos contiene la información de cada entidad respectivamente y cada dato es acompañado con la especificación sobre el tipo de dato y la opción de poder o no tener un valor vacío. Los modelos se los exporta para que puedan ser utilizados por los controladores y las rutas.
2. Controladores: al igual que los modelos existen 12 controladores, uno por cada modelo que se creó. Por medio de estos controladores se podrá realizar diferentes consultas a la base de datos para poder recuperar o almacenar información mientras se dé el funcionamiento de la aplicación móvil. En estos controladores se podrá visualizar diferentes funciones para la extracción de datos que hacen uso de las siguientes funciones: *findOne*, *findByPk* y *findAll*. Los controladores pueden ser usados entre ellos mismos en el caso de que se necesite tener información de otra entidad para realizar una función de búsqueda o de posteo.
3. Rutas: las rutas de la aplicación son las piezas de código que permiten establecer una comunicación entre dos sistemas, 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 o un recurso 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). Se cuenta con dos archivos (authentication.js e index.js) los cuales contienen todas las rutas que se van a necesitar para que por medio de las mismas la aplicación móvil pueda obtener y enviar datos. Las rutas tienen acceso a los controladores que se crearon y a las funciones que hay en cada uno de ellos. En el archivo “authentication.js” se encuentran las rutas para poder realizar el *log in* de los usuarios realizando las comprobaciones necesarias. Por otro lado, en el archivo “index,js” se encuentran todas las rutas para lo que son obtención y posteo de datos. Gracias a este archivo se puede obtener los datos de los estudiante y profesores que se necesitan para la visualización de la aplicación móvil y para el conocimiento para cada uno de los usuarios.

a. Rutas definidas para el funcionamiento de la aplicación móvil:
    i. /user-by-email/:email
    ii. /login/:email/:password
    iii. /students/:id
    iv. /student/:id
    v. /estudiante/:id
    vi. /profesor-by-id/:id
    vii. /reuniones/:id
    viii. /reunion-by-id/:id
    ix. /user-by-id/:id

x. /reuniones-nuevas/:id

xi. /create/:subject/:description/:date/:id/:studId/:mail

4. Conexión a la base de datos: en este archivo se envían todos los datos para poder conectarse a la base de datos de forma segura, en el caso de existir algún error en los datos proporcionados se mostrará un mensaje en la terminal de que la conexión no se ha podido realizar. También se tiene las diferentes relaciones que existen entre las tablas de la base de datos que pueden ser: uno-muchos, muchos-muchos o de uno-uno.

Tecnologías utilizadas en el desarrollo de la aplicación móvil

La tabla 1 resume las tecnologías y las versiones que fueron utilizadas en el desarrollo de la aplicación móvil.

Tecnología	Versión
MySQL	8.0.21 (MySQL Community Server - GPL)
Node.js	14.12.0
Ionic	5.4.16

Tabla 1. Versiones de las tecnologías utilizadas.

MySQL

Una base de datos es la encargada de almacenar y guardar datos estructurados que van a ser organizados en diferentes tablas que se relacionan de distintas formas. Por este motivo, se ha elegido utilizar el servidor de bases de datos que nos ofrece MySQL ya que es considerado el software más utilizado en el mercado por ser libre para el público. El modelo con el que trabaja es cliente/servidor basándose en un servidor SQL de multihilos que puede soportar varias librerías, herramientas, APIs y clientes. Gracias a este sistema se puede

añadir, acceder y procesar una variedad de datos de manera flexible y a gran velocidad (Barroso, 2005).

NODE JS

Es un entorno de tiempo de ejecución de JavaScript que contiene todo lo que se necesita para poder ejecutar un programa que ha sido escrito en JavaScript (Lucas, 2019). Este entorno trabaja con un modelo de entrada (solicitudes) y salida (respuestas) lo que ayuda a que el programa sea liviano y trabaje eficientemente en tiempo real, además de proporcionar varias herramientas que ayudan a satisfacer las necesidades que la mayoría de los programadores tienen. Debido a que Node.js es excelente para la creación de aplicación rápidas porque puede trabajar con varias conexiones simultáneas brindando un gran rendimiento y escalabilidad, se ha decidido utilizar para el desarrollo del proyecto.

IONIC/ANGULAR

Ionic es un framework libre de uso que se utiliza para desarrollar aplicaciones móviles híbridas que gracias a la combinación de HTML5, CSS y JavaScript se logra tener fácilmente interfaces muy amigables con los usuarios (Krama, 2020). Las aplicaciones creadas con este framework pueden ser comercializadas y descargadas en las diferentes plataformas que se conocen mundialmente: Android y IOs.

