

**Quinta Entrega**

**Documentación**

**Programación de Aplicaciones de Escritorio**

**Integrantes de equipo:**

* Kury Josafath Vázquez Correa, 727571
* Ana María Anaya García, 732638

**Profesor:** Francisco Javier Sevilla Medina

**Fecha de entrega:** 24 de noviembre de 2023

**Introducción**

A diario en ITESO se organizan una gran cantidad de eventos, se ofrecen talleres, se imparten cursos, conferencias, entre muchas otras actividades extracurriculares, instauradas por los diferentes departamentos y sociedades estudiantiles de las diferentes carreras que existen en la institución. Sin embargo, los canales de difusión por los que se dan a conocer estos eventos no son los óptimos, como el correo institucional o carteles físicos esparcidos por el campus, causando que muchas de las actividades pasen desapercibidas o reciban muy poca audiencia. Es por esto que nuestra idea de proyecto es una plataforma en donde se desplieguen todos los eventos que existen en el campus y en la que los alumnos puedan registrarse directamente para asistir.



**Descripción del proyecto**

Para acceder a la plataforma el usuario deberá registrarse con una cuenta de correo electrónico y tendrá la opción de hacerlo directamente con su cuenta de Google, podrá escoger la universidad u organización de la que quiere formar parte y su papel dentro de la misma, maestro, alumno, colaborador, etc.. Así como la opción de seleccionar el tipo de eventos que le gustaría ser notificado (académicos, deportivos, culturales, etc.). De la misma forma cada usuario deberá registrarse indicando su nombre, correo electrónico, carrera, entre algunos otros datos que servirán para identificar los eventos que serán desplegados en su perfil.

Después de iniciar sesión con su cuenta personal de Google, la plataforma de eventos mostraría en pantalla principal todos los eventos existentes en una franja determinada de tiempo (definida por el usuario) para que los usuarios puedan registrarse a los que deseen asistir o simplemente estar informados de su existencia. Dentro de cada evento estaría la información detallada del mismo, así como el link de registro, participantes, organizador, opción de agregarlo al calendario personal de Google y datos extra que sean necesarios para el evento.

Dependiendo del tipo de evento, ya sea interno o externo, habrá la opción de generar un código QR para los participantes, para ser presentado y escaneado el día del evento por los organizadores, esto con la intención tener los datos de asistencia y participación de la actividad. El evento también contará con una sección de comentarios y calificación, donde los usuarios podrán subir imágenes del evento, así como su reseña del mismo, como forma de retroalimentación de los participantes y así tomar decisiones para los futuros eventos.

Para dar de alta un evento nuevo se tendrá que iniciar sesión con una cuenta de cliente que te lo permita. Para crear la nueva actividad se requerirá la información como el título, la descripción, seleccionar las carreras, departamentos o público en general para los que va dirigido el evento, tipo de registro, link de registro, fecha de vencimiento, datos o links extra, así como tipo de evento (cultural, académico, deportivo), etc..

Cada usuario tendrá un apartado de perfil donde podrá modificar sus datos, y también contará con un apartado en donde se muestren todos los eventos a los que ha asistido, y a los que está registrado y en donde podrá acceder a los QR que deberá presentar al ingreso.

