

Florida International University
Knight Foundation School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: YumFIU

Team Members: Nicholas de Souza, Mohammad Qureshi, Daniel Ortiz

Product Owner(s): Michael Mullin

Mentor(s): N/A

Instructor: Masoud Sadjadi

The MIT License (MIT)
Copyright (c) 2016 Florida International University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document presents the information necessary to gain a good understanding of YumFIU, an innovative campus food delivery app designed to cater to the needs of FIU students, particularly those with disabilities. In response to a request from FIU Embrace, this project aims to provide a user-friendly platform resembling Uber Eats and GrubHub, allowing students to order from on-campus restaurants for delivery by FIU Embrace staff. Our progress includes developing the front end and integrating various APIs to showcase restaurant information. This abstract provides an overview of the project's context, solution, current status, technologies used, project setup guidelines, and a glimpse into future features.

Table of Contents

INTRODUCTION	5
CURRENT SYSTEM	5
PURPOSE OF NEW SYSTEM	5
USER STORIES	
IMPLEMENTED USER STORIES	7
PENDING USER STORIES	10
PROJECT PLAN	
HARDWARE AND SOFTWARE RESOURCES	12
SPRINTS PLAN	13
<i>Sprint 1</i>	13
<i>Sprint 2</i>	13
<i>Sprint 3</i>	14
<i>Sprint 4</i>	15
<i>Sprint 5</i>	16
<i>Sprint 6</i>	17
<i>Sprint 7</i>	18
SYSTEM DESIGN	
ARCHITECTURAL PATTERNS	20
SYSTEM AND SUBSYSTEM DECOMPOSITION	21
DEPLOYMENT DIAGRAM	22
DESIGN PATTERNS	22
SYSTEM VALIDATION	23
GLOSSARY	37
APPENDIX	38
APPENDIX A - UML DIAGRAMS	38
<i>Static UML Diagrams</i>	38
<i>Dynamic UML Diagrams</i>	40
APPENDIX B - USER INTERFACE DESIGN	52
APPENDIX C - SPRINT REVIEW REPORTS	69
APPENDIX D - USER MANUALS, INSTALLATION/MAINTENANCE DOCUMENT, SHORTCOMINGS/WISHLIST DOCUMENT AND OTHER DOCUMENTS	74

REFERENCES	80
-------------------------	-----------

INTRODUCTION

Current System

Currently there is no system in place as this is the first semester of development for YumFIU.

Purpose of New System

The purpose of the new system, YumFIU, is to address the challenges faced by students with disabilities at FIU in navigating crowded on-campus restaurants. FIU Embrace aims to foster collaboration and inclusivity within the university community. Recognizing the difficulty that students with disabilities encounter in accessing on-campus dining services, FIU Embrace identified the need for a dedicated food delivery app. YumFIU is designed to provide a solution by allowing Embrace students to conveniently order food, with the added benefit of having their orders delivered by Embrace employees. The system seeks to enhance accessibility and improve the overall dining experience for individuals with disabilities at FIU.

USER STORIES

The following section provides the detailed user stories that were implemented in this iteration of the YumFIU project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

- Header Tabs
- Search Bar
- Restaurant Categories / Filter Bar
- Restaurant Items
- Yelp Data
- Google Places API Data
- Bottom Tabs
- About
- Menu Items
- Dynamic App Use
- Navigation
- Checkboxes
- View Cart
- Redux
- Modal

Pending User Stories

- Firebase Database to store order and user data and send to FIU Embrace staff
- Login and Account System with FIU Authentication
- Delivery Person Tag to give and track orders to an FIU Embrace staff member
- Reward System for coupons
- Orders Screen to view old receipts
- Account Screen to view account information
- Chat system to communicate with delivery personnel

- Screen for delivery driver to see order status and for restaurant staff to update
- App loading screen
- App logo

PROJECT PLAN

This section describes the planning that went into the realization of this project. This project incorporated the agile development techniques and as such required the sprints to be planned. These sprint plannings are detailed in the section. This section also describes the components, both software and hardware, chosen for this project.