Aplicación Móvil

Para el funcionamiento de la aplicación se crearon 16 vistas que nos ayudarán a obtener información sobre los usuarios, agendar reuniones entre tutor y estudiante y generar reportes sobre reuniones y usuarios. El funcionamiento de las vistas es el siguiente:

1. Created-meetings: en esta página se mostrarán todas las reuniones que han sido creadas por un profesor.

2. First-time: en esta página el usuario ingresará el correo electrónico para que la aplicación le envié un mail para poder crear una contraseña.
3. Home: esta es la página principal en donde los usuarios pueden ingresar el correo electrónico y contraseña para iniciar sesión. En el caso de que un usuario no tenga una contraseña creada existe la opción de “Registro” para que pueda seguir todos los pasos necesarios para crear una contraseña.
4. Info-meeting: en esta página se va a encontrar la descripción sobre las reuniones que se han realizado.
5. Meeting-list: en esta página se va a poder observar el listado de todas las reuniones que le pertenezcan a un usuario.
6. New-meeting: esta página estará disponible sobre para los profesores ya que se la utilizará para crear una reunión entre el profesor y un estudiante.
7. Reset-pswd: en esta página el usuario podrá crear su contraseña para poder iniciar sesión en la aplicación.
8. Student-list: esta página estará disponible solo para profesores en donde se va a mostrar una lista de todos los estudiantes que le ha sido asignados como tutorados.
9. Student-profile: en esta página se podrá observar la información detallada de cada estudiante.
10. Tutor-profile: esta página está disponible solo si el usuario es un estudiante y en ella podrá encontrar información relevante sobre el profesor que le ha sido asignado.
11. User-profile: en esta página se encuentra la información compartida y relevante de los usuarios.
12. Accepted-meetings: en esta página se encuentra el listado de reuniones aceptadas.
13. Active-users: en esta página se encuentra el listado de usuarios activos.

14. Condition-student: en esta página se encuentra el listado de los estudiantes que estén condicionados.
15. Meeting-gpa: en esta página se encuentra el listado de los estudiantes clasificados por el gpa.
16. Rejected-meetings: en esta página se encuentra las reuniones que han sido rechazadas.

Para realizar la conexión de la aplicación con la API creada se utiliza un servicio “api-rest.service” en el cual se coloca la url en la que la API está corriendo y de esta manera poder obtener cada una de las rutas que se crearon anteriormente para obtener o ingresar datos. En este servicio se crean las todas las funciones que se van a necesitar para cada una de las vistas utilizando el URL de las rutas en la API. Este servicio está disponible para todas las vistas que se encuentren en la aplicación mediante la importación del servicio por medio de la siguiente línea de código “import {ApiRestService} from '../api-rest.service'”.

Es necesario poder guardar el usuarioId y rolId durante toda la vida útil de la aplicación, por este motivo, se creó otro servicio “route-state.service” mediante el cual se podrá tener presente en todas las rutas los datos antes mencionados haciendo que sea mucho más fácil el manejo de los diferentes componentes dentro de la aplicación móvil. Un gran ejemplo de la aplicación de este servicio es en el menú de cada uno de los usuarios ya que no todos los usuarios pueden realizar u obtener la misma información.

Pruebas y Resultados de la Funcionalidad de la Aplicación Móvil

A continuación, se mostrará el funcionamiento de la aplicación desde la vista de un profesor, estudiante y decano. Para cada uno de los ejemplos se ha tomado a tres usuarios al azar para poder presentar la diferentes vistas y funcionamiento.

Inicio de Sesión

En caso de que el usuario ya posea una contraseña dentro del sistema (en la base de datos) podrá realizar el inicio de sesión normalmente. Se ingresa el correo electrónico de la institución y la contraseña asignada a la misma para poder ingresar a la aplicación. La primera vista que se observa al iniciar sesión es la del perfil del usuario en donde se puede observar información compartida entre profesor y estudiante. En el caso de que el usuario ingrese incorrectamente el correo o la contraseña se mostrará un mensaje de aviso para que proceda a la corrección de los mismos. Las pantallas antes mencionadas se muestran en la Figura 5.

