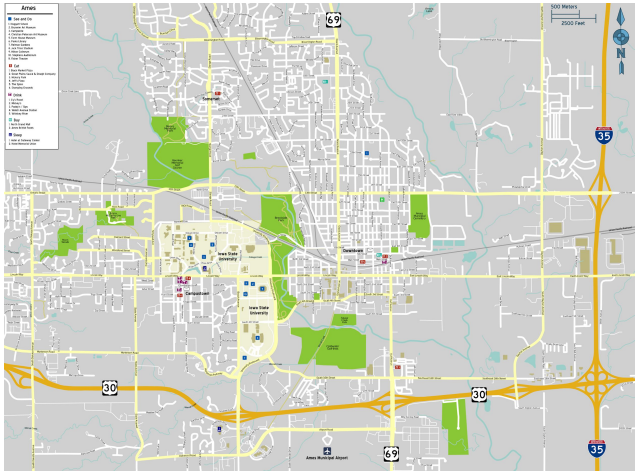
Real-Time Re-Recommendation System for POI Visits

By Team 22: James Eehinus, Dheepak Nalluri, Andrew Peters, John Smolinski, and Luke Sun
Client/Advisor: Goce Trajcevski
Website: https://sdmay20-22.sd.ece.iastate.edu/

High level overview

- App for helping users find nearby POIs
- Routes the user between all selected POIs
- Web & Android version

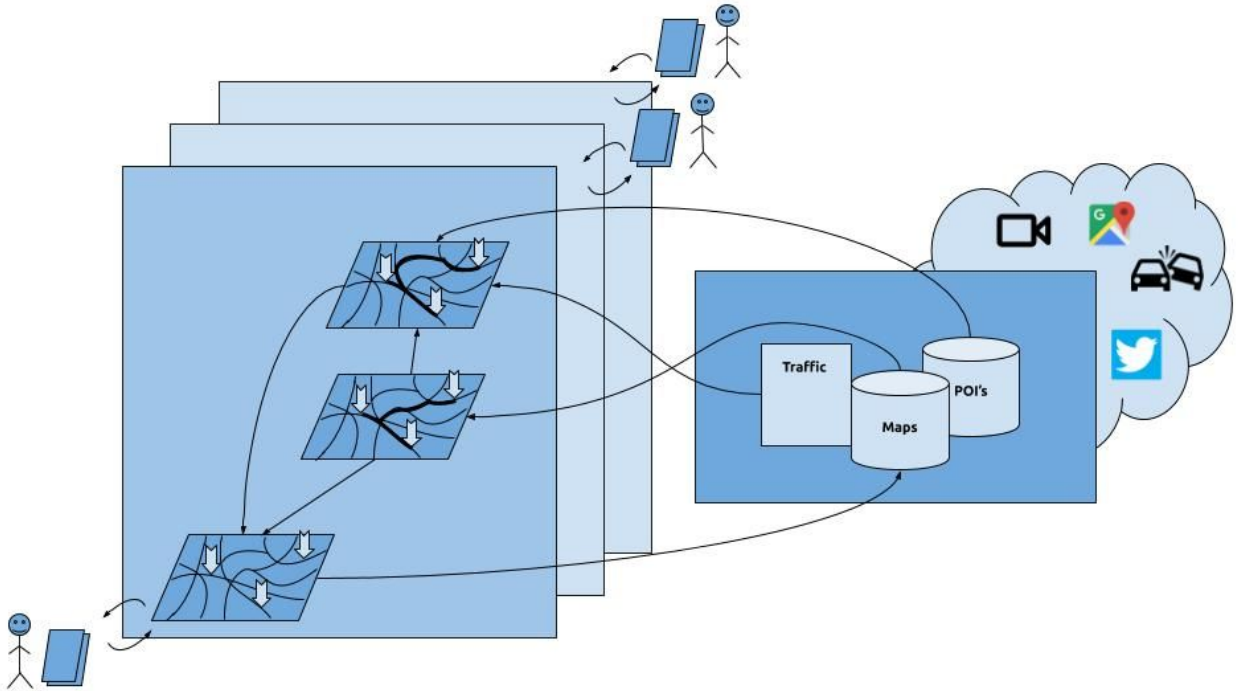


Problem Statement & Needs

- If travelers visit new locations they will be unfamiliar with the surrounding area
- Might not be able find POI’s (restaurants, hotels, bars)