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

- React Native and JavaScript
- Expo
- Redux
- Yelp API
- Google Places API
- NextCloud
- Xcode
- VSCode
- Git
- Node

Sprints Plan

Sprint 1

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. Become familiar with NextCloud
- User Story – 2. Evaluate design choices, styles, and colors
- User Story – 3. Research different application modalities (web vs mobile)
- User Story – 4. Determine tech stack based on modality

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story – 1. Become familiar with NextCloud [1 Story Point]
 - User Story – 2. Evaluate design choices, styles, and colors [4 Story Points]
- Mohammad Qureshi
 - User Story – 1. Become familiar with NextCloud [1 Story Point]
 - User Story – 4. Determine tech stack based on modality [4 Story Points]
- Daniel Ortiz
 - User Story – 1. Become familiar with NextCloud [1 Story Point]
 - User Story – 3. Research different application modalities (web vs mobile) [4 Story Points]

Sprint 2

The product owner chose the following user stories to be done during the sprint.

- User Story - 1. Set up the React Native build of the application and Repo [2 Story Points]
- User Story - 2. Begin development on header tabs for the app [3 Story Points]
- User Story - 3. Construct search bar for restaurant search [3 Story Points]
- User Story - 4. Work on restaurant categories [2 Story Points]
- User Story - 5. Build the RestaurantItem component [5 Story Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story - 5. Build the RestaurantItem component [5 Story Points]

- Mohammad Qureshi
 - User Story - 1. Set up the React Native build of the application [2 Story Points]
 - User Story - 2. Begin development on header tabs for the app [3 Story Points]
- Daniel Ortiz
 - User Story - 3. Construct search bar for restaurant search [3 Story Points]
 - User Story - 4. Work on restaurant categories [2 Story Points]

Sprint 3

The product owner chose the following user stories to be done during the sprint.

- User Story - 1. Search Functionality and Filter for Delivery / Pickup (APIs) [5 Story Points]
- User Story - 2. Bottom Tabs for App [2 Story Points]
- User Story - 3. Build the “About” Component [3 Story Points]
- User Story - 4. Build the “MenuItem” Component [3 Story Points]
- User Story - 5. Make About.js Dynamic [2 Story Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story - 1. Search Functionality and Filter for Delivery / Pickup (APIs) [5 Story Points]
- Mohammad Qureshi
 - User Story - 2. Bottom Tabs for App [2 Story Points]
 - User Story - 3. Build the “About” Component [3 Story Points]
- Daniel Ortiz
 - User Story - 4. Build the “MenuItem” Component [3 Story Points]
 - User Story - 5. Make About.js Dynamic [2 Story Points]

Sprint 4

The product owner chose the following user stories to be done during the sprint.

- User Story - 1. Add Navigation to the App [2 Story Points]

- User Story - 2. Checkboxes for Menu Items [1 Story Points]
- User Story - 3. ViewCart.js Button [2 Story Points]
- User Story - 4. Add Redux to half the app [5 Story Points]
- User Story - 5. Add Redux to the other half of the app [5 Story Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story - 1. Add Navigation to the App [3 Story Points]
 - User Story - 2. Checkboxes for Menu Items [2 Story Points]
- Mohammad Qureshi
 - User Story - 3. ViewCart.js Button [2 Story Points]
 - User Story - 4. Add Redux to half the app [3 Story Points]
- Daniel Ortiz
 - User Story - 5. Add Redux to the other half of the app [5 Story Points]

Sprint 5

The product owner chose the following user stories to be done during the sprint.

- User Story - 1. Add Modal to View Cart [5 Points]
- User Story - 2. Construct Firebase Database [5 Points]
- User Story - 3. Make Order Completed Sections [5 Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story - 1. Add Modal to View Cart [5 Points]
- Mohammad Qureshi
 - User Story - 3. Make Order Completed Sections [5 Points]
- Daniel Ortiz
 - User Story - 2. Construct Firebase Database [5 Points]

Sprint 6

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. Review codebase to transfer to the next semester [5 Story Points]
- User Story – 2. Refactor order completed feature code [5 Story Points]
- User Story – 3. Add survey stored on NextCloud [5 Story Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story – 1. Review codebase to transfer to the next semester [5 Story Points]
- Mohammad Qureshi
 - User Story – 3. Add survey stored on NextCloud [5 Story Points]
- Daniel Ortiz
 - User Story – 2. Refactor order completed feature code [5 Story Points]

Sprint 7

The product owner chose the following user stories to be done during the sprint.

- User Story - 1. Work on poster [5 Points]
- User Story - 2. Practice presentation and review rubric [5 Points]

The team members indicated their willingness to work on the following user stories.

