**Foodar 100**

**System Requirements Specification (SRS)**

1. Description
2. Purpose
3. Scope
4. System Overview
5. Use Cases

**Description**

Foodar is a food delivery service application. Foodies will order take-out from a local restaurant, then use Foodar to find a Driver to pick up and deliver it to them. Delivery fees will be calculated based on distance driven, plus a base fee. Users must register a card to autopay drivers the approved amount.

User Roles:

1. Foodie: will create a delivery mission to pick up their food from a restaurant.
2. Driver: can accept delivery missions, and manage income.

Primary stakeholders have decided that Foodar users should have the ability to register to keep track of used services.

**Purpose**

The purpose of this document is to provide a guide for developers and testers who are responsible for the development of the platform. It provides the information necessary to design, develop, and test the software.

**Scope**

This document serves as a description of the required improvements to the project. It contains a workflow description, functional and nonfunctional requirements, and appropriate diagrams to fully describe the system.

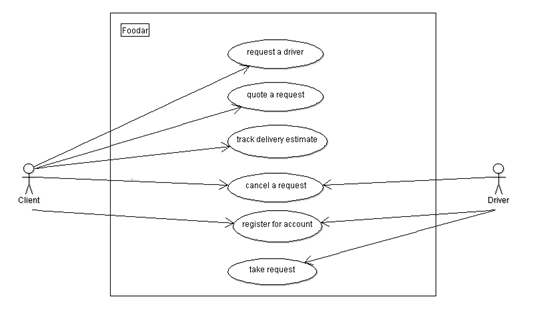
**System Overview**

This enterprise project consists of one layer. The front-end is the one layer and is the portal through which users will interact with the application. Primary interactivity is achieved using JavaScript executing in the browser. JavaScript is used to enhance a user's experience and provide additional functionality.

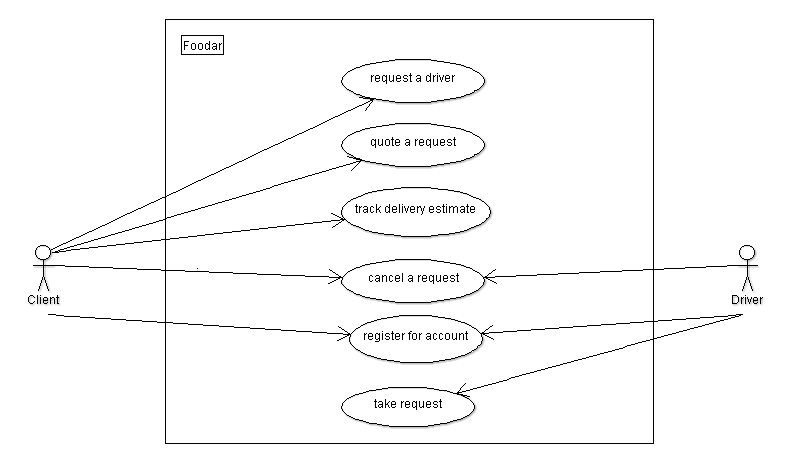
**Use Cases**

1. **All Users**
   1. **Request System**
      1. Create a request to find nearest driver
         1. algorithm should be clean
      2. Cancel a request
      3. Quote a request
         1. algorithm to generate price
            1. base price + request cost + driver fee + taxes
            2. need to generate tax table
      4. Track current time until receipt of food
   2. **Member System**
      1. Register a new member
      2. Sign-in a member
      3. Track requests

Use Case 0001 - Request System



Use Case 0002 - Member System



MVC or Model, View, Controller.

* one that is the visual part that users interact with - the view,
* one that represents the object being modeledthe model,
* and something that manages data between the different parts - the controller.

