Constrained Re-Planning in Spatial Crowdsourcing, PIRM II

Team 51
Steven Sheets (Backend Engineer, Test Engineer)
Logan Anderson (Frontend Engineer, Test Engineer)
Nicholas Heger (Frontend Engineer, Progress Manager)
Jared Weiland (Backend Engineer)
Jame Volpe (Frontend/Backend Engineer)

Advisor: Goce Trajcevski

sdmay21-51@iastate.edu

Project Recap - I

Project Goal:

Create a spatial crowdsourcing algorithm that runs on a mobile and web application that match workers with tasks from consumers.

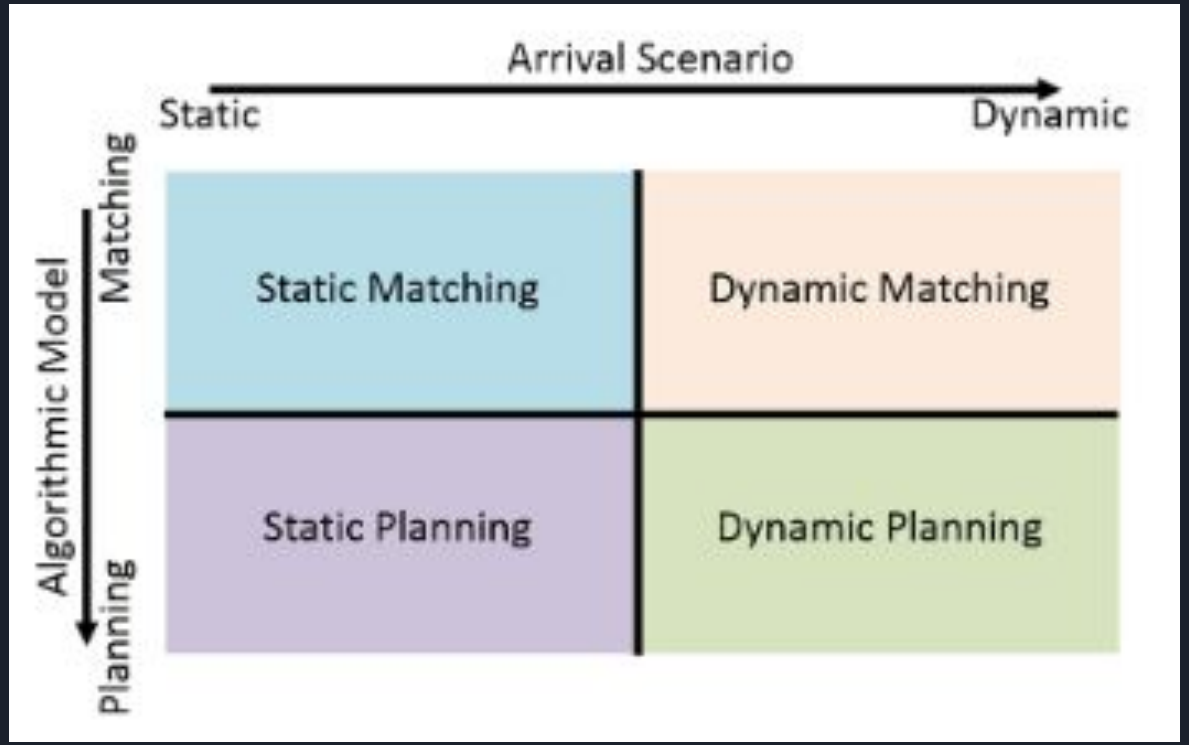
Project Recap - II

4 types of spatial algorithms

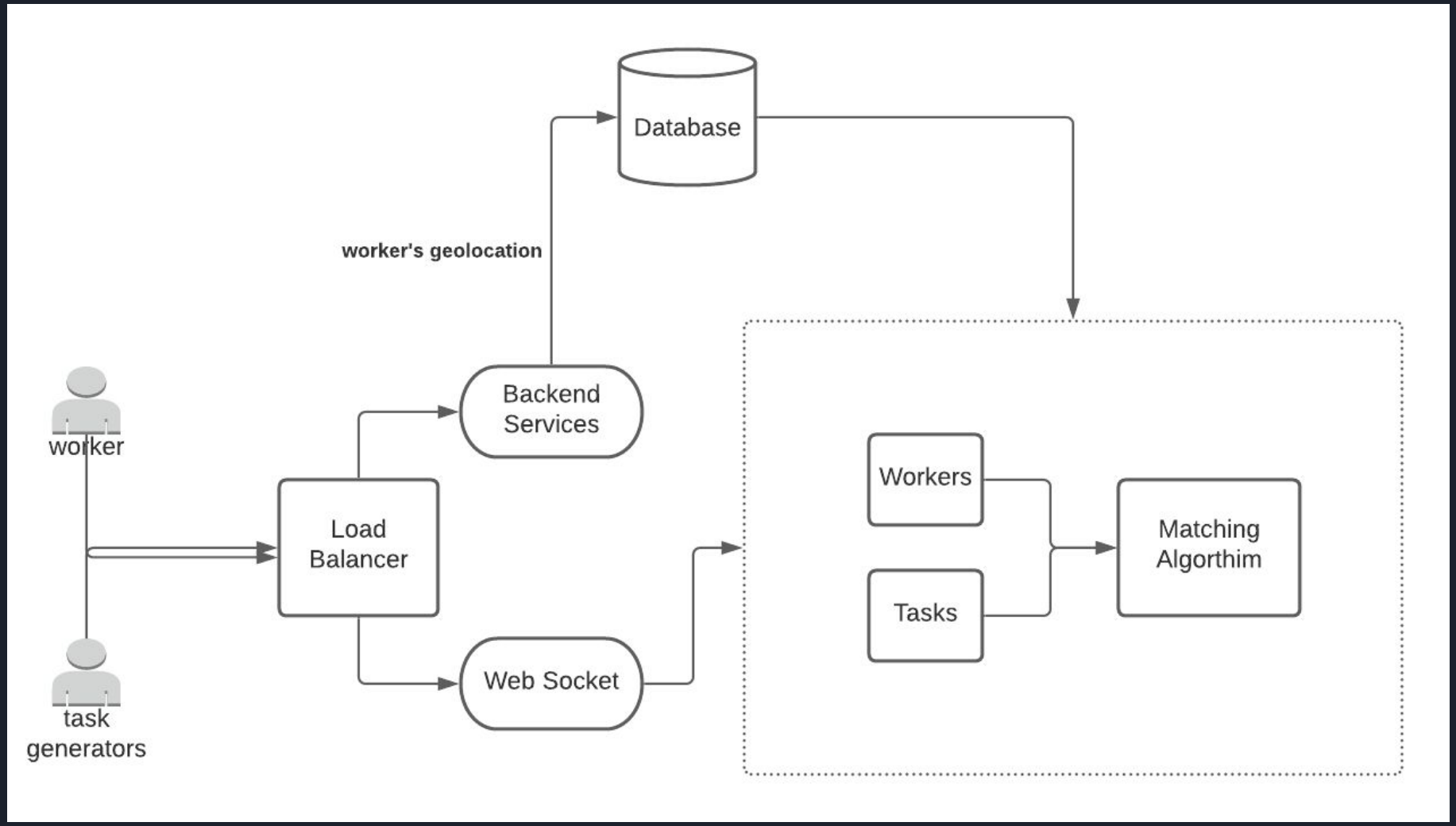
- Static Matching
- Static Planning
- Dynamic Matching
- Dynamic Planning

