SP-7: Owl Walk

Final Report

Version 1.1

CS4850 Section 1 Spring 2024

Professor Perry

Friday 26 April 2024

Project Team:

Roles	Name	Major Responsibilities
Project Owner	Faculty Member	N/A
Team Leader	Sean Suss	Team Management Requirements Analyst
Team Members	Ashton Forde	Lead Developer
	Fray Kriston	Project Manager
Advisor / Instructor	Sharon Perry	Facilitate project progress; advise on project planning and management.

Ashton Forde	Kristion Fray	Sean Suss

GitHub Link: https://github.com/ksuowlwalk2024/Owl-Walk-2024

Website Link: https://ksuowlwalk2024.github.io/

LINES OF CODE: 643 (spread over many different files)

Number of Components: 7 Screens, 9 images, 1 Logo, 8 Visual Studio Code Extension

1. Introduction

1.1 Overview

This report entails the culmination of our efforts throughout CS4850 over the semester. This report will place an emphasis on the design, development, and deployment of our application "Owl Walk", an interaction mobile navigation application developed to guide users throughout the Kennesaw State University campus'. This project was driven by the goal of allowing individuals to explore the different campuses.

1.2 Objectives & Scopes

This project's scope is defined by the intended audience and the technological requirements necessary to meet our goals. This application, designed for faculty, students, and guests, aims to promote campus exploration and allow self-guided tours. Key functionalities include point of interest information as well as navigation.

2. Project Management

The execution of Owl Walk required all team members to practice efficient project management practices. This section outlines our approach and includes the planning stages, team roles, and responsibilities.

2.1 Project Plan

The project started with a detailed planning phase. We placed and emphasized milestones throughout our project to ensure a timely delivery of components. We adopted a modular methodology in which we worked in small sections of the project and put things together to form the full application. This methodology allowed us to avoid feeling overwhelmed, as first-time developers, and make changes as needed. Weekly meetings were held to ensure progress, discuss challenges, and adjust deadlines as needed.

2.2 Team roles Aswell as responsibilities

- Project Manager (Kristion Fray): Oversaw the project's execution, while also ensuring team members met deadlines.
- Lead Developers (Ashton Forde & Sean Suss): Tasked with architecture as well as primary developer of this project. Responsibilities include coding core functions.
- UI / UX Designer (Kristion Fray): Designed User Interface & User Experience, while also ensuring application was intuitive and functional.
- Documentation Specialists (Sean Suss, Kristion Fray, Ashton Forde): Responsible for compiling this report, and upkeep of all documentation

2.3 Version Control System Description

Version control was managed using GitHub, which also facilitated collaboration and enabled our team to track changes throughout our project. The repository was structure was organized into different commits made by the development team. By using Git Bash on each developer's local machine and GitHub, we were able to track the different features and changes made in the app. This system facilitated efficient workflow among team members and allowed us to have a history of all project versions, which was invaluable for addressing any issues that may have arisen.

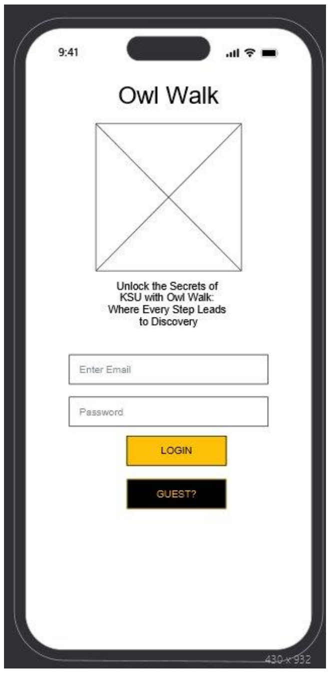
3. Software Design Architecture

The architecture of Owl Walk is designed to be robust, scalable, and responsible. This is done to cater to our target audience's needs. This section discussed architectural strategies, design patterns, and key interactions between system components.

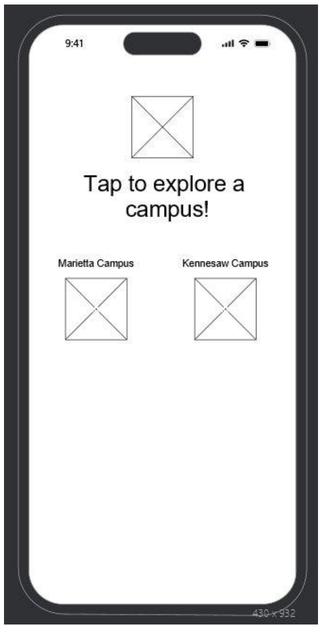
3.1 Architectural Drawings

We designed the architecture of the app to be as simple as possible for the targeted userbase of students, future students, facility, and guests. In the early stages of development, we created design mock-up of the different screens we will use inside the app. Listed below are the assorted designs we created for the different states/screens for the app.

