Mapper: Optimizing Everyday Travel with AI and Location-Based Task Management

Department of Mathematics and Computer Science
Nashville, TN 37208

Ashim Chand, Sandesh Lamsal, Sudhan Pandey

Abstract

Mapper is an AI-powered travel assistant that combines to-do lists with map services to help users complete tasks efficiently while traveling. It uses natural language processing (NLP) to understand user tasks and identify nearby locations where those tasks can be completed. By suggesting stops along a user's current route, Mapper minimizes extra travel time and boosts daily productivity. It is ideal for commuters, busy professionals, and gig workers who want to make the most of their travel. Mapper reduces unnecessary trips, saves fuel, and supports eco-friendly habits, making travel smarter, faster, and more purposeful through real-time, intelligent task recommendations.

Objectives

- Build AI that reads tasks and finds matching locations along the user's route.
- Suggest relevant stops with minimal detours to improve travel efficiency and productivity.
- Use machine learning to refine recommendations based on user preferences and past behavior.
- Seamlessly integrate task locations into routes for a smoother, smarter travel experience.

System Architechure & Techonologies

- **Frontend & Interface:** Built using **React.js** to deliver a clean, responsive interface for task management, route visualization, and real-time interaction with smart suggestions.
- **Backend & AI Layer:** Developed with **Node.js** and integrated **OpenAI** and **nltk** for natural language processing, allowing the system to interpret and categorize user tasks intelligently.
- **Mapping & Route Engine:** Integrated with the **Google Maps API** to enable real-time route generation, nearby location search, and optimal stop suggestions with minimal detours.
- **Database & User Storage: MongoDB** handles secure storage of user tasks, route history, and behavior data to support personalized, adaptive recommendations over time.
- **Recommendation & Learning System:** A custom-built engine uses machine learning to refine future task-location matches by learning from user actions, preferences, and travel patterns.

Fig 1: Federated Learning Architecture

Design Model

Fig 1: Design Wireframe for the application

Application Prototype

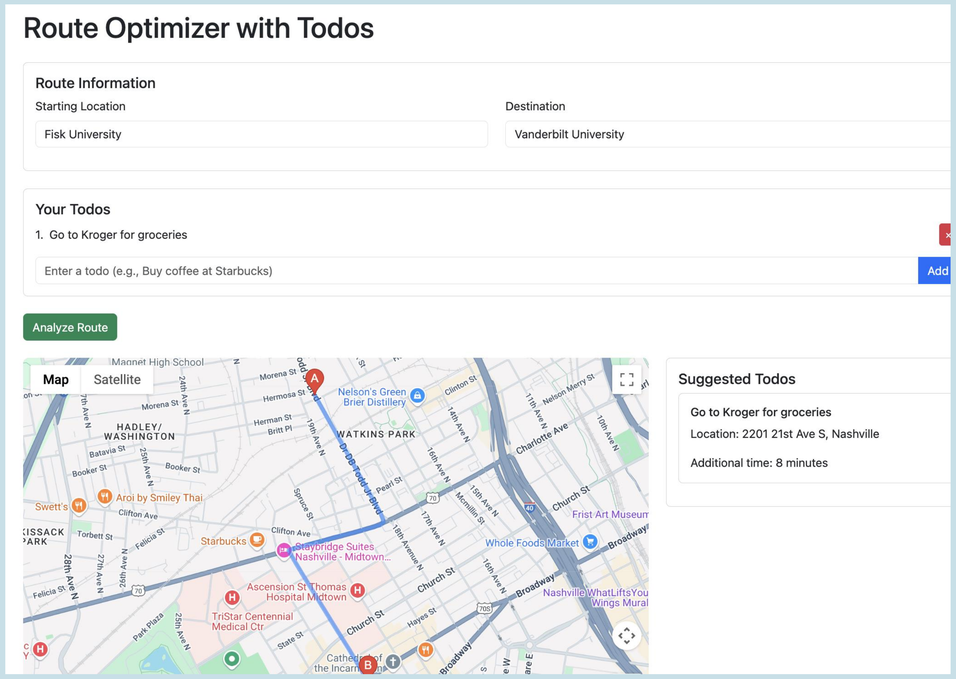

Fig 2: Prototype for the web applicatoin

Implementation & Challenges

We followed an Agile development approach, working in iterative sprints to design, build, and test features efficiently. Major challenges included interpreting vague task descriptions, accurately matching tasks with nearby locations, and ensuring minimal travel detours. We addressed these using NLP (OpenAI, nltk), Google Maps API, and custom route optimization algorithms.

- **Real-Time Updates:** Ensuring recommendations adapt to the user's changing location required frequent data refresh and efficient handling of API rate limits.
- **Task Prioritization:** Determining which tasks to suggest first based on urgency and proximity involved designing a dynamic, context-aware prioritization system.
- **UI/UX Consistency:** Designing an intuitive, user-friendly interface that effectively combined task lists and maps was a key challenge solved through iterative feedback and prototyping.
- **System Integration:** Seamlessly connecting frontend, backend, AI models, and external APIs demanded careful module coordination and extensive testing to ensure reliability and responsiveness.

Conclusion

Mapper effectively combines AI and map services to enhance productivity during travel by suggesting smart, task-based stops. It reduces wasted time, lowers fuel consumption, and supports sustainable habits. By learning from user behavior, Mapper continues to improve, offering a practical solution for busy individuals seeking efficiency in their daily routines.

Future Work

- Integrating public transit options
- Real time traffic predictions
- Voice based task input
- Expanding AI capabilities to suggest eco-friendly routes, collaborative errands
- Deeper calendar synchronizations