Current Implementation plan

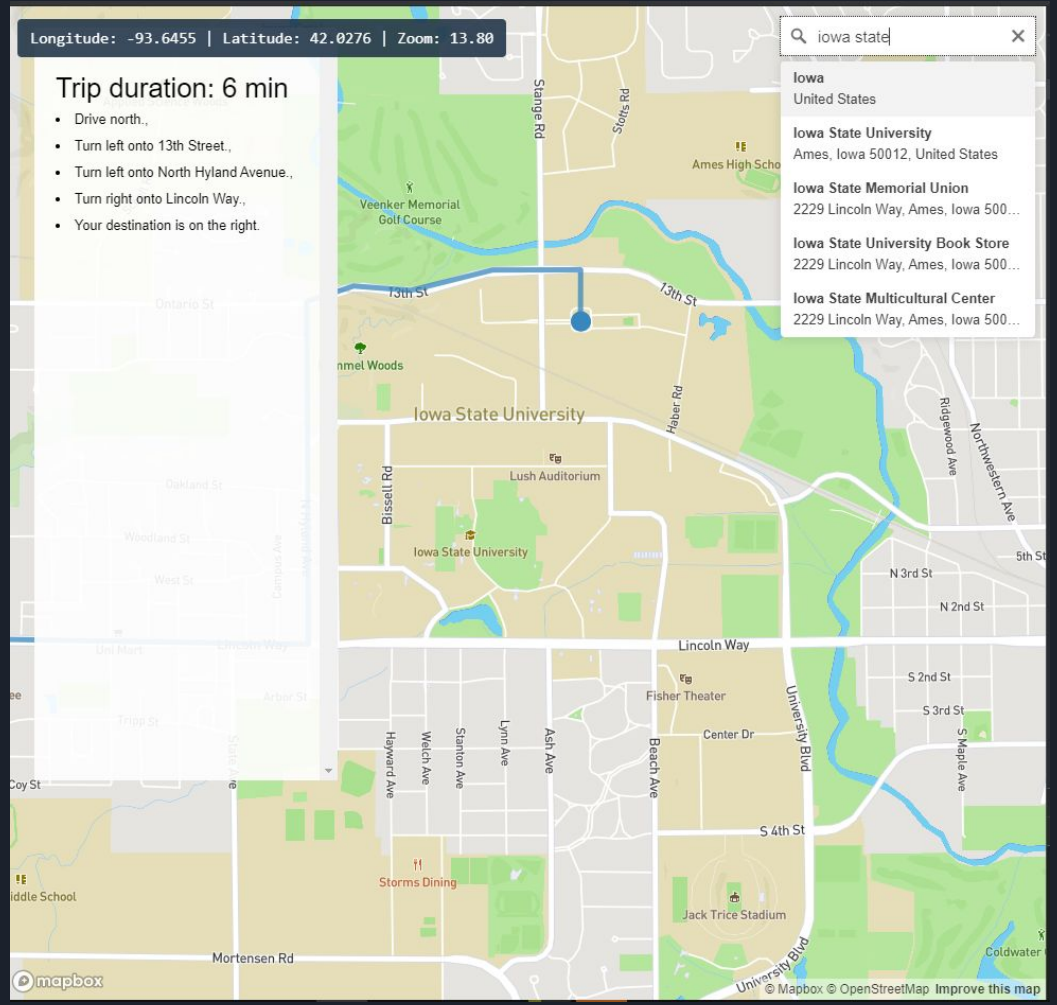
- Initial: Static Matching
- Final: Dynamic Planning