- Nicholas de Souza
 - User Story - 1. Work on poster [3 Points]
 - User Story - 2. Practice presentation and review rubric [2 Points]
- Mohammad Qureshi
 - User Story - 1. Work on poster [3 Points]
 - User Story - 2. Practice presentation and review rubric [2 Points]
- Daniel Ortiz
 - User Story - 1. Work on poster [3 Points]
 - User Story - 2. Practice presentation and review rubric [2 Points]

SYSTEM DESIGN

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

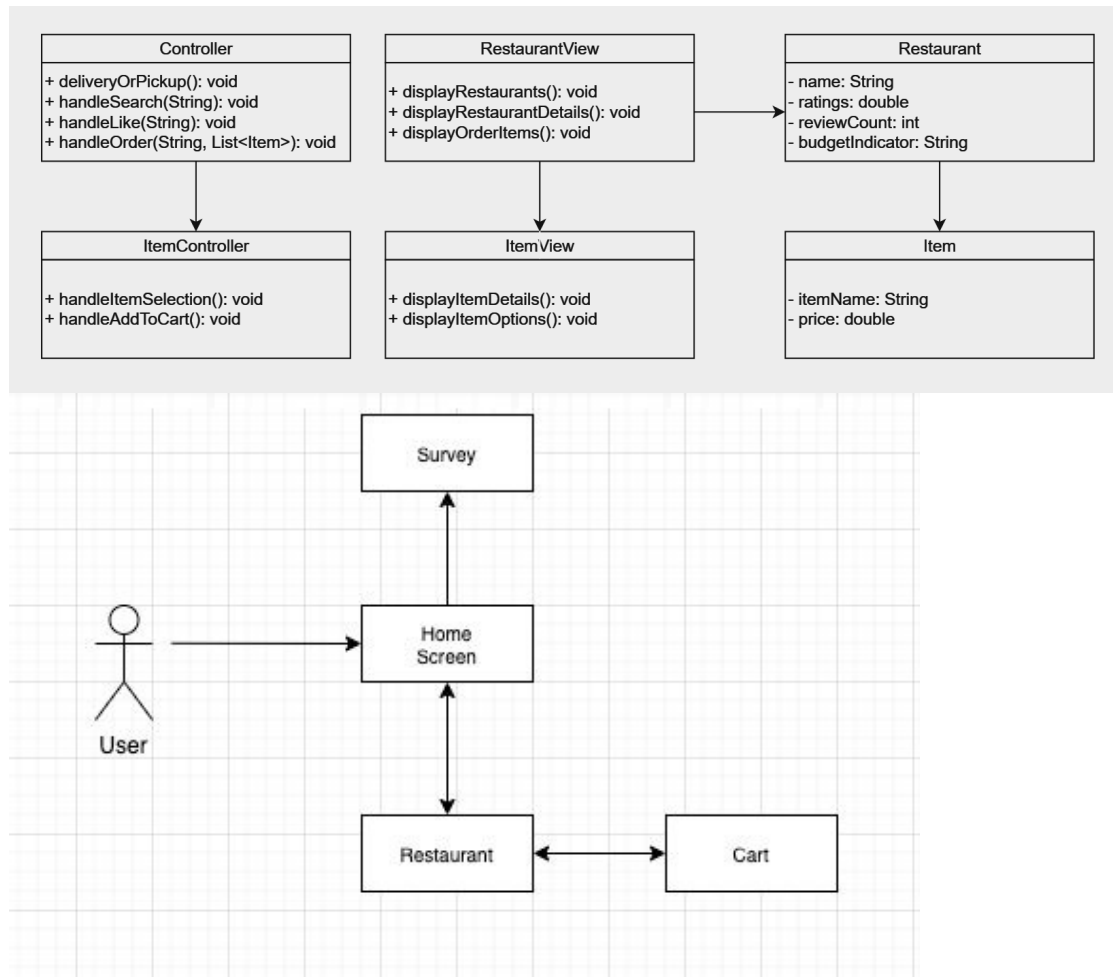
Architectural Patterns

This application uses the Flux Architecture which helps with the flow of data. It complements React Native's component-based structure by providing a unidirectional data flow. Everything begins when the user interacts with the application, with data flowing as the user changes the current view. The dispatcher is a central hub that manages the flow of data so as the user is using YumFIU, the dispatcher receives their actions and dispatches them to the registered stores.

