Universidad Interamericana de Puerto Rico
Recinto de Arecibo

Control de Asistencia para Escuelas

Grupo de: Bryan Vega, Ryan Vega y Sebastián Galeano

COMP3400

Profesor Dastas

Tabla de Contenido:

Contents

Resumen del Proyecto:

El proyecto consiste en desarrollar una aplicación móvil y web que permita registrar de forma automatizada la asistencia de estudiantes, profesores y personal administrativo en una escuela. El objetivo principal es agilizar y mejorar el proceso de registro de asistencia, reduciendo errores y proporcionando información útil para la toma de decisiones por parte de la administración escolar. El proyecto se realizará utilizando las herramientas y todo lo que se ha aprendido en la clase, como la creación de una base de datos en MariaDB, el uso de una maquina virtual del recinto y el uso de la herramienta Linux para el desarrollo y operación de software.

Objetivos:

El objetivo principal es automatizar el registro de la asistencia de estudiantes, profesores y personal administrativo para mejorar la eficiencia y precisión en la gestión escolar. Además, se busca reducir errores en el registro manual y ahorrar tiempo tanto para el personal administrativo como para los docentes. La aplicación también tiene como objetivo enviar notificaciones automáticas a los padres en caso de ausencias y facilitar la generación de reportes detallados para una toma de decisiones informada por parte de la administración escolar. En resumen, el proyecto busca optimizar el proceso de control de asistencia en la escuela mediante el uso de tecnología y funcionalidades específicas.

Lista de los miembros del equipo (grupo de trabajo) y rol:

BRS:

- Bryan Vega – Scrum Master
- Ryan Vega – Development Team
- Sebastián Galeano – Product Owner, Development Team

Product Log:

En este Proyecto se busco realizar varias tareas dentro un tiempo específico:

1era Semana:

La primera semana nos enfocamos en la creación del código y base de datos junto a la creación de el documento de Word.

Miembro que Realizo la tarea:	Tarea Realizada:	Dia trabajada:
Bryan Vega	Creación del documento en Word	8/5/24
Sebastián Galeano	Desarrollo del Código	8/5/24,
Ryan Vega	Creación de Diagramas y tablas	8/5/24

2da Semana:

En la segunda semana se trabajo con la creación de tablas y diagramas como la de los Diagramas ER y la del Work Breakdown Structure. También trabajamos con la creación de la aplicación utilizando HTML.

Miembro que Realizo la tarea:	Tarea Realizada:	Dia trabajada:
Bryan Vega	Creación del Work Breakdown Structure, y Diagrama de la Arquitectura del Software: Ayudo a completar el código de la aplicación y finalizar el documento de Word.	13/5/24 15/5/24
Sebastián Galeano	Desarrollo del código en HTML y edición de código previo.	13/5/24, 15/5/24
Ryan Vega	Creación de Diagramas ER y ayudar con el código HTML Ayudo con la finalización del WBS	13/5/24 15/5/24

Work Breakdown Structure (WBS)

1. Planificación y Diseño

- Definir los requisitos de la aplicación
- Crear un documento de especificaciones
- Diseñar la arquitectura de la aplicación
- Crear el plan de desarrollo

2. Desarrollo de Backend

- Configurar el entorno de desarrollo
- Implementar la base de datos
- Integrar APIs externas (si es necesario)
- Realizar pruebas unitarias

3. Desarrollo de Frontend

- Diseñar la interfaz de usuario (UI)
- Desarrollar la interfaz de usuario (HTML, CSS, JavaScript)
- Integrar con el backend
- Realizar pruebas de interfaz de usuario

4. Implementación de Funcionalidades de Asistencia

- Desarrollar el sistema de registro de asistencia
- Implementar la notificación de ausencias
- Desarrollar la gestión de ausencias y justificaciones
- Integrar con el calendario escolar

5. Desarrollo de Funcionalidades para Administradores

- Crear el panel de administración
- Desarrollar la gestión de usuarios (estudiantes, maestros, administradores)
- Implementar la generación de informes y estadísticas

6. Pruebas y Ajustes

- Realizar pruebas de integración
- Realizar pruebas de aceptación del usuario
- Corregir errores y realizar ajustes según la retroalimentación

7. Despliegue y Puesta en Marcha

- Preparar el entorno de producción
- Desplegar la aplicación en servidores
- Configurar la seguridad y los permisos de acceso
- Capacitar al personal escolar en el uso de la aplicación