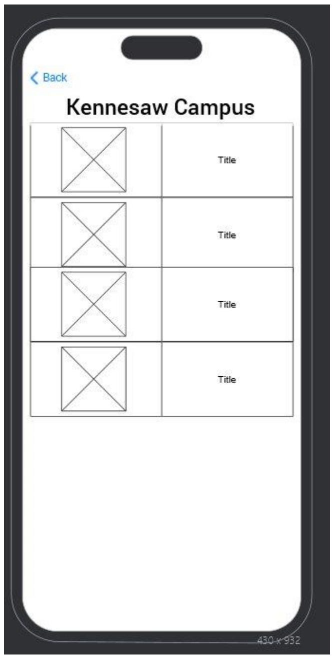
A. Login Screen


9:41
Owl Walk
Unlock the Secrets of KSU with Owl Walk: Where Every Step Leads to Discovery
Enter Email
Password
LOGIN
GUEST?
430 x 932

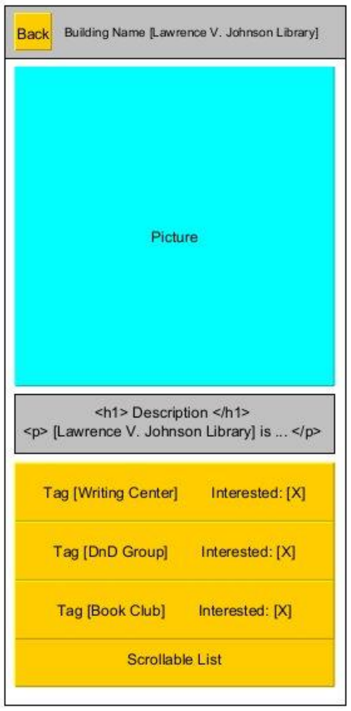
B. Campus Selection Screen



C. Landmark Selection Screen



D. Building Description Screen


Back
Building Name [Lawrence V. Johnson Library]
Picture
<h1> Description </h1>
<p> [Lawrence V. Johnson Library] is ... </p>
Tag [Writing Center] Interested: [X]
Tag [DnD Group] Interested: [X]
Tag [Book Club] Interested: [X]
Scrollable List

3.2 Interaction of Software Components

By leveraging React Native CLI, we gained access to many versatile software components that enriched our application's functionality. Our implementation utilized key features such as React Native Image, StyleSheets, StackNavigation, and Touchable to enhance the user experience.

The React Native Image component facilitated seamless integration of images from project directories, enriching the visual appeal of the app. With StyleSheets, we wielded the power to customize the app's appearance, ensuring a unique and captivating user interface. StackNavigation facilitated smooth navigation between nested screens, providing users with intuitive exploration pathways throughout the app.

Moreover, React Touchable enabled the creation of interactive components, triggering events upon user interaction, thereby enhancing user engagement and interactivity. These indispensable tools provided by React Native CLI were instrumental in bolstering the overall functionality and appeal of our application.

4. Implementation Details

The implementation phase of Owl Walk was very important in the transformation of our designed features and architectural plans into a functional system. This section outlines our core technologies used, the coding standards adopted, and the key functionalities implemented during this project.

4.1 Technologies Used

Owl Walk developed a combination of modern software development tools and technologies to ensure efficient performance and maintainability.

- Front end development utilized React.js to build a dynamic and responsive user interface. This choice was guided by Reacts component-based architecture, which allowed us to efficiently reuse code.
- Backend Development utilized Firebase Realtime Database to ensure authentication for the log-in screens. We plan to have Json files to use to store data of users for our app. This choice was guided by scalability as well as its ease of integration with various database systems.
- Database: Intergrated Realtime Firebase Databases with Log in Authentication

4.2 Coding Standards

To maintain code quality and ensure consistency across the project, we used the following code standards.

- Style Guide: Implemented extensions of Visual Studio Code such as
- Code Reviews: We used weekly code reviews to maintain efficient code and prevent potential bugs from making it into our final product.
- Documentation: We documented classes and methos usage to help maintain the code effectively and allow different developers to work together seamlessly.

4.3 Key functionalities implemented

Several functionalities were core to Owl Walk, with each designed to meet our specific needs the needs our target audience:

- User Authentication
- Point of Interest Viewing

5. Development Progression/Narrative

The development process began with the decision on what type of environment would best suit the Owl Walk app. We knew that this app would be best suited for React Native due to its flexibility of features and libraries it introduces. While using React Native, we had to determine which framework to use.

5.1 Frameworks

Our choices of frameworks were between React Native Expo and React Native Command Line Interface (CLI). React Native Expo includes Expo, which is a set of tools and services built around React Native aiming to simplify the development process. Expo is aimed for beginners in mobile development due to its higher level of abstraction it offers compared to React Native CLI to hide some of the more complex features. Expo offers unique features such as

- A host of built in APIs on installation
- Dynamic updating of code

- Its own mobile app that you can install on your mobile device where you can run your project on your own device

Expo, however, is restricted to only using APIs and components available to the Expo SDK.