- Users will not use the application if it is too inconvenient
- Application must be able to find POI’s in a timely manner

Concept Sketch



Functional & Non-Functional Requirements

Functional Requirements

- Server generates a route for a user
- User is able to list POIs

Non-Functional Requirements

- Web and mobile application
- Scalability

Technical Limitations & Constraints

Assumptions

- We have access to the APIs and databases
- Security is not a priority
- Product will only be used in the US

Limitations

- Maximum user service
- Constrained to other route databases
- One semester of development time

Potential Risks & Mitigations

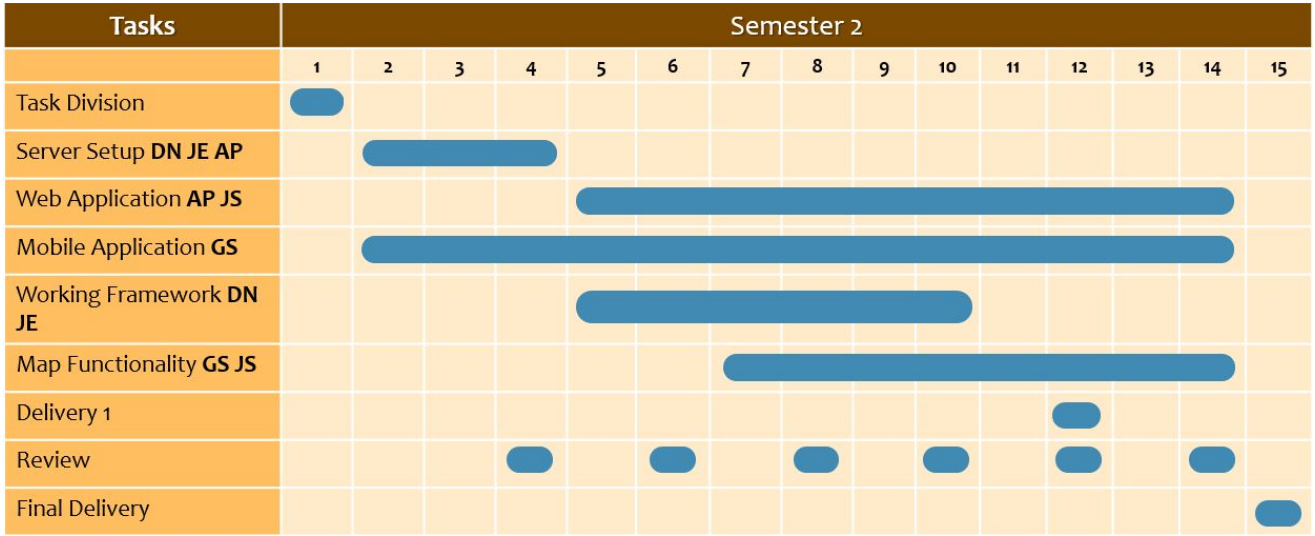
- Risks
  - Google API require payments after certain amount of use
  - Lack of Machine Learning knowledge
  - Unknown API understanding and usage
- Mitigations
  - Testing limited to under Google’s paywall.
  - Machine learning given set deadline to learn if not able to implement dropped.
  - Split needed API usage into what would satisfy requirements, cut excess features.