8. Soporte y Mantenimiento

- Establecer un proceso de soporte técnico
- Monitorizar el rendimiento y la seguridad de la aplicación
- Realizar actualizaciones y mejoras según sea necesario
- Proporcionar soporte continuo a los usuarios

Diseños del Proyecto:

Diagrama ER:

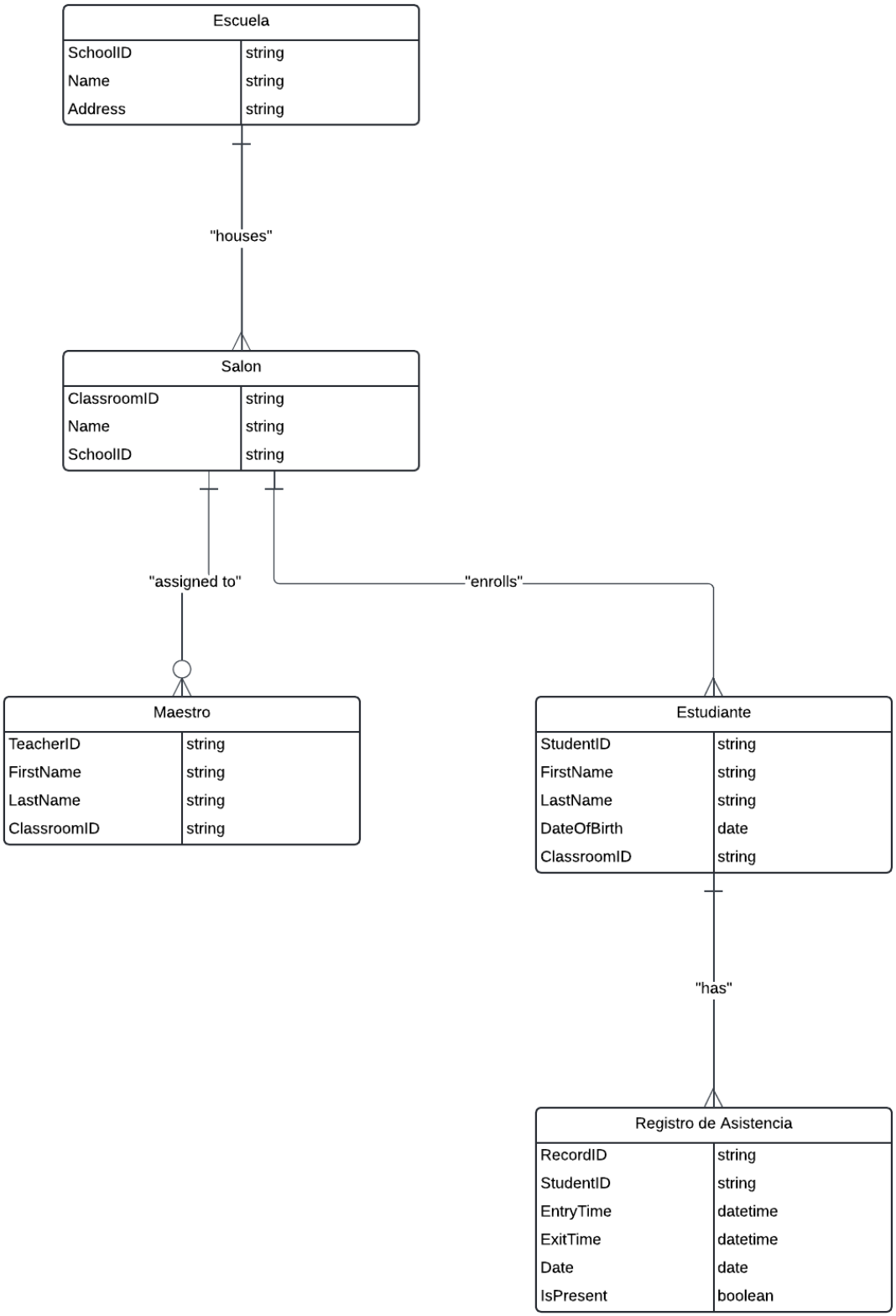
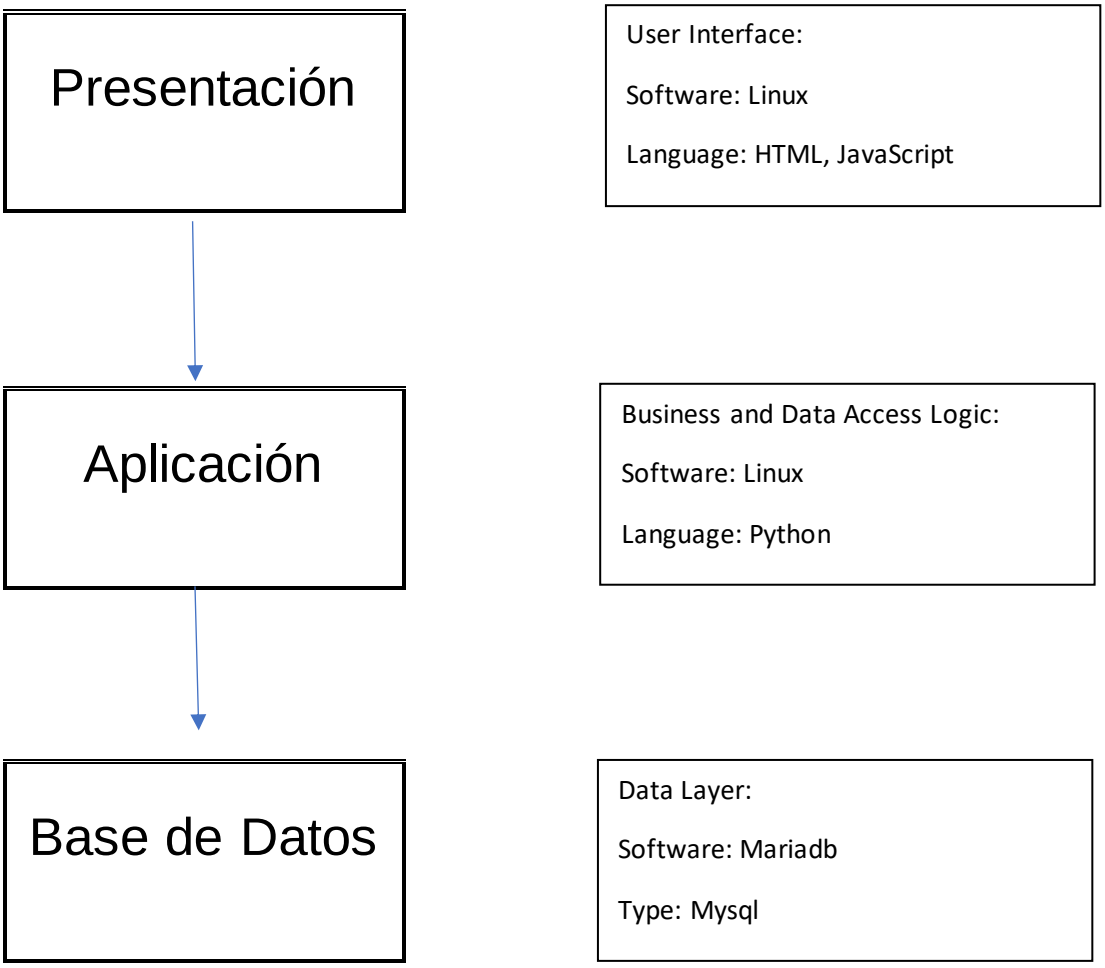