React Native CLI is the official command line interface for React Native development. The benefits that React Native CLI offers are,

- More flexibility and control over project setup and code
- Direct access to the native code of your application allowing further customization
- Ability to link native modules manually
- Use of any native libraries needed

React Native CLI comes with a lot more steps with execution and setting up than Expo, with the use of a virtual machine required to view your app on your local machine.

As a team, we decided it was best to develop Owl Walk in React Native CLI because of the benefit of flexibility and control it offers. This will help the future scalability of the app if decided to be further worked on.

5.2 Virtual Device and Hot Reloading

To view the changes that we made through our code, we needed to install and set up a virtual device that would run on our local machine that would host the application. Android Studio was the software that we landed of the best virtual device integration for Owl Walk. Due to time constraints of the project, we decided to only focus on developing Owl Walk for Android devices.

Android Studio allows users to pick an android virtual device to run to mimic the use of a smartphone. With the app loaded on the virtual device, we can use “Hot Reloading” which is the act of editing code and having the edited code automatically update in the virtual device.

5.3 Visual Studio Code

For all the code development, we used the IDE Visual Studio Code with a combination of a host of extensions to create the right React Native environment needed. The extensions used in development were:

- React Native for es6/es7
- React Native Tools
- ES7+ React/Redux/React-Native snippets
- Babel JavaScript
- Code Runner
- C/C++

6. Testing Phase

Testing was a critical phase in the development of Owl Walk. Our testing phase ensured the application met all functional requirements and maintained high performance under various conditions. This section outlines our test planning, strategies, and execution that lead to the robustness and reliability of Owl Walk.

6.1 Test plan and strategy

The test plan was developed rather early in Owl Walk's life cycle. This was done for the team to clearly define our testing objectives and schedule testing timelines. The plan included types of testing, test cases, and a test environment.

- Types of Testing: Defined our scopes of testing, integration testing and system testing.
- Test cases: Detailed test cases ensured our design covered all user stories and functional requirements. Each test case was designed with a description, test steps, expected results, and an acceptance criterion.
- We used a dedicated testing environment to ensure accurate results.

The different types of testing strategies employed were manual testing and performance testing.

Test execution involved running the tests cases as per our test plan. The team uses a continuous deployment strategy to streamline testing when new code is committed. We used results logging and debugging

- Results Logging: All our tests' results were logged, and defects were tracked. This aided in issue resolution and maintained a clear audit log.
- Debugging: Bugs identified during tests were logged and prioritized based on how much they impacted user experience.

7. Challenges & Risk Assessment

Throughout the Owl Walk life cycle, our team faced various challenges and risks that could have impacted the project's outcome. This section outlines what challenges we faced, and the strategies we employed to help manage them. This section also delves into the overall risk assessment process that ensured our project stayed on track.

7.1 Overview of challenges faced

We faced several challenges during the completion of this project.

- Technical Challenges: Our main hurdle was working with the application environment. As first-time developers, debugging and getting our systems integrated proved to be a very complex task.
- Resource Limitations: Limited time also proved to create a hurdle for developers. As most time was spent just learning syntax and such, we were forced to rush our debugging/testing phase.

7.2 Risk Mitigation strategies employed

To address these challenges and mitigate risks, we employed the following strategies,

- Risk Identification: Before beginning the project, potential risks were identified through group brainstorming. Risks were categorized and then prioritized based on potential impact.
- Regular Meetings: The project strategies were reviewed regularly to adjust and make changes to incorporate any new information we learned. This technique proved crucial to ensure we met deadlines and helped team members.
- Communication Plans: We established clear communication channels and regularly updated each member to keep all team members engaged and informed. This transparency aided in managing team expectations and facilitated smoother collaboration.

8. Conclusions & Summary

As we conclude the development of Owl Walk, this report has captured the comprehensive journey of our project. The project has successfully met its objectives through delivering a robust, scalable, and function campus exploration application that allows its users to discover new areas on campus, as well as conduct self guided tours.

8.1 Key discoveries

- Functionalities and User Experience: Application delivers a range of features such as POI viewing, sliding and swipe gestures, and even POI information, which have significantly improved user experience. Usability tests ensure the applications design is easy to use, which makes it accessible to all intended users.
- Technical Robustness: Architectual and technological design choices have been effective in the application through its demonstration of reliability during testing phases.

8.2 Future techniques

- Engage target audience: Continuous engagement with the target audience would have allowed us to deliver a product more tailored to their needs
- Fully integrate database with Firebase RealTime Databases
- Add User functionality saved to each unique user.
- Create comment section on different building and landmark description screens to create community engagement

9. Appendix

Appendix A. Training Certificate For React Native: https://kennesawedu-my.sharepoint.com/:w:/g/personal/aforde3_students_kennesaw_edu/ERQDkDd8-AFFkurC8nIqzpwB_icy75QDO8sAqO6ksiPIHg?e=wDdceC