System and Subsystem Decomposition

In the YumFIU app, the system decomposition is organized into distinct subsystems that seamlessly integrate to ensure efficient functionality. At the core is the Flux architecture, which acts as the overarching system responsible for managing data flow. The dispatcher, a pivotal component, serves as a central hub for directing user interactions throughout the application. This decomposition ensures a unidirectional flow of data, facilitating a smooth user experience as individuals engage with YumFIU. Additionally, the subsystems intricately collaborate to address the unique challenges faced by differently abled students, allowing them to effortlessly order food and receive deliveries from Embrace employees.

Deployment Diagram



Design Patterns

YumFIU utilizes a Component-Based Architecture pattern as UIs are built by creating and combining reusable components, each managing its own state. It also uses Redux, signifying that the project uses the Flux architecture, allowing for state management to be done in a predictable way.

SYSTEM VALIDATION

As the inaugural semester of this project wraps up, our primary focus has been on laying the foundation for a robust and user-friendly food delivery application, like Uber Eats and GrubHub. During this initial phase, our team has diligently worked on ideation, design, and the implementation of core features.

User Interface (UI) Components: The validation process has ensured that fundamental UI elements exhibit the expected behavior. This ensures a seamless and intuitive user experience as we move forward.

Data Integration: Integration with external APIs, including Yelp and Google Places, has undergone validation to confirm the accuracy and relevance of data displayed. This is foundational for providing users with up-to-date and comprehensive information about restaurants.

Navigation Flow: The application's navigation flow has been reviewed to ensure users can easily traverse through different sections.

Functionality Testing: Core functionalities, such as viewing restaurant items, adding them to the cart, and exploring dynamic app use scenarios, have been validated.

GLOSSARY

Header Tabs:

The clickable navigation elements typically located at the top of the application interface, allowing users to access different sections or functionalities.

Search Bar:

A user interface element that enables users to input search queries, facilitating the search for specific restaurants, items, or categories within the application.

Restaurant Categories / Filter Bar:

An interface component that allows users to filter and categorize restaurants based on specific criteria, enhancing the browsing experience.

Yelp API Data:

Information retrieved from the Yelp platform, including restaurant details, reviews, and ratings, to enhance the richness of data within the application.

Google Places API Data:

Data sourced from the Google Places API, providing geolocation-based information about restaurants, addresses, and other relevant details.

Bottom Tabs:

Navigation elements typically positioned at the bottom of the application screen, enabling users to switch between major sections or views.

Menu Items:

Individual food items or dishes offered by restaurants, displayed within the application for users to browse and select.

Navigation:

The overall structure and flow of movement within the application, including transitions between different screens and sections.

Checkboxes:

Interactive elements that users can select or deselect to indicate choices, preferences, or actions within the application.

View Cart:

Functionality allowing users to review and manage the items they have added to their shopping cart before completing the order.

Redux:

A state management library used in the application to manage and centralize the state, ensuring predictable state changes and improved maintainability.

Modal:

A temporary overlay or pop-up window that appears on top of the application's content, typically used to convey important information or prompt user actions.

APPENDIX

Sample code showing the implementation of Redux:

Test Case Example

```
let cartReducer = (state = defaultState, action) => {
  switch (action.type) {
    case "ADD_TO_CART": {
      let newState = { ...state };
      if (action.payload.checkBoxValue) {
        newState.selectedItems = {
          items: [...newState.selectedItems.items, action.payload],
          restaurantName: action.payload.restaurantName,
        };
      } else {
        newState.selectedItems = {
          items: [
            ...newState.selectedItems.items.filter(
              (item) => item.title !== action.payload.title
            ),
          ],
          restaurantName: action.payload.restaurantName,
        };
      }
    }
  }
}
```