Diagrama de la Arquitectura del Software:



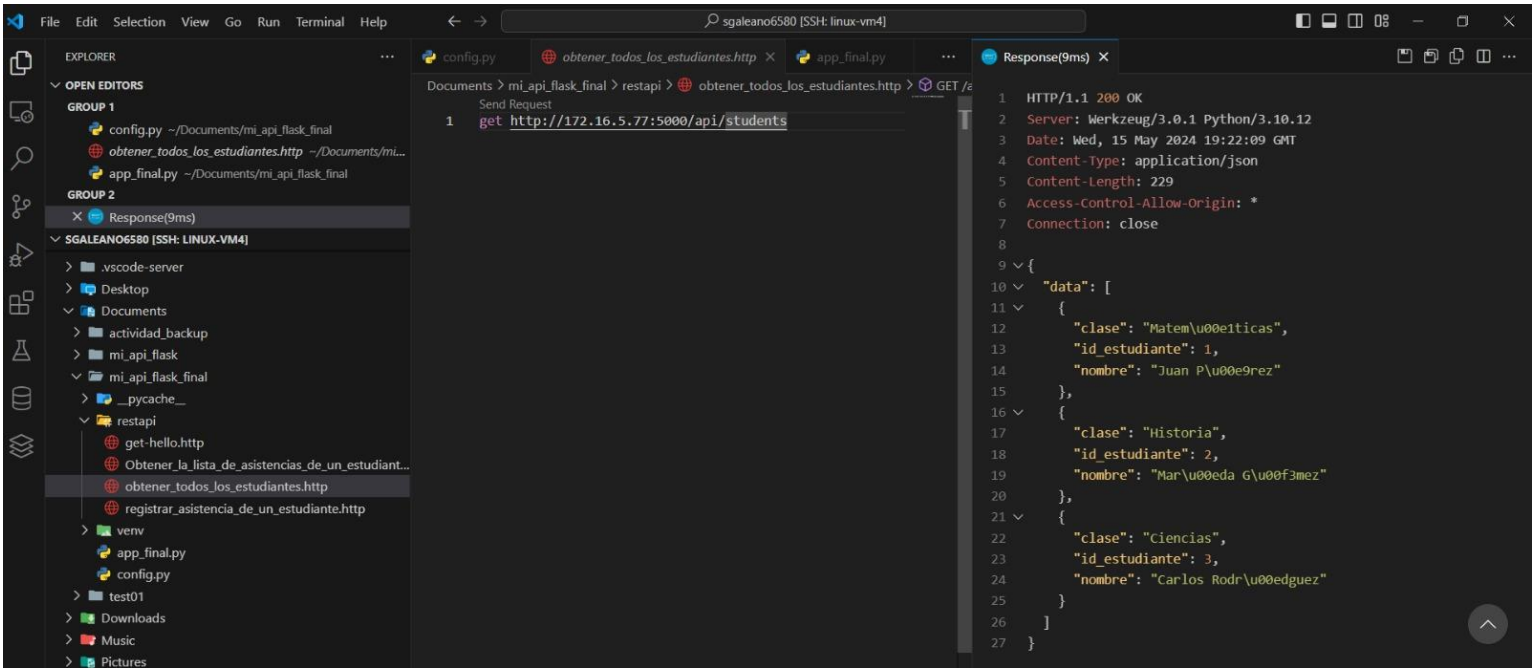
Codigo:

