MANUAL TÉCNICO

Prototipo funcional de un nuevo sistema que se desea implementar llamado “Iniciativa Control De Tráfico Inteligente” el cual tiene el propósito de regular el tráfico en una urbanización basado en la densidad de autos en los carriles para ajustar los tiempos de las luces de forma dinámica. Se le solicita que realice el prototipo para dos tipos de intersección, se deberá de almacenar un registro histórico para detectar las horas pico en el tráfico así como implementar un sistema de alerta en caso de accidentes o brindar prioridad para vehículos de emergencia. Se les solicita que construyan un sistema inteligente de tráfico aplicado a dos tipos de intersecciones en las cuales se debe regular la densidad de autos en cada carril para ajustar el tiempo de las luces verde y rojo de forma dinámica en función de la cantidad de autos que existan en cada carril. Deberá de fabricar estas intersecciones en forma de prototipo, las flechas delgadas indican los posibles caminos que puede tomar un vehículo dependiendo del carril de donde provenga, en base a esto deben de emular el comportamiento del semáforo con luces led, representando las tres luces de un semáforo, rojo, naranja y verde. La densidad del carril se pueden medir a través de sensores infrarrojos o de un sensor ultrasónico, queda a discreción del grupo y su creatividad como realizar esta medición, en el cual, al detectar un 75% del carril ocupado, deberá activar la luz verde para que disminuya este valor, no es necesario que el flujo vehicular llegue a una densidad de cero, el valor mínimo para cambiar la prioridad del carril es del 25% para que baje su prioridad en la luz verde. Esta densidad máxima del carril es aplicable a todos los carriles, por lo que deberá desarrollar un algoritmo capaz de gestionar los tiempos de forma dinámica y establecer prioridades en los carriles con una densidad vehicular alta.

Tecnologías Utilizadas

- Lenguaje: JavaScript
- React para la parte del frontend
- Python
- Atlas para crear el cluster de la base de datos de MongoDB
- Proteus

Arquitectura del Backend

```
backend/
├── database/
│   ├── MongoDB
├── Models/
│   ├── densidadVehiculas.js
├── app.js
├── condigSemaforo.py
├── mongo.py
├── semaforo.py
├── send_mongo.py
├── send_to_mongo.py
```

Arquitectura del Frontend

```
frontend/
├── public/
├── src/
│   ├── assets
│   ├── App.css
│   ├── LinesChart.jsx
│   ├── index.cs
│   ├── main.jsx
├── vite.config.js
```

Librerías Utilizadas

Librerías sobre un entorno de Python

pip3 install paho-mqtt
pip3 install --upgrade paho-mqtt
pip3 install rpi-lgpio
pip3 install pymongo

Documentacion sobre los pines RGB:
https://projects.raspberrypi.org/en/projects/introduction-to-the-pico/8

Documentacion sobre el sensor IR:
https://diyprojectslab.com/ir-sensor-with-raspberry-pi-pico-micropytho/

Librería para conectar la api a la base de datos en Atlas
python -m pip install "pymongo[srv]"

Documentación sobre conexión al Cluster de Atlas
https://www.mongodb.com/resources/products/fundamentals/clusters

Librerías para componentes en React
npm install @mui/material @emotion/react @emotion/styled

Documentación sobre Material UI
https://mui.com/material-ui/getting-started/

Costo del Prototipo

1	Pantalla LCD 1602 Color azul	Q34.00 c/u	Q34.00
2	Módulo sensor de obstáculos infrarrojo	Q18.00 c/u	Q36.00
3	Led RGB catodo común ultra brillante 5 mm	Q2.00 c/u	Q06.00
1	Cable de datos USB a micro USB	Q15.00 c/u	Q15.00
6	Resistencias 220	Q00.05 c/u	Q03.00
	Total		Q 94.00

Consideraciones y Restricciones para la Construcción del Prototipo

1. Materiales y Componentes:
    - Uso de Raspberry Pi 3 como unidad central de procesamiento.
    - Sensores de proximidad infrarrojos para la detección de vehículos.
    - LEDs para la emulación de las luces del semáforo.
    - Botón de emergencia para la generación de alerta.
    - Conectividad a Internet para la transmisión de datos a la plataforma web.
2. Software y Comunicación:
    - Programación en Python para la gestión de entradas y salidas digitales.
    - Implementación de Script WebSocket para la comunicación en tiempo real.
    - Uso de MQTT mediante HiveMQ para la comunicación con el sitio web.
    - Base de datos en MongoDB para almacenamiento.
3. Restricciones:
    - El sistema opera con una latencia mínima para el cambio de luces basado en la densidad vehicular.
    - La interfaz web es intuitiva y accesible para los administradores.
    - El prototipo fue encapsulado para proteger los componentes electrónicos.

Solución aplicada para el manejo de los tiempos del semáforo de forma dinámica

1. Detección de Vehículo:
    - Los sensores instalados en cada carril detectan la presencia de vehículos y calculan la ocupación del carril en porcentaje.
    - Si la ocupación de un carril supera el 75%, el sistema prioriza el tiempo de luz verde en dicho carril.
    - Cuando la ocupación baja del 25%, el carril pierde prioridad y se ajusta la duración del semáforo de forma proporcional.
2. Algoritmo de Asignación de Tiempos:
    - Se implementó un algoritmo de reparto de tiempos basado en ponderaciones.
    - Cada carril recibe una prioridad calculada con base en su nivel de ocupación.
    - El sistema balancea los tiempos de luz verde y rojo asegurando una distribución equitativa del flujo vehicular.
3. Interacción con la Plataforma Web:
    - Se almacenó un registro histórico de los datos en MongoDB.
    - Se generaron reportes de horas pico y se presentan los datos en el dashboard web.
    - En caso de emergencia, el sistema permite el bloqueo de los semáforos y envío de alertas a través de Twilio.