# Ruta Fácil: Soluciones Inteligentes para el Uso del Sistema de Transporte

**Melissa Galeano Ruiz, Laura Cortés Camello, Nicolás Guarín Gaviria, Keni Brandon Gracia**

**Resumen:** Moverse por la ciudad de Medellín puede ser un desafío, especialmente para quienes llegan por primera vez y no conocen las rutas de transporte disponibles. La falta de información clara y accesible puede generar confusión, retrasos e incluso trayectos innecesariamente largos.

Nuestro proyecto busca solucionar este problema mediante una inteligencia artificial capaz de procesar lenguaje natural. A través de una simple pregunta, como "Estoy en [origen] y quiero ir a [destino], ¿cuál es la mejor ruta?", el sistema analizará las opciones de transporte público en Medellín y recomendará la mejor alternativa.

Esta interfaz inteligente facilitará la movilidad tanto para habitantes como para visitantes, brindando respuestas rápidas y precisas sobre rutas de buses y metro, mejorando así la experiencia de desplazamiento en la ciudad. Se complementará con información específica del metro de Medellín, como horarios para PAC o dudas generales sobre el sistema.

**1. Objetivo** Crear soluciones basadas en IA para optimizar rutas vehiculares, reducir la congestión del tráfico y mejorar la movilidad urbana.

**2. Métodos**

- Utilización de algoritmos de IA, datos de tráfico en tiempo real, características de la red vial y patrones de viaje individuales para formular estrategias de enrutamiento inteligente.
- Uso de WebScraping para obtener datos, seguido de limpieza y procesamiento con técnicas de NLP para entrenar un modelo que entienda consultas en lenguaje natural y recomiende rutas óptimas.
- Evaluación de la precisión de las respuestas mediante pruebas con usuarios y métricas de rendimiento.

**3. Resultados Esperados**

- Implementación de simulaciones y estudios de caso para evaluar la efectividad de las soluciones propuestas en la reducción del tiempo de viaje, consumo de combustible y emisiones.

- Desarrollo de un chatbot basado en RAG capaz de recomendar rutas óptimas y responder preguntas relacionadas con el sistema de transporte del metro de Medellín en tiempo real.

**4. Alcance**

- Evaluación de la escalabilidad y viabilidad económica de las soluciones de gestión del tráfico basadas en IA.
- Centrado en mejorar la comunicación entre el Metro de Medellín y los usuarios, facilitando la planificación de trayectos y la reducción de tiempos de viaje.

**5. Implementación Técnica** El proyecto se desarrolló con tres componentes principales:

1. **API de Rutas con FastAPI:** Se construye una API con FastAPI que interactúa con la API de Google Routes para calcular rutas entre dos direcciones en Medellín. Primero, carga la clave de la API desde un archivo de entorno utilizando *.env*. Luego, establece un modelo de datos con Pydantic para estructurar las solicitudes de origen y destino. La API cuenta con un único endpoint (/computeRoutes), que recibe la dirección de inicio y destino, junto con el modo de viaje (por defecto "TRANSIT"). Al recibir la solicitud, el sistema la envía a la API de Google Routes y procesa la respuesta para extraer los pasos de la ruta, incluyendo instrucciones, distancia, duración y modo de transporte.Para mejorar la presentación de la información, el código incluye dos funciones clave: extract_route_info, que analiza la respuesta y extrae los pasos de navegación, y format_instructions, que da formato a las instrucciones en texto estructurado para una mejor comprensión del usuario. Si la API de Google no encuentra una ruta, se devuelve un mensaje indicando la falta de resultados. Finalmente, el código ejecuta la aplicación FastAPI con uvicorn, permitiendo que la API esté disponible en un servidor local para recibir solicitudes.

2. **Interfaz en Streamlit:** se implementa una interfaz web interactiva utilizando Streamlit para que los usuarios ingresen un origen y un destino dentro de Medellín y obtengan una recomendación de ruta óptima. La aplicación envía estos datos a una API desarrollada en FastAPI, que consulta Google Routes para obtener instrucciones de viaje. Una vez que la API devuelve la ruta recomendada, la aplicación genera un prompt para un asistente virtual basado en Google Gemini, el cual traduce las instrucciones técnicas en un lenguaje más amigable y fácil de entender para el usuario. Además, la aplicación limpia el texto generado eliminando etiquetas específicas como `<think>` y guarda la respuesta humanizada en un archivo de texto. Para hacer la experiencia más cercana y útil, la IA genera una respuesta con consejos adicionales, como advertencias sobre el clima o recomendaciones para el viaje. La interfaz muestra la respuesta de manera clara y estructurada para facilitar la navegación del usuario. Si se detectan errores en la solicitud a la API, la aplicación maneja excepciones y muestra mensajes de error adecuados. También se configura un directorio para almacenar los informes generados con las respuestas del asistente

virtual. Esto permite a los usuarios acceder posteriormente a las instrucciones de su ruta sin necesidad de hacer una nueva consulta.

3. **Análisis de Comentarios en Redes Sociales:** Este código implementa un scraper de Twitter utilizando la biblioteca Tweepy para buscar comentarios recientes relacionados con cómo llegar al Metro de Medellín. Para ello, se configura una consulta (query) que busca tweets que contengan frases como "cómo llegar desde" junto con palabras clave como "metro de Medellín", excluyendo los retweets para evitar duplicados. La búsqueda se realiza a través de la API de Twitter utilizando un Bearer Token, que autentica la solicitud. Luego, el código obtiene hasta 100 tweets recientes que coincidan con la consulta, extrayendo información relevante como la fecha de publicación, el nombre de usuario, el contenido del tweet, la cantidad de likes y retweets.

   Una vez recopilada la información, el código la almacena en un archivo CSV llamado comentarios_metro_medellin.csv, organizando los datos en columnas para facilitar su análisis. Durante el proceso, también imprime cada tweet recuperado en la consola, permitiendo visualizar los resultados en tiempo real. Para mejorar la gestión de errores, el código captura excepciones de Tweepy y muestra un mensaje en caso de que ocurra algún problema con la solicitud. Además, se limita la cantidad de tweets a 50 en la ejecución para evitar sobrepasar las restricciones de la API de Twitter, asegurando un uso eficiente de los recursos disponibles

   A continuación, se presenta una captura de la interfaz desarrollada:

**6. Conclusión** Este estudio presenta un sistema de recomendación de rutas basado en IA que mejora la movilidad urbana en Medellín. La combinación de FastAPI, Google API y Streamlit permite una navegación en tiempo real basada en datos. Como mejoras futuras, se planea integrar opciones de transporte multimodal y expandir las capacidades de IA para ofrecer recomendaciones más personalizadas. También se prevé el uso de modelos más avanzados de procesamiento de lenguaje natural para una interacción más precisa con los usuarios.

## 7. Referencias

**[1]** Arenas Montoya, M. A. (2024). *Prototipo rutas de transporte público usando inteligencia artificial.*

**[2]** Ruano Morales, R. A. (2019). *Transmovilgt: aplicación móvil para determinar la ruta a utilizar por el usuario de transporte público de Guatemala por medio de servicios de Google Maps* (Doctoral dissertation, Universidad de San Carlos de Guatemala).

**[3]** Juan Sampedro, S. (2015). *GeoTurismo, aplicación móvil para rutas turísticas*.