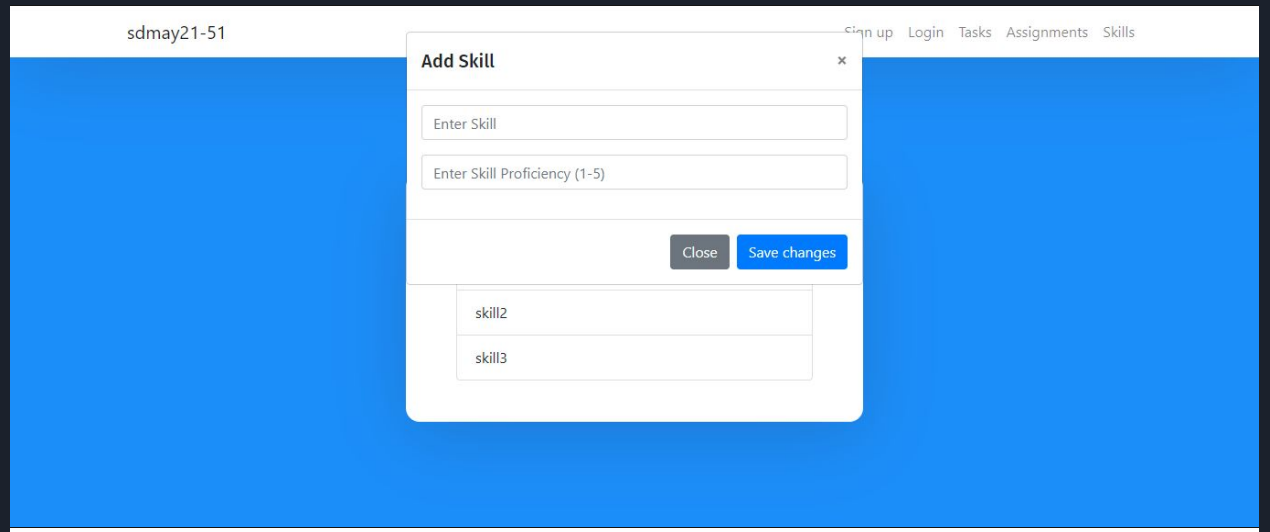


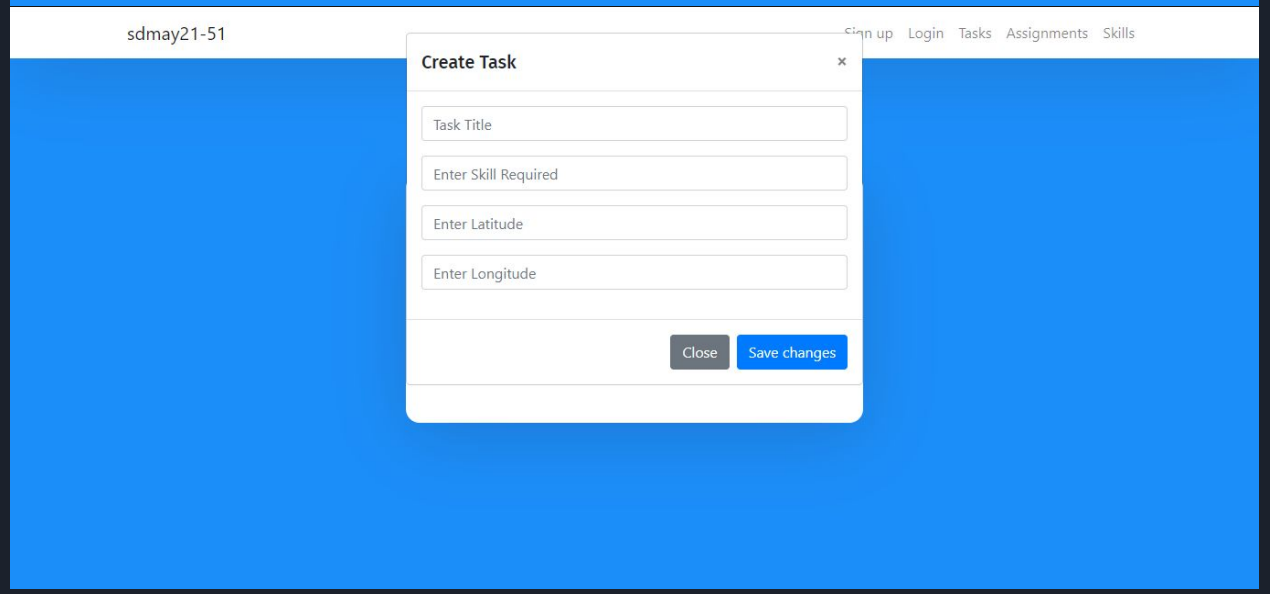
System Design - Component Diagram



Current Screenshots of Desktop App







Engineering Standards

- IEEE/ISO/IEC 29119-2-2013 - ISO/IEC/IEEE International Standard - Software and systems engineering —Software testing —Part 2:Test processes

- IEEE/ISO/IEC 29119-3-2013 - ISO/IEC/IEEE International Standard - Software and systems engineering — Software testing —Part 3: Test documentation

- 29119-4-2015 - ISO/IEC/IEEE International Standard - Software and system Engineering -- Software testing --Part 4: Test techniques