**Foodar 100 – Requirements (Dive Four)**

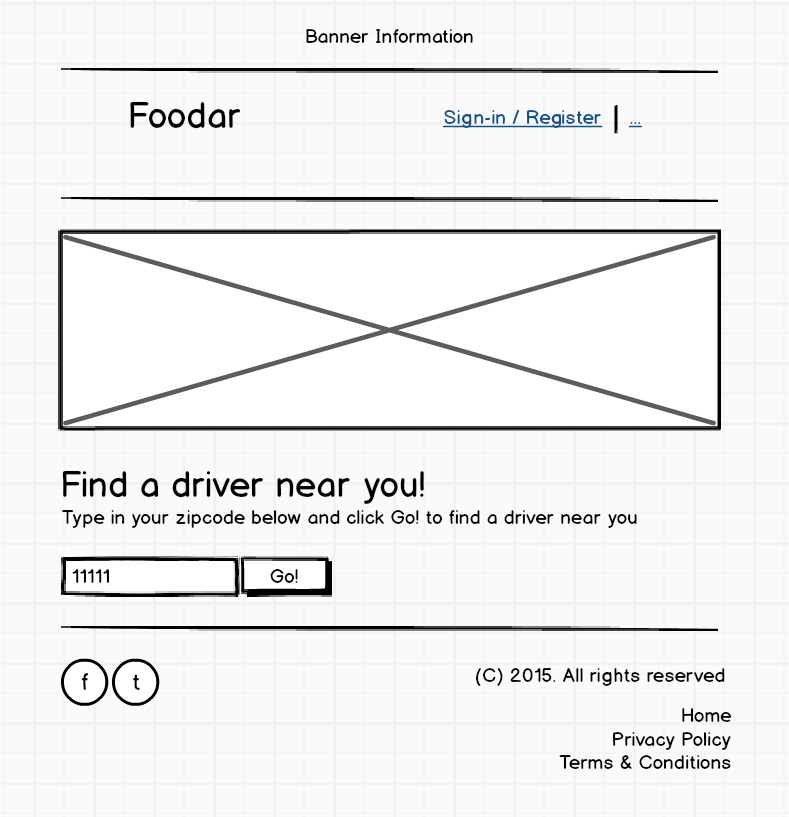
**Next step.**

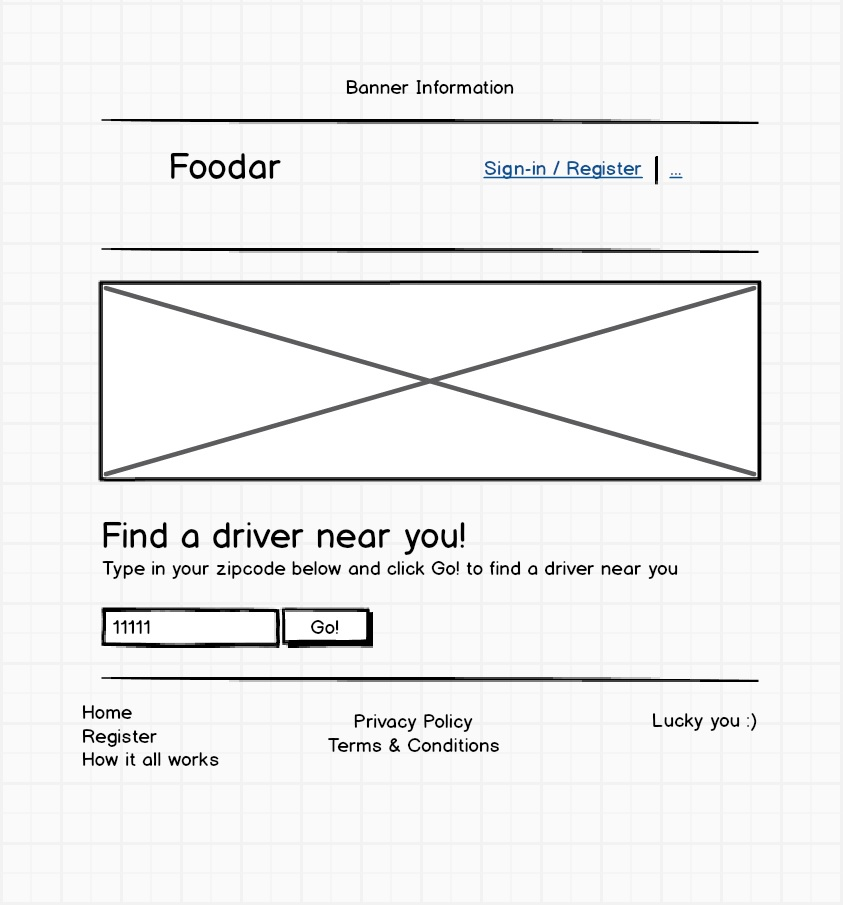
Continue to get the next requirements to implement.