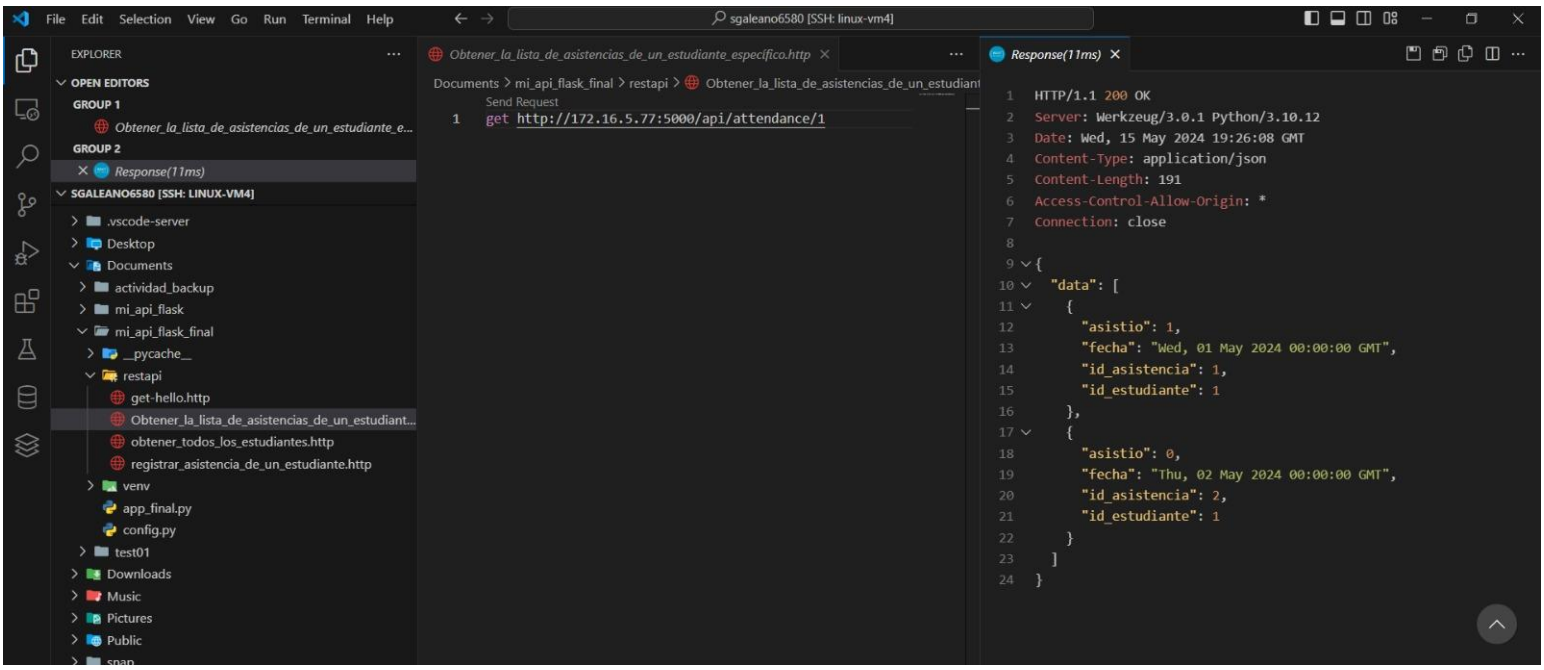
App_Final.py

```
from flask import Flask, jsonify, request
import mariadb
import sys

# Importar configuración de acceso a la base de datos
from config import DATABASE_CONFIG

app = Flask(_name_)

try:
    conn = mariadb.connect(**DATABASE_CONFIG)
except mariadb.Error as e:
    print(f"Error on connection: {e}")
    sys.exit(1)

cursor = conn.cursor()

# Ruta de prueba
@app.route('/api/hello', methods=['GET'])
def hello_world():
    return jsonify({'message': '¡Hola, mundo con Flask!'})

# Obtener todos los estudiantes
@app.route('/api/students', methods=['GET'])
def get_students():
    cursor.execute("SELECT * FROM Estudiantes")
    students = cursor.fetchall()
    student_list = []
    for student in students:
        student_list.append({
            "id_estudiante": student[0],
            "nombre": student[1],
            "clase": student[2]
        })
    response = jsonify({"data": student_list})
    response.headers.add("Content-type", "application/json")
    response.headers.add("Access-Control-Allow-Origin", "*")
    return response

# Registrar la asistencia de un estudiante
```

```python
@app.route('/api/attendance', methods=['POST'])
def register_attendance():
    datos = request.json
    id_estudiante = datos.get('id_estudiante')
    fecha = datos.get('fecha')
    asistio = datos.get('asistio')

    cursor.execute("INSERT INTO Asistencias (id_estudiante, fecha, asistio)
VALUES (?, ?, ?)",
                   (id_estudiante, fecha, asistio))
    conn.commit()

    response = {"message": "Attendance recorded"}
    return jsonify(response), 200

# Obtener la lista de asistencias de un estudiante específico
@app.route('/api/attendance/<int:id_estudiante>', methods=['GET'])
def get_student_attendance(id_estudiante):
    cursor.execute("SELECT * FROM Asistencias WHERE id_estudiante = ?",
(id_estudiante,))
    attendances = cursor.fetchall()
    attendance_list = []
    for attendance in attendances:
        attendance_list.append({
            "id_asistencia": attendance[0],
            "id_estudiante": attendance[1],
            "fecha": attendance[2],
            "asistio": attendance[3]
        })
    response = jsonify({"data": attendance_list})
    response.headers.add("Content-type", "application/json")
    response.headers.add("Access-Control-Allow-Origin", "*")
    return response

if _name_ == '_main_':
    app.run(host='0.0.0.0', port=5000)
```

“Screen Shoot” de las funcionalidades de la aplicación:





Preguntas:

1. ¿Qué salió bien? ¿Por qué?

   Unas de las cosas que nos salió bien fue el trabajo en equipo y la división de tareas ya que hemos trabajado como grupo antes y sabemos las fuerzas de cada uno.

2. ¿Qué salió mal? ¿Por qué?

Nos dio un poco de trabaja ver como crear los diagramas y un poco de trabajo con el código en HTML ya que nos hacia falta mas practica con las creaciones de aplicaciones utilizando HTML

3. ¿Qué aprendió?

   Hemos aprendido como bregar con creación de base de datos en un terminal del recinto, la creación de APIS y utilizar la herramienta de Linux.