**Stack de Tecnologías**

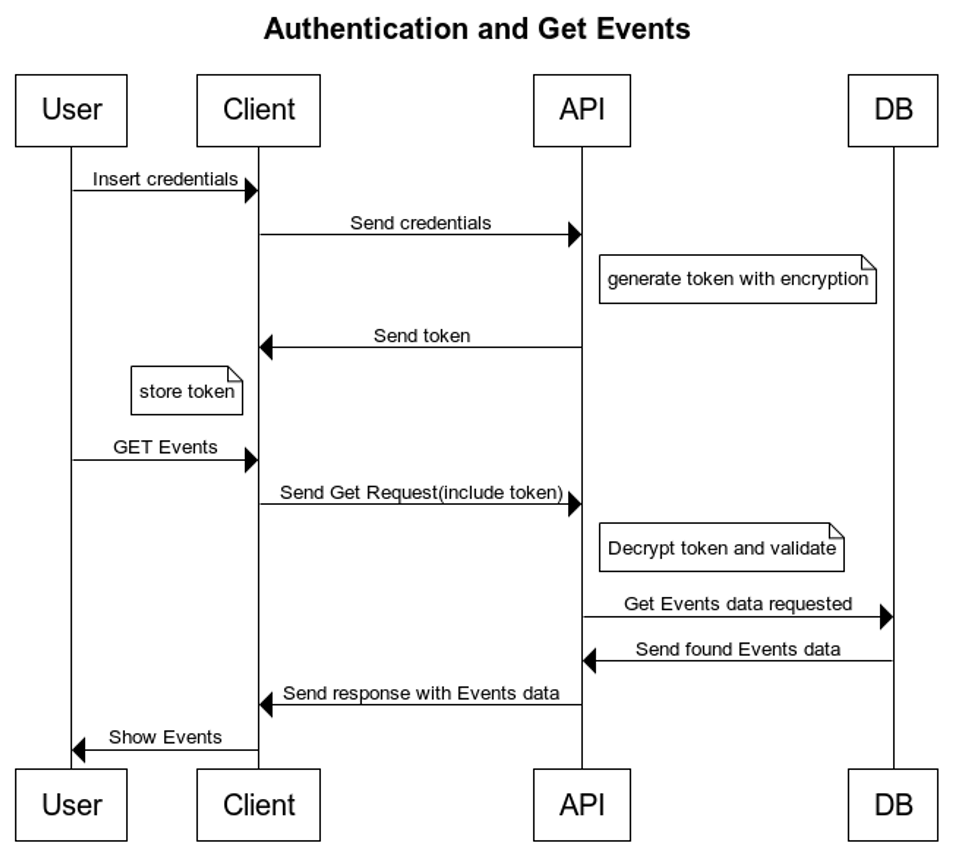
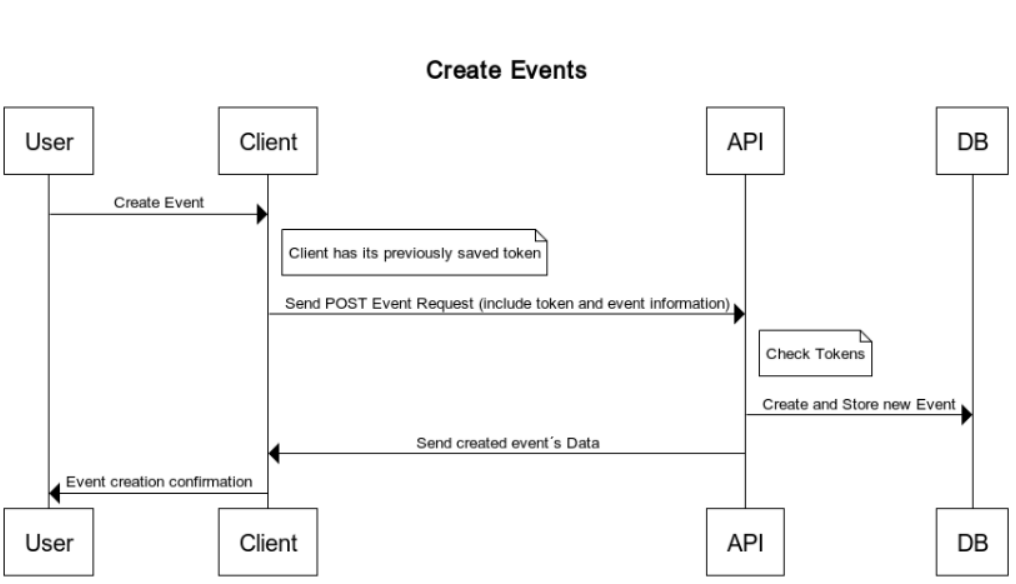
* Front
  + Angular
* Back
  + Node js
  + Express
  + Multer
  + Mongoose
  + Swagger
* Base de datos
  + MongoDB
* Integración con 3ros
  + Inicio de sesión con cuenta de Google
  + Calendario de Google
  + Librería para generación de QRs
* Cloud
  + S3 Bucket

**Alcance**

* Passport (Google): Se tendrá la opción de manejar el inicio de sesión de los usuarios mediante su cuenta de Gmail.
* Subir y descargar imágenes mediante un bucket de S3 para que los organizadores suban imágenes alusivas a los eventos antes de su fecha de realización, y de igual forma suban fotografías posteriores al evento para que los usuarios puedan descargar las que deseen.
* Se utilizará una API para la generación de QRs como forma de boleto para entrar a los eventos.
* Extensión con Google Calendar para añadir los eventos a los que el usuario asistirá a su calendario personal, con la misma cuenta que inició sesión.