Engineering Constraints

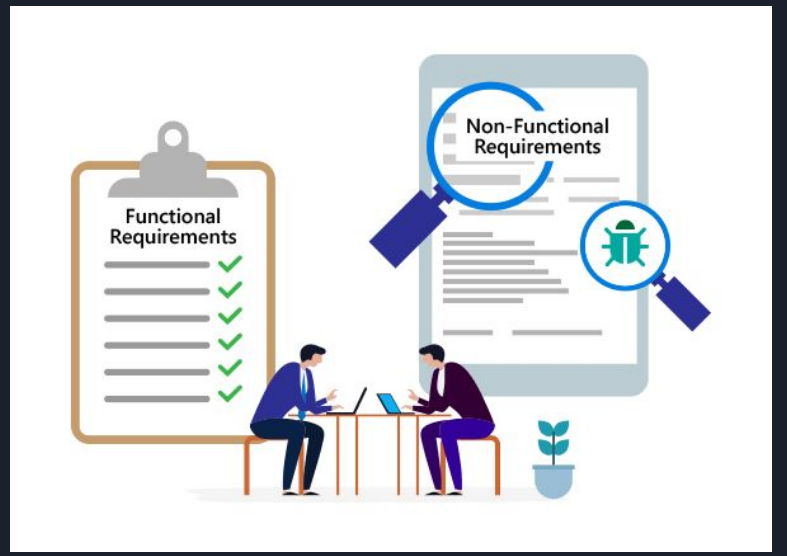
- Must run as a mobile and desktop app
- Server needs to be able to handle algorithm processing
- Application requires internet connection
- Free Mapbox API
  - 50,000 monthly map loads
  - 100,000 monthly direction requests
  - 100,000 monthly geocoding requests
- Project must work without a budget
- Project must be completed within the semester

Functional Requirements

- Allow employers and workers to be able to create accounts (stored in DB)
- Take input of workers: skills, location, and reputation
- Take input of tasks: location(s) and skills required
- Optimize a schedule based on task and worker input
- Alert workers of incoming tasks
- Re-optimize schedule in the event of new constraints
- User interface for visualization of work schedule

Non-Functional Requirements

- Reliability - few bugs or issues that impede user experience
- Performance - algorithm is polynomial time and app is optimized for web/mobile
- Scalability - able to be used by a large number of users simultaneously
- Maintainability - readable code with documentation
- Usability - intuitive/easy to use
- Modularity - be able to switch between components (like algorithm used)

