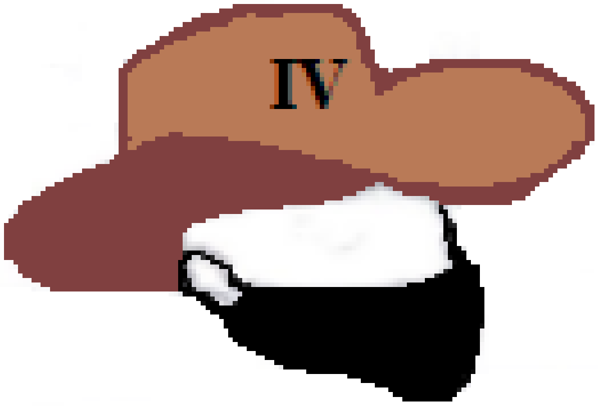
What's
For Lunch?



Lunch Decider Mobile Application

Requirements Document

Andrew Harrison

Ashton Webb

Dylan Miles

Keaton Whitaker

Sunil Rasaily

02/23/2023

Table of Contents

1.0 Introduction

1.1 Purpose

The purpose of this document is to describe the requirements of the Lunch Decider mobile application which, is to take the ease off of indecisiveness. Choosing what you want to eat can be a difficult task. Are you hungry for multiple things? Or are you the type of person to just not want to make decisions? Luckily this app is here to take all that weight off your shoulders and choose for you.

1.2 Outline

The rest of this document will go into detail about the requirements such as functional requirements, non-functional requirements, major functional requirements, and minor functional requirements. Identifying and addressing risks will be described. Targeted Users will be identified. Lastly the discussion of the projects Use Case Diagram.

2.0 Scope

The main objective of this project is to produce three prototypes that build from one another. Each prototype will be broken up into three separate parts. After the first prototype is finished the team will present it to the customer to get feedback. After feedback, the team will start the second prototype that addresses the issues in prototype 1. Once again, when prototype two is finished the team will present the product to the customer to get feedback. Once the team receives feedback about prototype two, the team will start on prototype three. This will be the final prototype the team will complete. During this, the team will fix up any last details that failed to be completed in prototype two. Prototype three will then be delivered to the customer for use.

3.0 Targeted Users

Are you unsure of where you want to eat and use an android or own a computer? Then you’re the application’s targeted audience. Anyone who has trouble deciding where they want to eat can easily use the lunch decider application to figure it out for them. However, the customer wants the team to think of them as a representative of Yelp. A Use Case Diagram showing the relationship the user and server have with the application will be shown in appendix 1.

4.0 Requirements

4.1 Functional Requirements

4.1.1 Spin the Wheel

Upon entering the application, the user will be able to automatically spin a wheel. This wheel will include the most popular restaurants/ user favorites. The user will also be able to re-spin as many times as the user would like. The wheel will not always stay the same you will be able to add various filters (price range, mile radius, etc.). Nevertheless, upon initial entry into the app, you will have the ability to spin a completely random wheel. Ultimately deciding where the user will go for lunch.

4.1.2 Enable Location Services

Geographical data will be provided by the user. The user will be prompted when first opening the app to allow permission for their current location to be used. If the user selects “allow”, the application will access their current location and populate into the wheel nearby restaurants, within the given mile radius. If the user selects “don’t allow” the app will ask for their current zip code and populate restaurants based on the zip code provided.

4.1.3 Search/Lookup Restaurant

Per the request of the client, there will be an option to search/look up restaurants in the area. The search/lookup option will be connected to yelp. Users will be able to search for a restaurant and it takes them directly to the yelp page of that restaurant, where the users will be able to see reviews, menus, and directions and find out whether that restaurant offers delivery or not. The app should run smoothly with Yelp.

4.1.4 Add Favorites

Users will be able to add restaurants of their choosing to their favorites. This will not only make that restaurant's information easy to find but this leads to the implementation of a favorites wheel. The favorites wheel will include only the restaurants that the user has selected as favorites. Meaning that the user will be able to spin the wheel of only the restaurants that the users really love and enjoy. Regardless of what the wheel lands on the user will love it because it is a favorite of theirs.

4.1.5 Add Criteria

The users will be able to add many different filters/criteria to better narrow down what the users want to be available on the wheel. Filters will include but will not be limited to price ranges, mile radius, how many stars the restaurant has, types of food, and much more.

4.2 Non-Functional Requirements

4.2.1 Usability

The app will be available on the Google Play Store for Android users.

If time permits, it will also be made web-based/ available to access via the web.

The interface will have friendly labels to reduce the learning curve.

4.2.1.1 User Interface

This application will use a user-friendly interface that allows you to interact with different features. Tabs, drop-down menus, swipe menus, and the use of a mouse and keyboard will be implemented. An animated wheel will be used for deciding what restaurant you eat from. There will be multiple tabs that will direct you to another page such as create lists, edit criteria, and spin options. The app will implement a search menu that allows you to search for a restaurant. There will also be an add to favorite button when searching for restaurants.

The user is going to be able to open up the app and just instantly spin the wheel to get an option of where to eat at. Not only can you just spin to find a random restaurant, but you'll also be able to spin by a favorites list and create your own personal lists with specific criteria to spin by. The application will also be able to store your previous list's data for easy and fast access the next time the user uses the app.

In addition, the customer would also like for us to have some specific features for the users. The customer would like for us to have a take me to Yelp area in the output choices. For this, whenever a restaurant choice is outputted, specific criteria based on what you filled out will appear as well as a link to yelp so you read reviews and write any if you'd like to. The customer would also like for us to create a web-based version of the application as well.

