Informe:

Esta práctica se basa en un motor de simulación de tráfico en tiempo real, que representa una ciudad con calles, semáforos y vehículos. Se han utilizado técnicas de programación concurrente y distribuida, junto con una interfaz gráfica en Pygame.

Se ha utilizado RabbitMQ para el enfoque distribuido, a través de la librería aio-pika. La arquitectura está basada en dos zonas, la zona_norte que se encarga de enviar vehículos y la zona_sur, que se encarga de recibirlos y mostrarlos gráficamente. El objetivo es simular un sistema distribuido donde cada parte del mapa puede estar gestionada por una instancia diferente.

Las tareas que se ejecutan concurrentemente son:

El movimiento de los vehículos.

El cambio de estado de los semáforos.

La recepción y visualización de los vehículos en tiempo real.

La forma en la que se hace uso de la concurrencia es la siguiente:
Se lanza un hilo desde zona_sur separado para la interfaz en Pygame, que escucha a su vez una corutina con asyncio para recibir mensajes desde RabbitMQ. Los mensajes contienen vehículos que se añaden a una cola y luego se visualizan en pantalla.

El movimiento de cada vehículo está controlado por su dirección y velocidad, y se detiene si el semáforo correspondiente con el que se cruza está en rojo (alternan entre verde y rojo).

Cada vez que se añade un vehículo a la cola, lo hace con una posición de entrada y una trayectoria asignada aleatoriamente, coincidente con las calles.

Requerimientos:

Para instalar y ejecutar el sistema:
Instalar RabbitMQ, Pygame y aio-pika.

Iniciar zona_sur.py y zona_norte.py en terminales divididas.

Posibles extensiones:

Introducir rutas programadas para cada vehículo con giros de dirección.

Sincronizar los semáforos entre zonas para simular un tráfico más realista.

Práctica realizada por Lucía Bermejo y Guillermo Salicrú.