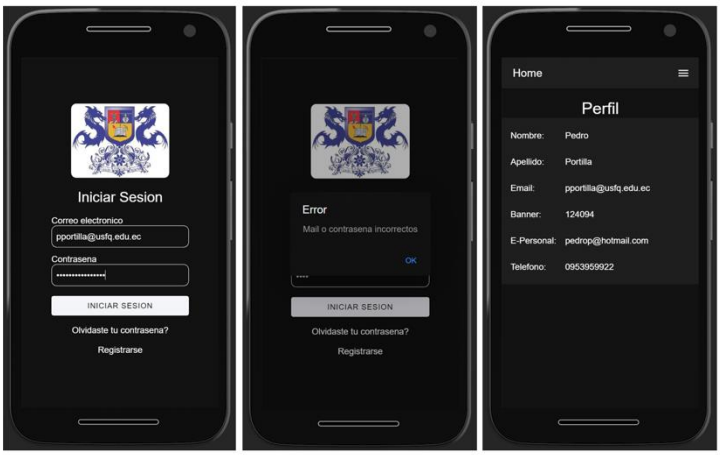

Figura 5. Vistas Inicio de Sesión.

Inicio por primera vez

Cuando un usuario no cuente con una contraseña en la base de datos deberá seguir los siguientes pasos para poder iniciar sesión en la aplicación móvil:

1. Seleccionar el link de “Registrarse”.
2. Ingresar el correo electrónico de la institución y presionar en el botón “Continuar”.
3. Se enviará un correo electrónico para poder crear una contraseña. Dar click en el botón “Restablecer Contraseña” que se muestra en la Figura 6.

4. La aplicación se redireccionará a la siguiente pantalla en donde el usuario deberá ingresar el correo electrónico, contraseña y la confirmación de la contraseña.
5. Al hacer click en el botón “Continuar” la aplicación le redireccionará a la pantalla principal para que pueda iniciar sesión normalmente.
6. Una vez iniciado sesión se mostrará la información del usuario en su perfil.

Figura 6. Correo electrónico para crear una constraseña.

Vista de profesor

Para un usuario que cumpla con el rol de profesor la aplicación servirá y se visualizará de la siguiente manera:

El usuario (profesor) ingresa a la aplicación con su cuenta de correo electrónico y su contraseña. Las opciones del menú a las que puede acceder se observan en la Figura 7.

Home: perfil del profesor en donde encontrará su información personal

Estudiantes: lista de estudiantes asignados como tutorados. En esta vista el profesor podrá seleccionar a algún estudiante para poder tener la información detallada.

Reuniones: lista de reuniones aceptadas y no aceptadas por parte de sus estudiantes. El profesor podrá seleccionar una de las reuniones para poder ver los detalles ya sea una reunión aceptada o una rechazada.

Nueva Reunión: esta vista está disponible solo para el profesor ya que este usuario será el único que puede crear una reunión con sus estudiantes asignados. Para la creación de una reunión se necesita ingresar un tema, descripción, seleccionar una fecha y hora del calendario y escoger a uno de los estudiantes.

Cerrar Sesión: acción que regresa a la pantalla de home

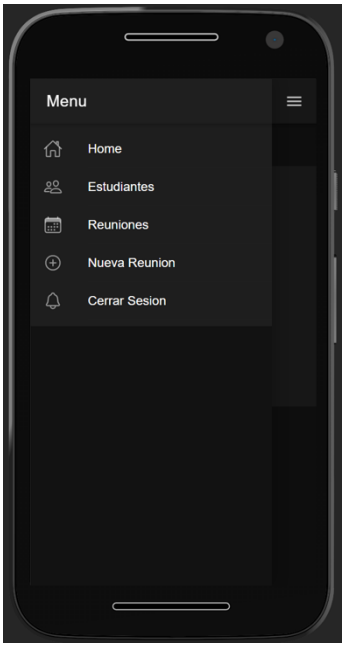

Figura 7. Vista menú del usuario profesor.

Vista de estudiante

Para un usuario que cumpla con el rol de estudiante la aplicación servirá y se visualizará de la siguiente manera:

El usuario (estudiante) ingresa a la aplicación con su cuenta de correo electrónico y su contraseña. Las opciones disponibles para este usuario en el menú se observan en la Figura 8.

Home: perfil del estudiante en donde encontrará su información personal

Tutor: esta vista estará disponible solo para los estudiantes en la cual podrán encontrar toda la información necesaria del tutor que se le ha sido asignado

Reuniones: lista de reuniones aceptadas y no aceptadas. El estudiante podrá seleccionar una de las reuniones para poder ver los detalles.