Check to Ensure
Redux is Integrated
Properly for States

Sample code showing the Yelp API in use:

```
const getRestaurantsFromYelp = () => {
  const yelpUrl = `https://api.yelp.com/v3/businesses/search?term=restaurants&location=${city}`;

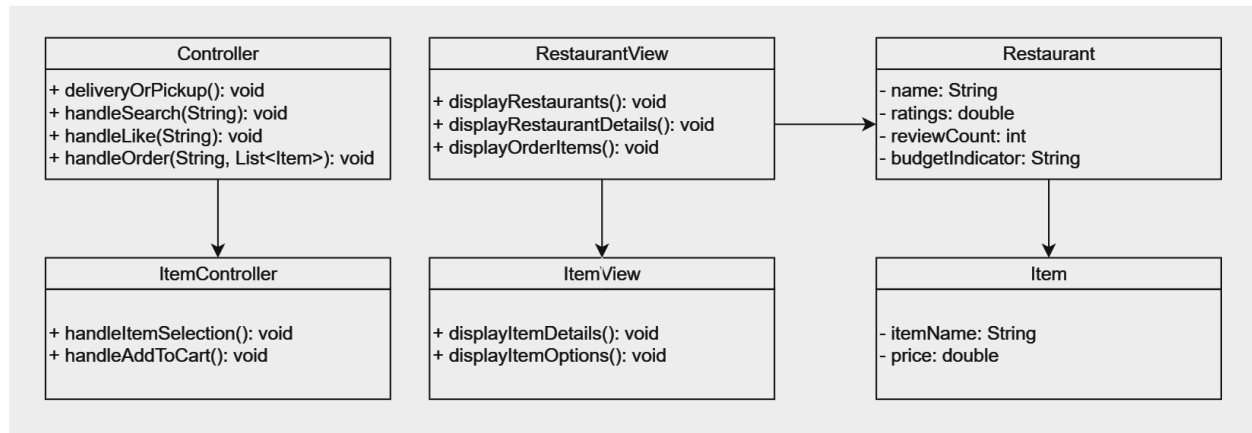
  const apiOptions = {
    headers: {
      Authorization: `Bearer ${YELP_API_KEY}`,
    },
  };

  return fetch(yelpUrl, apiOptions)
    .then((res) => res.json())
    .then((json) => {
      const yelpRestaurants = json.businesses.filter((business) =>
        business.transactions.includes(activeTab.toLowerCase())
      );
    });
};
```

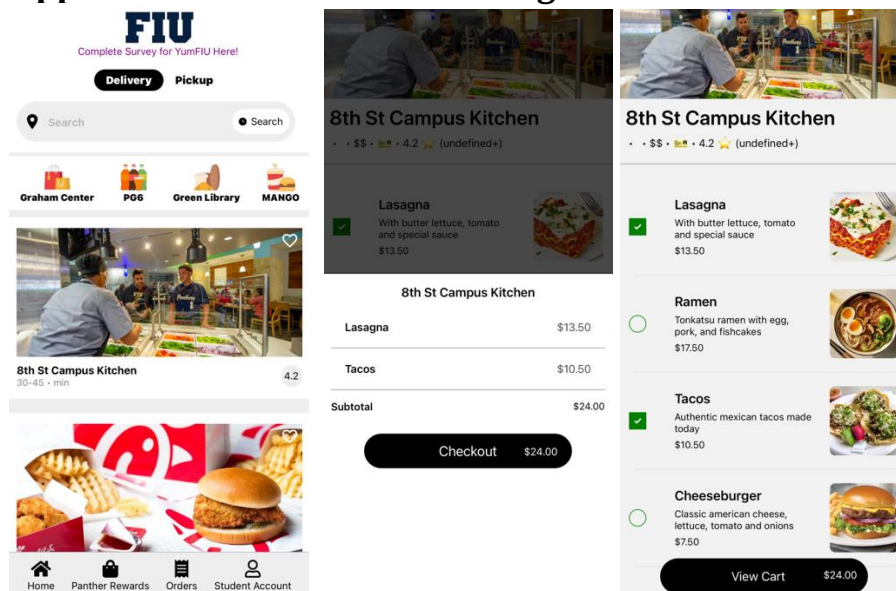
Sample Code showing the Google Places API in use:

```
export default function SearchBar({cityHandler}) {
  return (
    <View style={{marginTop: 15, flexDirection: "row"}}>
      <GooglePlacesAutocomplete
        query={{key: GOOGLE_PLACES_KEY}}
        onPress={({data, details = null}) => {
          const city = data.description.split(",")[0]
          console.log(city)
          cityHandler(city)
        }}
        placeholder='Search'
        styles={{
          textInput: {
            backgroundColor: '#eee',
            borderRadius: 20,
            fontWeight: "700",
            marginTop: 7,
          },
          textInputContainer: {
            backgroundColor: "#eee",
            borderRadius: 50,
            flexDirection: "row",
            alignItems: "center",
            marginRight: 10
          }
        }}
      />
    </View>
  )
}
```