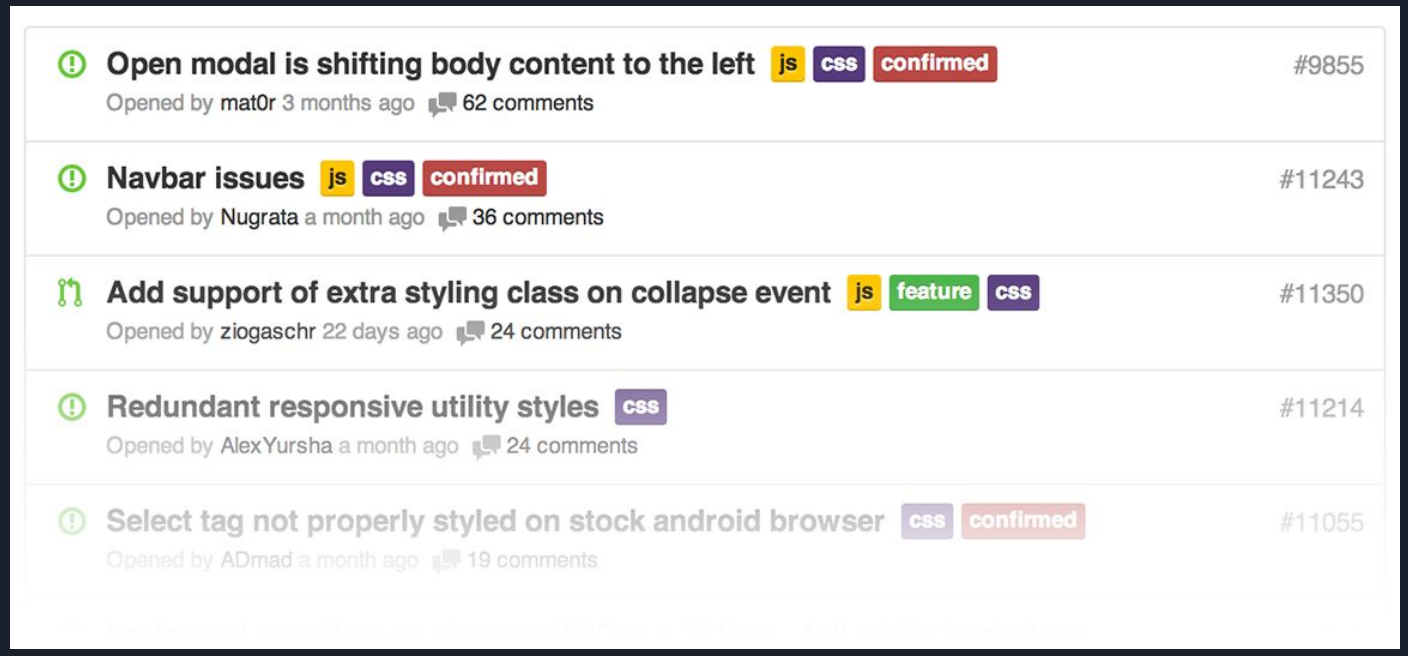


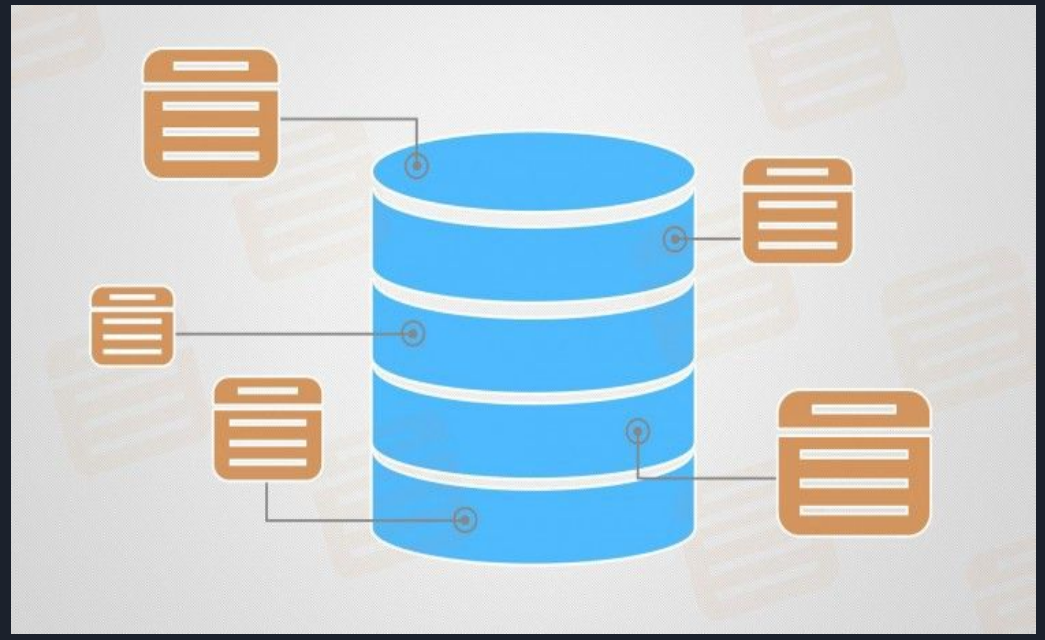
Technical Challenges

- Implementing spatial crowdsourcing algorithm
- Establishing a server
- Frontend-backend communication/familiarizing ourselves with Apollo Graphql
- Familiarization with Mapbox API
- Working without in-person team interaction for much of the project
- Makes sure there is not memory bloat in the client

Immediate Next Steps

- Expand frontend-backend communication
- Work on implementing alternate spatial crowdsourcing algorithms
- Run our backend on a server and set up CI/CD.





Thank you!

Any questions, concerns, or comments are greatly appreciated!
Looking forward to feedback!