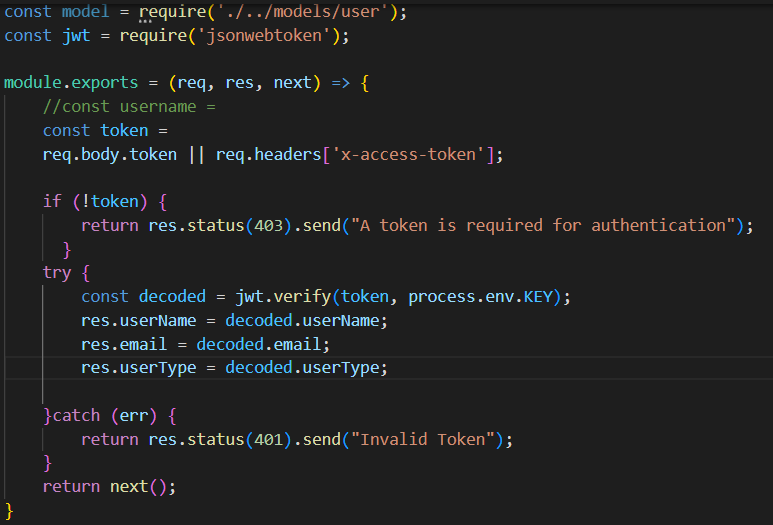
**Diagramas**

**(Autenticación Actualizado)**

****

**Últimas adiciones:**

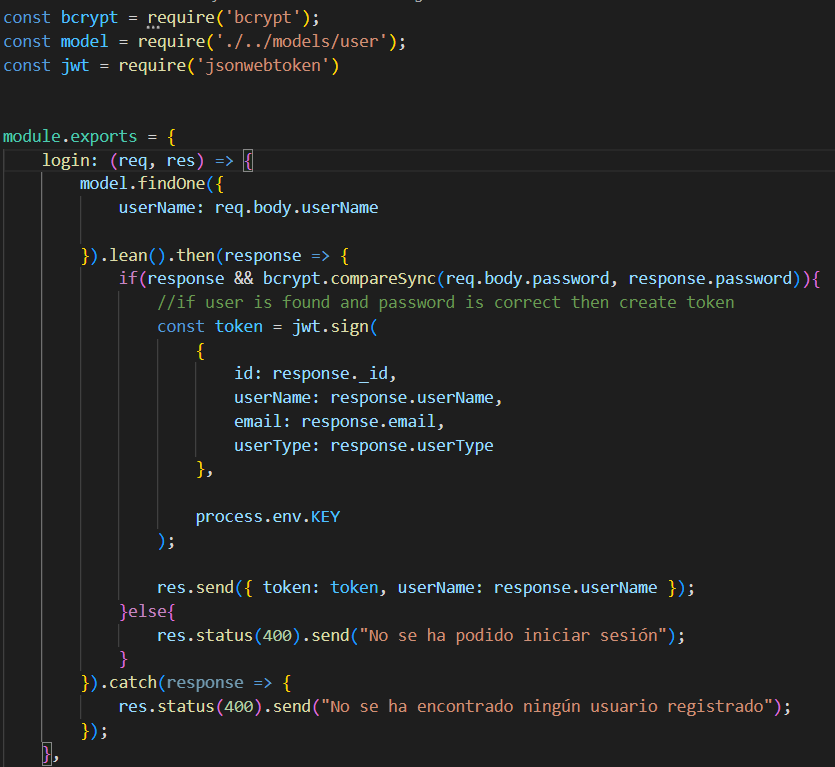
* **Middleware de autenticación:**

****

* **Middleware para tipo de usuario:**

****

* **Función para Log in en controlador “Users”:**

****

**Entidades**

* Usuario
  + Registro como ‘usuario’
  + Inicio de sesión
  + Ver y editar el perfil
  + Registro para eventos
  + Ver eventos a los que está registrado
  + Dar reseñas a eventos que ha asistido
  + Obtener QR de eventos
* Cliente
  + Inicio de sesión
  + Registro como ‘organización’
  + Dar de alta y editar eventos
  + Ver y editar perfil de ‘organización’
* Administrador
  + Inicio de sesión
  + Dar de alta/baja a ‘organización’
  + Dar de alta/baja a ‘usuarios’
  + Habilitar y deshabilitar usuarios y eventos
  + Dar de alta/baja/editar eventos