4. ¿Qué requiere fortalecer?

   Pienso que algo que nos falta fortalecer el coding en HTML y el uso de Linux.

5. ¿Qué beneficios pudiera aportar la metodología AGILE SCRUM en el proyecto?

   La metodología Agile Scrum podría aportar beneficios como flexibilidad, entregas incrementales, colaboración efectiva, visibilidad y mejora continua al proyecto de una aplicación para Control de Asistencia en escuelas.

6. ¿Cree que debería haber alguien que distribuya el trabajo en su grupo? ¿Por qué?

   Diría que no es necesario que alguien distribuya el trabajo en el grupo porque si se crea un roadmap con todo lo necesario para completar el proyecto solo tenemos que ir completando las tareas para ir progresando con el proyecto de manera eficiente.

Codigo HTML:

Index.html:

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Asistencia de Estudiantes</title>
  <link rel="stylesheet" href="styles.css">
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 20px;
    }

    h1 {
      text-align: center;
      color: #333;
      margin-bottom: 30px;
    }

    #students-list {
      max-width: 600px;
      margin: 0 auto;
    }
```

```
      .student {
        background-color: #fff;
        border-radius: 5px;
        box-shadow: 0 2px 5px rgba(0, 0, 0, 0.1);
        padding: 20px;
        margin-bottom: 20px;
      }

      .student h3 {
        color: #007bff;
        margin-top: 0;
 }

      .student p {
        margin: 5px 0;
        color: #666;
      }
    </style>
</head>
<body>
    <div class="container">
      <h1>Registro de Asistencia de Estudiantes</h1>
      <div id="students-list"></div>
    </div>

    <script src="/js/script.js"></script>
</body>
</html>
```

Script.js:

```
document.addEventListener('DOMContentLoaded', function () {
    // Hacer una solicitud GET a la API para obtener la lista de estudiantes
    fetch('http://127.16.5.77:5000/api/students')
        .then(response => response.json())
        .then(data => {
            const studentsList = document.getElementById('students-list');

            // Iterar sobre la lista de estudiantes y crear elementos HTML para mostrarlos
            data.data.forEach(student => {
                const studentDiv = document.createElement('div');
                studentDiv.classList.add('student');
                studentDiv.innerHTML = `
                    <h3>${student.nombre}</h3>
                    <p>ID: ${student.id_estudiante}</p>
                    <p>Clase: ${student.clase}</p>
                `;
                studentsList.appendChild(studentDiv);
            });
        })
        .catch(error => console.error('Error:', error));
});
```

Styles.css:

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 20px;
}

h1 {
    text-align: center;
}

.student {
    margin-bottom: 10px;
}
```

Mariadb Database Code:

ControlAsistencias:

```
CREATE DATABASE ControlAsistencias;
USE ControlAsistencias;

CREATE TABLE Estudiantes (
    id_estudiante INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(255),
    clase VARCHAR(50)
);

CREATE TABLE Profesores (
    id_profesor INT PRIMARY KEY AUTO_INCREMENT,
    nombre VARCHAR(255),
    asignatura VARCHAR(100),
    horario VARCHAR(100)
);

CREATE TABLE Asistencias (
    id_asistencia INT PRIMARY KEY AUTO_INCREMENT,
    id_estudiante INT,
    fecha DATE,
    asistio BOOLEAN,
    FOREIGN KEY (id_estudiante) REFERENCES Estudiantes(id_estudiante)
);
INSERT INTO Estudiantes (nombre, clase) VALUES
('Juan Pérez', 'Matemáticas'),
('María Gómez', 'Historia'),
('Carlos Rodríguez', 'Ciencias');
```

```
INSERT INTO Profesores (nombre, asignatura, horario) VALUES
('Ana López', 'Matemáticas', 'Lunes y Miércoles 8:00 - 10:00'),
('Pedro Martínez', 'Historia', 'Martes y Jueves 10:00 - 12:00'),
('Laura Fernández', 'Ciencias', 'Lunes y Miércoles 13:00 - 15:00');

INSERT INTO Asistencias (id_estudiante, fecha, asistio) VALUES
(1, '2024-05-01', true),
(1, '2024-05-02', false),
(2, '2024-05-01', true),
(2, '2024-05-02', true),
(3, '2024-05-01', true),
(3, '2024-05-02', false);
```

GitHub:

https://github.com/Vegatwin01/Proyecto-Final