Resource/Cost Estimate

- ISU Server
- Android Phone
- Google account for API

Google APIs

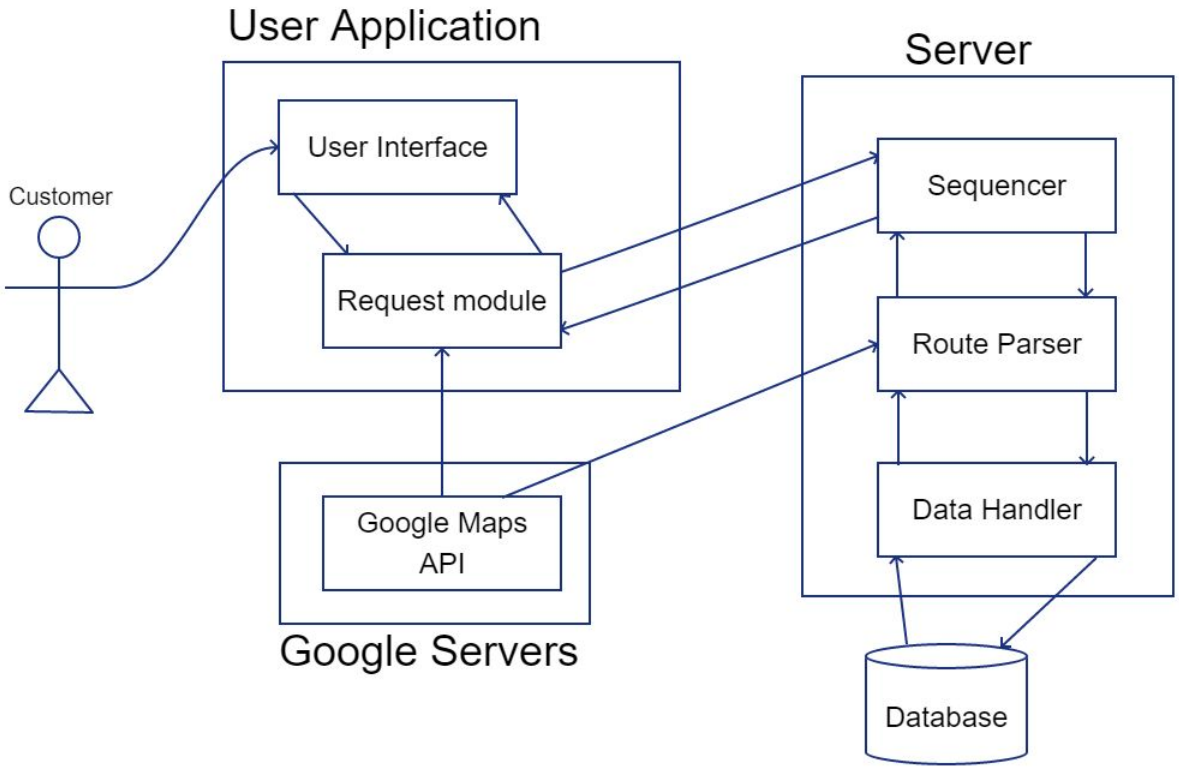
Project Timeline



Key

DN - Dheepak Nalluri
JE - James Eehinus
AP - Andrew Peters
GS - Luke Sun
JS - John Smolinski

Functional Decomposition



Detailed Design

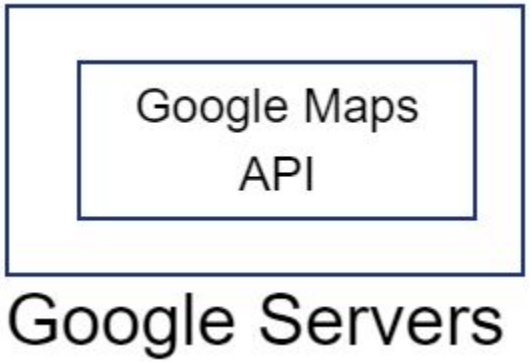
- Android Application
- Web Application
- Built in parallel to mirror functionality

Hardware & Software

- Android Phone
- Web browsers
- Server
- Android Studio
- WebStorm
- Visual Studio Code