4.2.2 Reliability

The application must always be available every day during peak lunch hours of 11 am – 1 pm.

The application should be able to support at least 10,000 users at the same time while maintaining optimal performance.

The app should perform without failure 97% of the time.

The time it takes to restore the system if there is a failure should be no more than 10 minutes.

4.2.3 Performance

The app will only take up to 5 seconds to output results/ respond to user input.

The app will be compatible with all recent android models.

The app will utilize the Android SDK, including its location API.

The app will utilize the Yelp API.

4.2.4 Security

App permissions: the app will not request many permissions and will ensure that users are aware of the permissions that the users are granting.

The app will undergo regular updates and patches to ensure known vulnerabilities are fixed.

The app will only use trusted up-to-date libraries or avoid third-party libraries that may have unknown vulnerabilities.

The app will be regularly tested and audited to identify and fix any vulnerabilities.

5.0 Risks

Table 1.0 shows a list of possible risks that could happen during the project. These possible risks are discussed more fully in the next section.

Table 1.0

Risks	Likelihood	Mitigation
Lack of Communication	Medium	Meetings, Discord, Slack
Sickness	Medium – High	Online meetings, Backup members
Time Management	Low – Medium	Detailed Schedule

5.1 Potential Risks

Many risks are possible when working on a project. One such potential risk is a lack of communication. This can lead to project failure when there is not enough communication between team members among each other and communication between the team members and customer.

Another potential risk could be if a team member gets sick. Covid-19 and other viruses can really hurt the team by putting a team member out of commission especially if that team member has a big piece of the project to do.

Time management can play a risk during this project. Many of the team members have other classes on top of this one. Some students also have part-time or full-time jobs outside of college. These issues can mess things up such as scheduling issues and communication issues.

5.2 Addressing Risks

Addressing the potential risks is needed so that you can have a good backup plan. Lack of communication hopefully shouldn't be an issue for the team. The team will have a regular meeting every Wednesday and have a full detailed schedule for the semester. All of the team's tasks are divided up and backups are assigned as well.

When addressing sickness among team members, the team was able to come up with a quick solution. Almost 90% of the meeting taking place this semester happens on discord via voice chat. If a member were to fall ill, the member would still be able to attend normal meetings. The team will also have backups for each activity that is in progress, so if a team member falls ill someone else will be able to pick up the slack.

To face time management issues, the team has a few options. Luckily, most of the members of the team have taken a time management course in the past. The team already has a detailed schedule for the project. As long as most team members can keep up with the schedule then there should be no issues.

Appendix 1 Use Case Diagram

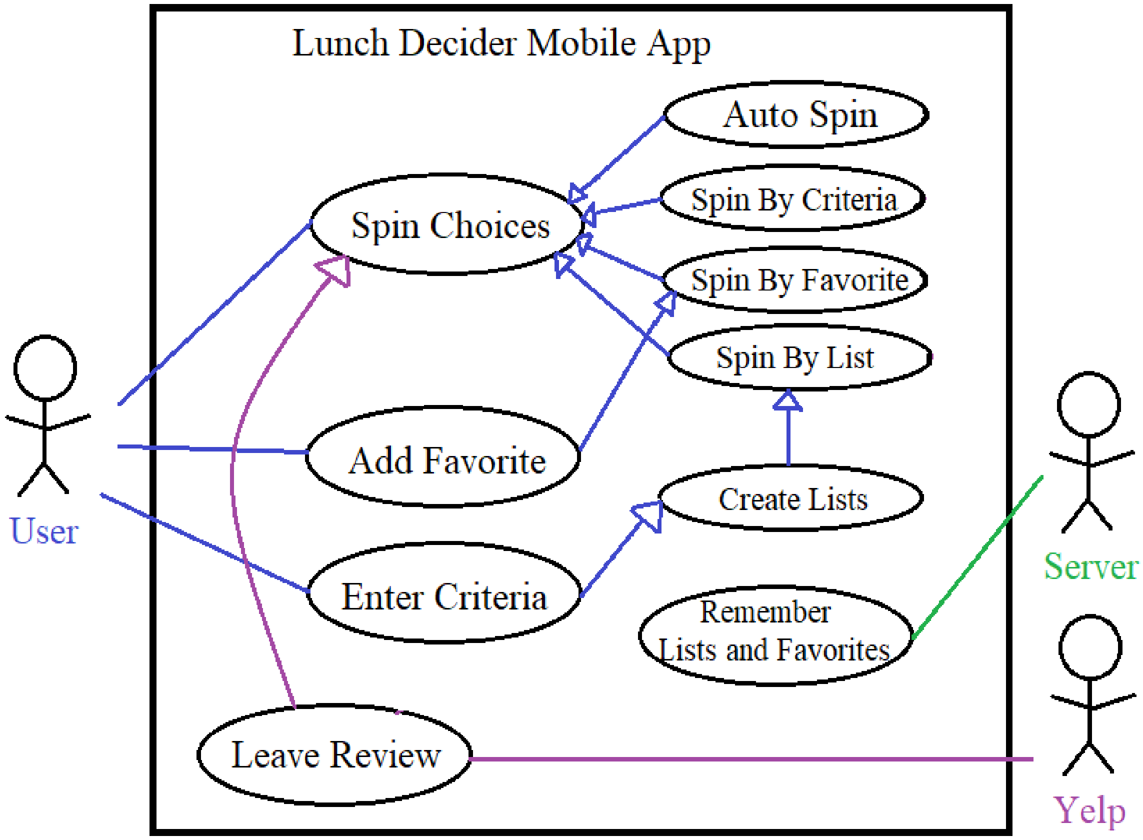


Figure 1.0