**Diagrama**

Diagrama, Esquemático

Descripción generada automáticamente

**Planeación**

* Setup
* Documentación
* Autenticación
* Registro de usuarios
* Diversos CRUDs
* Pruebas
* Despliegue



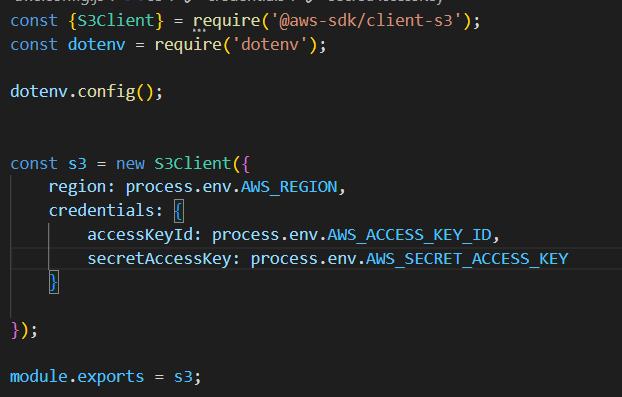
**Autenticación con Google**

**Diagrama

Descripción generada automáticamente**

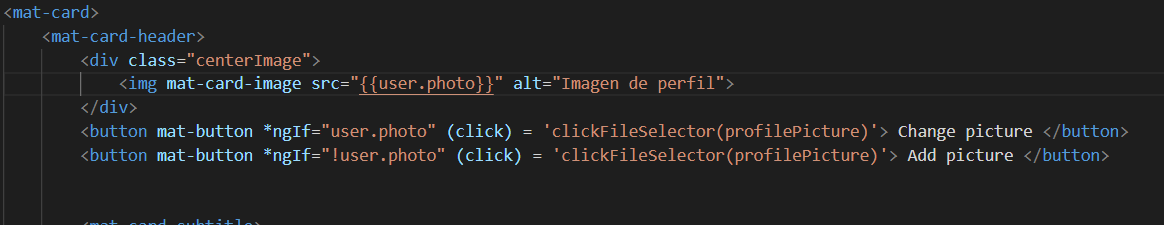
En caso que el usuario no este registrado previamente no podrá iniciar sesión.

**Carga de Archivos Multer-S3:**

Se añadió la funcionalidad de subir imágenes con las extensiones válidas .jpg, .jpeg y .png para que el usuario puede subir y cambiar su foto de perfil. Se creó un usuario de IAM y un Bucket para poder obtener las credenciales adecuadas para hacer la carga de archivos deseada. Dentro del bucket se diseñó que la ruta para que los archivos se subiera fuera de /users/idUsuario/imagen, para que así cada usuario tuviera su carpeta con su foto de perfil y fuera bastante sencillo y directo acceder a ellas. Se creó el endpoint para hacer esa carga de archivos, el cual además actualiza el atributo de “photo” en el usuario en cuestión con la fuente de la imagen que ahora puede ser abierta en el navegador, de este manera, ese atributo es luego utilizado en el html como fuente para ser desplegada en pantalla.







**Comunicación en tiempo real con Socket.io**

* Configuración del Servidor con Socket.IO

En el lado del servidor, es necesario establecer una conexión con Socket.IO.

const socketIo = require('socket.io');

//configurando sockets

const io = socketIo(server, {

    cors: {

        origin: '\*',

        methods: ['GET', 'POST']

        }

});

io.on('connection', (socket) => {

    //console.log('alguien se conecto');

    socket.on('userAttendEvent', (data) => {

        //aqui se guardaria en la base de datos

        //console.log('Llego una notificacion', data);

        socket.broadcast.emit('notifUser', data);

    });

});

* Configuración del Cliente Angular

Cuando un usuario agrega un evento a sus eventos. Función del lado del user que atenderá el evento.

attendEvent(event: Event){

    this.eventService.updateUserEventRequest(event, this.userLogin.\_id!).subscribe({

      next: (response: Event) => {

        this.socket.emit('userAttendEvent', response);

        //console.log(response);

      },

      error: () => {

        this.error = true;

      }

    });

  }

Función para que la org reciba la notificación.

ngOnInit(): void {

    this.socket.on('notifUser', (event: Event) => {

      if(event.idOrganizer === this.userId){

        //console.log('Un usuario nuevo atenderá tu evento.');

        this.snackBar.notificationSnackBar('Un usuario nuevo atenderá tu evento - ' + event.name);

      }

    })

  }