General description:

MadTransit Live is a web application designed to help Madison residents and visitors make smarter transportation choices. Finding parking downtown can be frustrating, and many people aren't sure when it's better to take the bus instead. This project combines real-time data and predictive modeling to provide users with up-to-date transit and parking information in one convenient place.

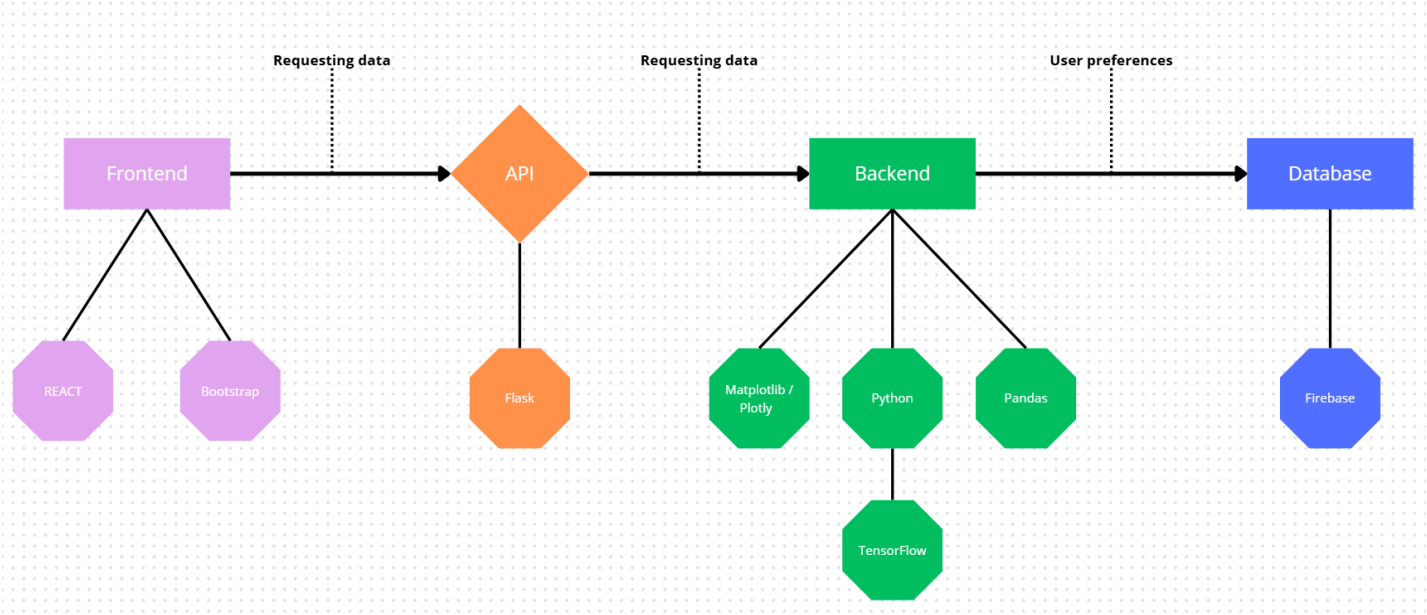
Parking garage availability

Recommending whether you should take car / walk / bus

potential additions: using weather data to recommend if you should walk

Core Features:

- **Live Map:** showing current parking availability
- **Transit Integration:** Shows Metro Transit bus routes, stops, and Park & Ride locations on the same map
- **Route Planner:** Suggests whether it's more efficient to drive or take the bus based on live conditions
- **Parking availability:** prediction of number of parked cars
- **Database:** summary of each user's preferences, allowing them to send notifications about bus schedules or recently visited parking lots.
- Functional Website (including responsive page)
- API returning routes and results of machine learn prediction



System Architecture

MadTransit Live follows a client-server architecture with a modular, scalable design.

Technology Stack

Frontend: React + Bootstrap for responsive UI.

Backend: Flask (Python) REST API for business logic and machine learning integration.

Database: Firebase for consumer preferences.

Data Streaming & APIs: Integration with Madison Metro Transit (GTFS-RT) and Madison Parking APIs.

Machine Learning Model: Python (scikit-learn or TensorFlow) for predicting parking availability.

Data Sources

Madison Metro Transit API (GTFS-RT) - Provides real-time bus route data, stop locations, and live vehicle tracking.

City of Madison Parking API - Supplies live data about garage occupancy and available spaces.

OpenWeatherMap API - Stores preferences, last used parking lots, and notification settings.

Not essential
Internal User Database - Stores preferences, last used parking lots, and notification settings.

Backend API Design

The Flask API exposes RESTful endpoints that return JSON data for the frontend.

/parking - GET - Returns live and predicted parking data.

/bus-routes - GET - Returns nearby bus routes and stops.

/weather - POST - Returns actual weather in a time and location (predict the weather).

/recommendation - POST - Suggests transport mode (drive/bus/walk) based on a given location (start and end).

/**predict-park -** POST - What will be the occupation of some parking in a given time.

Database Design

The database is not exactly essential, we could use local storage (cookies) istead.

User:
user_id - INT - user unique number
name, - STRING (textchar) - name of the user
email, - STRING (textchar) - email of login
password, - STRING (textchar) - password
hash, - INT - hash id
preferred_mode, - INT - Id of a type of transportation
last_routes - Array - list of last routes requests (can be changed for another approach if easier)

Modes:
mode_id - INT - mode unique number
name, - STRING (textchar) - name of the user

Hash:
hash_id - INT - hash unique number
code, - INT - random number used to encrypt the password

Design

Frontend project