**DUDETRUCK**

| | |
|---|---|
| Aaron Hartigan | 219247094 |
| Matthew Nix | 217474596 |
| Thomas Hoang | 219484409 |
| Alexandru Seremet | 219247393 |
| Christel Garcia | 219563358 |
| Lincoln Gallegos | 215590766 |
| John Veit | 218751027 |
| Vahak Gilian | 215258590 |

## Process Description

**Software Toolset:**

| | |
|---|---|
| Programming Languages: | JavaScript (ES6), JSX, CSS |
| Version Control: | git with hosting on GitHub |
| Bug Tracking: | Trello |
| Other Tools: | Sequelize, GraphQL, React |
| Other Components: | React Starter Kit |

**Explanations:**

JavaScript (ES6): Choosing to work in JavaScript is a side-effect of choosing to work with React. Even though most of the group knows Java (not JavaScript), JavaScript shares enough C-based syntax to be able to quickly learn it. We are choosing to utilize ES6 for some useful features such as module import/export (to simplify using libraries) and arrow functions (to simplify some verbose syntax).

JSX: Also A side-effect of choosing to use React. It is essentially HTML with minor changes.

CSS: The easiest language to style React Components. (No need for SASS or LESS)

Version Control: We using git because it is industry standard.

Bug Tracking: Trello is an extremely simple way to track bugs. While it does not have extensive features, it has the essential basics: opening new bug tickets, commenting on bugs, and tracking progress on bugs.

Sequelize: A JavaScript ORM for interacting with a SQL database without having to use raw SQL queries.

GraphQL: A query language for an API. We will use a JavaScript implementation of GraphQL to keep all our code in JavaScript. Furthermore, it integrates nicely with Sequelize. GraphQL also makes it easier for the frontend engineers to request data by bundling multiple endpoints into a single request, making it highly efficient for a mobile app (where many HTTP requests can bog down performance).

React: A highly efficient, lightweight JavaScript framework. Although React has a somewhat steep learning curve, it's highly modular design will save us time overall by having easy-to-debug components. We had the option to use React Native instead of React. However, our team is more familiar with React, and we believe it will be more efficient to work on a mobile-web app rather than a native app.

React Starter Kit: A boilerplate for React development. It sets up tools such as Babel and Webpack that allows us to automatically transpile our ES6 features and build our code. It also sets up Hot Module Reloading, which allows us to edit code and see the changes in real time.  It also sets up code for things such as an Express server and automated testing, without implementing any business logic.  This will allow us to immediately begin coding our features.  And most importantly, it uses the MIT License, which allows us to use, modify, distribute, and sell the code.

**Group Dynamics:**

The project manager for DudeTruck is Matthew Nix. We have selected to separate the roles out at this juncture. The UI designers are Matthew Nix, and Vahak Gillian. They have a vision of how the end product will look. The frontend development team contains Aaron Hartigan, Alexandru Seremet, and Lincoln Gallegos. They volunteered to do this part of the project. The backend team contains Thomas Hoang and Christel Garcia. They have experience with background servers and their communication. The tester will be John Veit. He likes looking for ways to improve a product through the creation process. As the project progresses these roles may change as one portion of the project may require more or less man power. We shall handle disagreements as adults by talking them out and coming to a mutual agreement on the subject. We believe this structure shall allow our group the best chance at success for now and in the future.

**Schedule / Timeline:**

To maintain consistency between our sub-groups, we’ll meet up in person for 2 hours every week, discussing the progress each group has made and their own individual limitations or problems that should be addressed to other groups for re-discussion. We’ll have the front-end team work on the skeletal implementation of our product, with them communicating with the design and UI team, which should take a reasonable 3-4 weeks. Concurrently, the documentation team will occasionally check on the website and update the readme files as necessary, keeping track of changes initiated by the UI team. The back-end team will be implementing GraphQL API endpoints. Each individual endpoint should take a developer about one day. We will need about 5-10 endpoints total (food truck location, update settings, etc.). Thus, the beta release should be ready between October and November, with the testers conducting weekly checks for potential bugs.

**Risk Summary:**

The major risks of DudeTruck are acquiring locating food trucks and filtering user search results based on food restrictions/preferences. These are the most important risks, but we are also concerned with gathering food truck data and analyzing consumer analytics. The food truck data and consumer analytics are not as important, because they are not fundamental to our software specifications.

The greatest concern of our service is being able to acquire and implement food truck vendors location and information for our users. In the event that we are not able to implement actual food truck vendors schedules and locations then we will have to make up vendors.

Since we are most concerned with implementing vendor information and locations as well as being able to filter user results we may need to adjust group dynamics to meet these specifications. We will make this adjustment by having more members brainstorm ideas and methods on implementing these important implementations. Another mitigation to allow more time implementing the important features of vendor information and user preferences would be to cut the time spent on other features.