Project Proposal

Hangry Mobile Food Truck Location Service

San Jose State University
CS 161 Software Project

Team Members

John Humlick
Amir Radman
Aaron Escoto
Daniel Tam
Natalia Zarubin

TABLE OF CONTENTS

Description of the Product:

Have you ever been in an unfamiliar location when hunger strikes? If so Hangry is for you. Hangry is your guide to food trucks and their locations. Don't get angry and hungry when you can find and map the location of food trucks and place your order ahead of time. Our goal for this application is to bring hungry users to mobile food truck owners.

Need for the Product:

Our product will eliminate the hassle of finding mobile food trucks around the city. We also plan to provide additional features for the user for an easy and quick process of finding food trucks.

Potential Audience:

Our potential audience consists of two groups -- hungry users who are looking for food trucks nearby and food truck owners who want to provide their services to a larger audience.

Discussion of Competing Products:

A few food truck applications exist in Android play store, such as "Foodtruck Nearby" and "Food Trucks", however, the mobile applications either lack in features or area they provide their food truck service. The user interface also leave something to be desired and is something our application can improve on.

High-level Technical Design:

We will utilize the Google Location Services to locate the food truck drivers' locations and then store the locations in real time via Amazon Web Services (AWS). Additionally, we will implement a messaging system over AWS that allows users to place an order (message) and the food truck drivers to receive that order (message). We may also implement a rating system where users allow access to their Google account profiles and rate the food trucks, with the ratings being stored on DynamoDB hosted by AWS.

Feature List:

Core Features:

- User and Food truck driver UI
- Geo-location
    - Food truck owners push their location to our server (AWS)
    - Tracking/proximity for users
- Profiles (Google account based)
- Rating System (Requires user login)
    - User comments / owner rebuttals

Product Backlog:

- Messaging system for orders (AWS)
- Truck is moving notification
- Proximity-based notifications
- Android pay
- Crowdsourcing
    - Users can submit food truck sighting and menu specials
- User submitted pictures of food / trucks

Resource Requirements:

Building this application requires various resources that we need to incorporate to ensure it delivers its promised features with optimized performance. One of the challenges we'll be facing would be using AWS and DynamoDB considering we are fairly new to these technologies. In addition, identifying valid food trucks could also be a challenge for us.

Potential Approaches:

There were two other approaches we could have gone with. We could have allowed the food truck drivers to enter in their locations to be stored statically, or we could have gathered their information and created profiles for them. The problem with these approaches comes from the static data. Food trucks are not stationary objects, they are

moving vehicles. We needed to find a solution that could collect locations dynamically in real-time.
Additionally, we could have utilized a single MySQL server on the internet to store locations, profiles, etc. We wanted to utilize a streamlined service that has a high availability, different database options to choose from, and is free for students. This is why we chose to go with AWS.

Assessment of Risks

Our project assumes a certain level of risk of new technologies that none of our team have ever used:

- Android Pay
- DynamoDB
- Google Location (GPS)
- AWS

Next Steps:

Since we are planning to use new technologies that we've never seen or used before, we will first have to get familiar with those technologies -- mainly Android Studio, AWS, and NoSQL database. We will achieve this by researching about the layer of the stack that each person is responsible for, either by reading the documentations or going through online tutorials. After each person is familiarized with their layer, we can go into a more detailed design of what we think will be feasible to create. The next step after designing would be to develop our actual product; we will first start with our core features and dig into our backlog for additional/replacement features if needed. Unit testing will be done incrementally with every feature we add. Integration testing will be done towards the end of our development phase and if all the functionalities work, we will go into deployment.