**Customer Stories**

|  |  |
| --- | --- |
| Story | Home Page  Update index.html to randomly select recipe from list |
| Rationale | A different recipe should show when different users navigate to the website. |

*- Continue onto next page -*





*wireframe*: index.html

*- Continue onto next page -*

**The following requirements are WHAT YOU NEED TO IMPLEMENT**

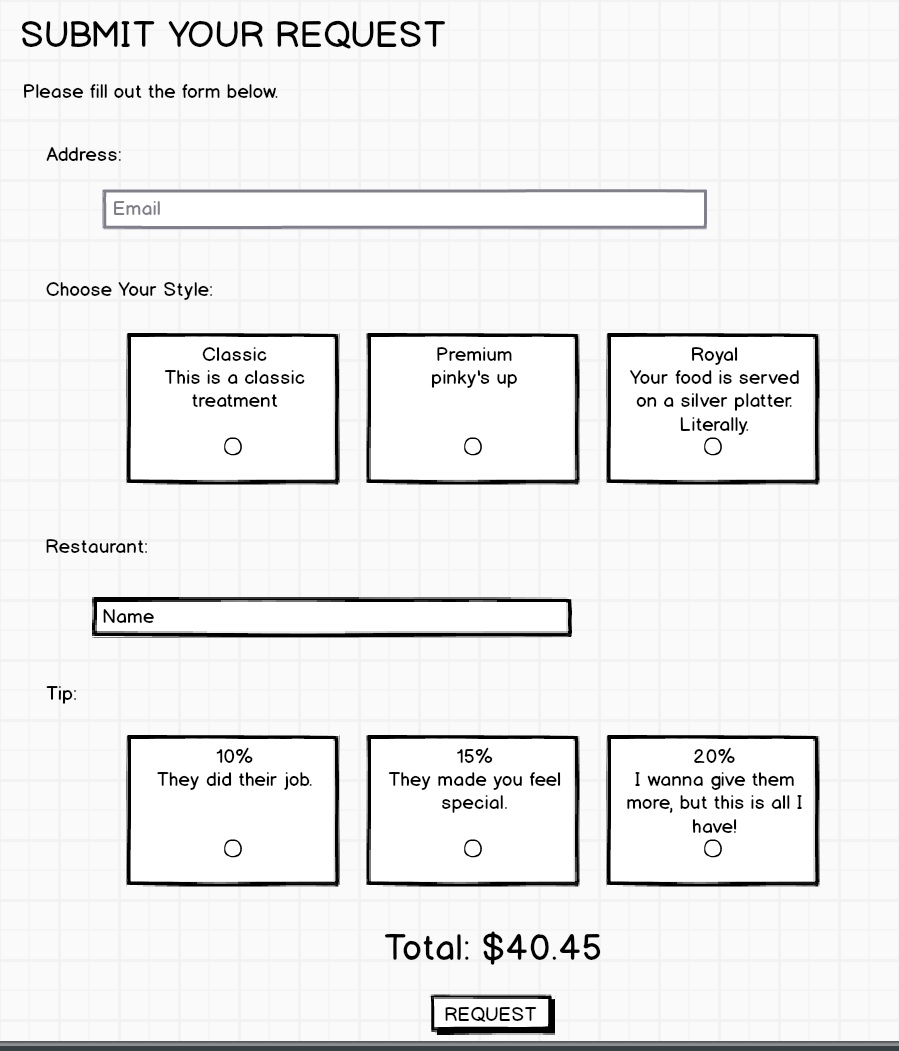
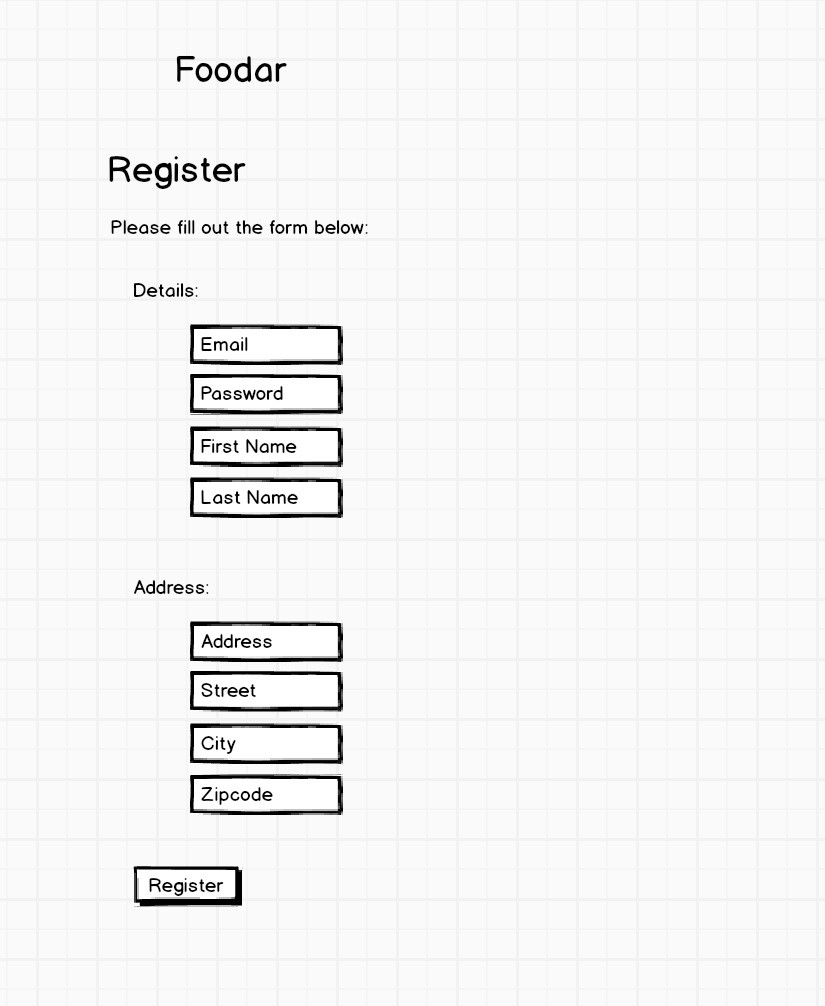
Use the below tables to understand what you will need to add to the project code

|  |  |
| --- | --- |
| Requirement | Home Page – contains() JavaScript function   1. create function to return true/false if a String is part of another |
| Number | 9 |
| Rationale | This function will be called in a different part of the program |

|  |  |
| --- | --- |
| Requirement | Home Page – zipcode search function   1. create a function that returns true/false if a zipcode is in a list |
| Number | 10 |
| Rationale | This function will be used in another Dive |

**Done.**

Now that you know a little about the system and a couple features you need to implement continue onto the next task.





**Objective:**

**HTML**

You've learned the basics of HTML and HTML5. You should be able to build a basic webpage given a set of requirements and layout elements to reflect a design.

**CSS**

You've learned CSS and how to include it in your webpages to develop a consistent style across several pages to form a website. You've learned tricks regarding color, position, and borders of elements and ways to style text and font.

**JavaScript**

This is the language to use to make your pages interactive. You've learned the fundamental concepts of the language from data types to functions and how to execute events and DOM manipulation.