Notificaciones: esta vista está habilitada solo para estudiantes en donde podrán tener todas las reuniones que su tutor vaya creando para que puedan rechazar o aceptar la reunión. Si se acepta o se rechaza una reunión la notificación será eliminada de la pantalla.

Cerrar Sesión: acción que regresa a la pantalla de home

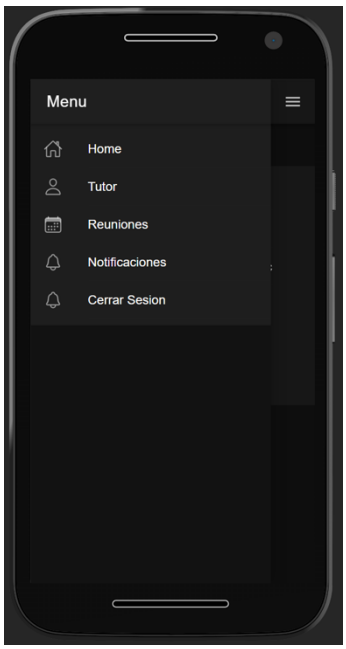

Figura 8. Vista menú del usuario estudiante.

Vista de decano

Para un usuario que cumpla con el rol de decano la aplicación servirá y se visualizará de la siguiente manera:

El usuario (decano) ingresa a la aplicación con su cuenta de correo electrónico y su contraseña. Las opciones disponibles para este usuario en el menú se observan en la Figura 9.

Home: perfil del estudiante en donde encontrará su información personal

Reuniones Aceptadas: esta vista está disponible para un usuario con rol de decano. En ella el usuario tendrá una lista de todas las reuniones que han sido aceptadas por los estudiantes con información relevante sobre cada una.

Reuniones Rechazadas: esta vista está disponible para un usuario con rol de decano. En ella el usuario tendrá una lista de todas las reuniones que han sido rechazadas por los estudiantes con información relevante sobre cada una.

Estudiantes GPA: esta vista está disponible para un usuario con rol de decano. En la misma el usuario tendrá información sobre los estudiantes clasificados por la nota total del GPA.

Usuarios Activos: esta vista está disponible para un usuario con rol de decano. Aquí se podrá encontrar información sobre todos los usuarios que estén activos dentro de la universidad.

Estudiantes condicionados: esta vista está disponible para un usuario con rol de decano. En esta vista el usuario tendrá información sobre los estudiantes que tengan un GPA menos a 2.5.

Cerrar Sesión: acción que regresa a la pantalla de home

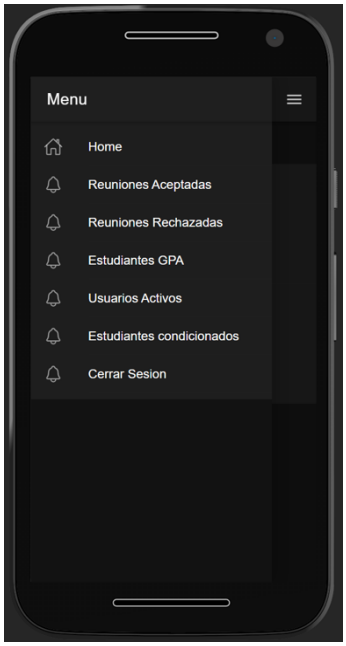

Figura 9. Vista menú del usuario decano.

Dificultades encontradas y sus soluciones

Durante el proceso de desarrollo de la aplicación se encontraron dos dificultades tanto para la obtención de datos como para el funcionamiento de los componentes de la aplicación. El primer y uno de los principales problemas que se encontraron al momento de querer obtener los datos en la aplicación fue un bloqueo de la información por parte de las políticas CORS (Cross-Origin Resource Sharing) ya que es un mecanismo de seguridad que controla todas las peticiones HTTP que se realice desde un navegador a un servidor que tenga un dominio diferente al de la página original (IBM, 2019). Para poder permitir que nuestra API entregue los datos correctos a la aplicación se tuvo que añadir el código de la Figura 10.

Gracias a esto se pudo conceder los permisos requeridos para poder extraer la información necesaria en cada una de las rutas de la API y pasar sin ningún problema a las interfaces de la aplicación móvil en IONIC.