Appendix A - UML Diagram



Appendix B - User Interface Design



Appendix C - Sprint Review Reports

Review 1

12/11/2023

Page 18 of 23

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story – 1. Become familiar with NextCloud 
- User Story – 2. Evaluate design choices, styles, and colors 
- User Story – 3. Research different application modalities (web vs mobile) 
- User Story – 4. Determine tech stack based on modality 

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Review 2

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story - 1. Set up the React Native build of the application and Repo [2 Story Points] 
- User Story - 2. Begin development on header tabs for the app [3 Story Points] 
- User Story - 3. Construct search bar for restaurant search [3 Story Points] 
- User Story - 4. Work on restaurant categories [2 Story Points] 
- User Story - 5. Build the RestaurantItem component [5 Story Points] 

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Review 3

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story - 1. Search Functionality and Filter for Delivery / Pickup (APIs) [5 Story Points] 
- User Story - 2. Bottom Tabs for App [2 Story Points] 

- User Story - 3. Build the “About” Component [3 Story Points] ✓
- User Story - 4. Build the “MenuItem” Component [3 Story Points] ✓
- User Story - 5. Make About.js Dynamic [2 Story Points] ✓

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Review 4

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story - 1. Add Navigation to the App [2 Story Points] ✓
- User Story - 2. Checkboxes for Menu Items [1 Story Points] ✓
- User Story - 3. ViewCart.js Button [2 Story Points] ✓
- User Story - 4. Add Redux to half the app [5 Story Points] ✓
- User Story - 5. Add Redux to the other half of the app [5 Story Points] ✓

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Review 5

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story - 1. Add Modal to View Cart [5 Points] ✓
- User Story - 2. Construct Firebase Database [5 Points] ✓
- User Story - 3. Make Order Completed Sections [5 Points] ✓

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected

- No movement to the product backlog is needed

Review 6

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story – 1. Review codebase to transfer to the next semester [5 Story Points] 
- User Story – 2. Refactor order completed feature code [5 Story Points] 
- User Story – 3. Add survey stored on NextCloud [5 Story Points] 

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Review 7

After a review, the implementation of the following user stories was accepted by the product owners: All stories were completed and accepted.

- User Story - 1. Work on poster [5 Points] 
- User Story - 2. Practice presentation and review rubric [5 Points] 

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- No stories were left unfinished or rejected
- No movement to the product backlog is needed

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Project Setup

1. Install Node v18.17.1
2. Install Xcode

3. Open Xcode once, accept conditions, and run a test project simulator for the iPhone 15 Pro
4. Clone repo
5. Enter the project terminal
6. run "npm install -g expo-cli"
7. run "npm run ios -- --reset-cache"

Running the Project

Have Xcode open in the background if it's not your primary IDE

- npm run ios (usually only use this)
 - Will automatically start the server and open the iOS simulator
- npm start (if you want to test another OS)
 - The terminal will give instructions on how to open different device simulators

APIs in Use

Make sure to update environment variables (API keys) so that the app runs as intended. You must get a new Google Places API key along with the Yelp API and Yelp Client API keys.

- Google Places API
 - Used to search for locations on the home screen of the app.
 - <https://developers.google.com/maps/documentation/places/web-service/overview>
- Yelp API
 - Used to display restaurant information including name, image, price, reviews, etc.
 - <https://docs.developer.yelp.com/docs/fusion-intro>

Future Features

Below are features that we would have liked to include but due to time constraints, we unfortunately could not. Feel free to look at this list as ideas that could be worked on in following semesters.

- Implementing authentication and a database so that account information could be stored.

- Creating a system so that FIU Embrace employees could begin taking orders from restaurants to deliver food to students.
- Creating a filtering system so that only open restaurants on the FIU MMC campus would show on the app.
- Implementing a payment processing system that would allow users to pay for their food.
- Implementing a navigation system so the food can be delivered to the right place.

REFERENCES

- Documentation for Google Places API:
 - <https://developers.google.com/maps/documentation/places/web-service/overview>
- Documentation for Yelp API:
 - <https://docs.developer.yelp.com/docs/fusion-intro>
- VSCode
 - <https://code.visualstudio.com/>
- Xcode
 - <https://developer.apple.com/xcode/>