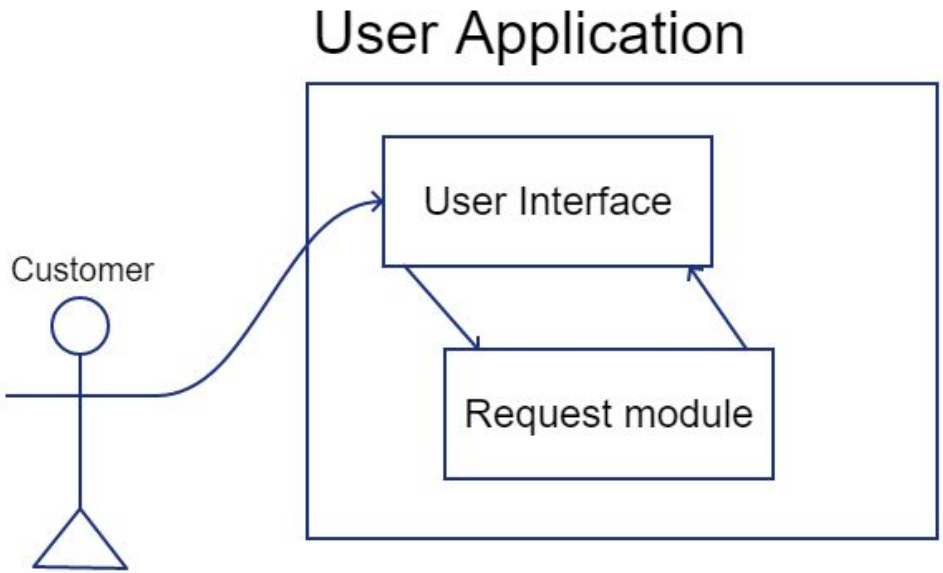
Google API

- Directions API
- Places API
- Integration with both web and mobile



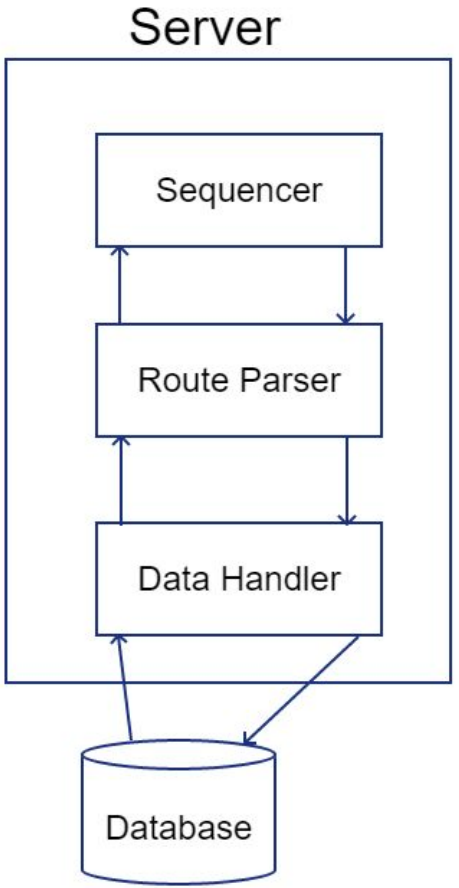
User side

- User input
- Packaging and server communication
- User information request



Server side

- Data request queue sequencer
  - Time sensitive
- Request processing
- User information database



Test Plan

- Client Acceptance
- Mocha with Meteor
  - Unit testing on server
  - Allows for client side testing

Demo

Standards And Practices

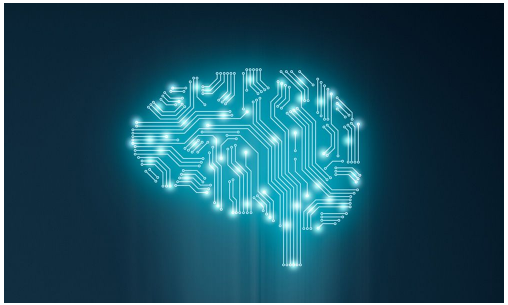
- IEEE 12207-2017
  - Software Life Cycle
- IEEE 14764-2006
  - Software Maintenance
- IEEE 29119-5-2016
  - KeyWord-Driven Testing

Contribution breakdown

- John Smolinski
  - Web Application Development
- James Eenhuis
  - Database Management
- Dheepak Nalluri
  - Server Functionality
- Andrew Peters
  - Web Application Development
- Luke Sun
  - Mobile Application Development

Future Prospects

- Machine learning
  - Routing
  - Recommendation of POI’s to users
- Sequencer for requests
- Planner Implementation



Thank you for listening!