Finalmente, el otro inconveniente encontrado en la aplicación móvil fue la diferencia de respuesta de la API con los componentes creados en IONIC ya que los componentes se

cargaban mucho más rápido que la información proveniente o en dirección a la API. Por este motivo, al momento de ejecutar la aplicación se presentaba un error ya que los objetos no estaban aún definidos. Para poder resolver este inconveniente en cada petición de la aplicación en IONIC se demoró la visualización de los componentes con un "setTimeout()" hasta que la información sea obtenida por parte de la API.

```
app.use(function(req, res, next){
  res.header('Access-Control-Allow-Origin', "*");
  res.header('Access-Control-Allow-Methods', 'GET,PUT,POST,DELETE');
  res.header('Access-Control-Allow-Headers', 'Content-Type');
  next();
})
```

Figura 10. Código para dar acceso de control.

Mejoras conocidas

Una vez concluido con el desarrollo de la aplicación móvil en su primera versión y revisadas las funcionalidades, es claro que es necesario realizar otras iteraciones de desarrollo para realizar las siguientes mejoras:

- Actualización versión API y Base de datos

  Debido a la fragmentación del proyecto en aplicación web y móvil, se cuenta con dos bases de datos y APIs diferentes. Siendo que la base de datos y API de la página web son más amplias en cuanto a estructura y funcionalidades. Por lo que es necesario que esta aplicación móvil sea actualizada hacia esa versión.

- Vista de reuniones creadas para profesores

  A fin de que los profesores puedan contar con un registro sobre las reuniones creadas, es necesario la implementación de una vista que permita enlistar

dichas reuniones. Cabe recalcar que en el presente desarrollo no se cubrió con esta vista para facilitar el flujo de datos en la aplicación.

- Implementación plugin notificaciones

  Dado que esta aplicación es híbrida y no tiene un acceso a todos los recursos de un Smartphone, no se puede enviar notificaciones al crear, aceptar o rechazar una reunión. Se investigó que IONIC brinda un plugin que permite resolver este inconveniente, por lo que se recomienda el análisis e implementación del mismo utilizando notificaciones tipo push como se detalla en la documentación de IONIC (https://ionicframework.com/docs/native/push).

- Implementación plugin calendario

  Para contar con un recordatorio de las reuniones aceptadas para profesores y estudiantes se debe implementar el plugin de Calendar que ofrece Ionic (https://ionicframework.com/docs/native/calendar) el cual facilita agregar los eventos al calendario del Smartphone.

CONCLUSIONES

El proyecto desarrollado ha permitido poner en práctica todos los conocimientos adquiridos en cada una de las materias impartidas a lo largo de la carrera universitaria.

Esta aplicación fue implementada mediante la creación de una base de datos en MySQL, una API Rest con Node.js y la vistas con IONIC, herramientas que gracias a sus beneficios han facilitado el desarrollo exitoso de la misma.

Al tratarse de una aplicación móvil creada por primera vez para la gestión de tutorías en el Colegio de Ciencias e Ingenierías de la Universidad San Francisco de Quito USFQ, no es posible realizar una comparación con otro sistema para este fin.

En base a los objetivos planteados al principio del proyecto puedo decir que se han ido cumpliendo a lo largo del desarrollo de la aplicación móvil brindando beneficios para estudiantes, profesores y decanos.

Tomando en cuenta las actualizaciones constantes de las herramientas utilizadas en este proyecto es necesario realizar un mantenimiento permanente para evitar daños en la aplicación y se recomienda a futuro implementar nuevas funcionalidades y mejoras necesarias.

REFERENCIAS BIBLIOGRÁFICAS

Barroso, A. (Septiembre de 2005). *e-REdING.* Obtenido de http://bibing.us.es/proyectos/abreproy/11096/fichero/Memoria%252F04+Cap%C3%ADtulo+4+Base+de+Datos+mySQL.pdf+

IBM. (21 de Mayo de 2019). *Cloud IBM*. Obtenido de https://cloud.ibm.com/docs/CDN?topic=CDN-cors-and-cors-requests-through-your-cdn&locale=es

Krama. (29 de Abril de 2020). *Krama.es*. Obtenido de https://www.krama.es/blog-20-04-29-que-es-ionic.html

Lucas. (4 de Septiembre de 2019). *OpenWebinars*. Obtenido de https://openwebinars.net/blog/que-es-nodejs/

Rouse, M. (Agosto de 2020). *SearchAppArchitecture*. Obtenido de https://searchapparchitecture.techtarget.com/definition/API-endpoint#:~:text=An%20API%20endpoint%20is%20a,server%20and%20receiving%20a%20response