Final Report

Austin Franklin, Ari Le, Constance Smylie, & Lane Wilson

Capstone Project Spring 2024

The University of West Florida

Friday, April 26, 2024

CIS 4595: Capstone Project

Dr. Bernd Owsnicki – Klewe

Table of Contents

List of Figures

A Product Description

For our capstone project, we designed a platform called Wellness Quest. The idea behind the project was to create a platform that got users excited about being active. The platform's design leverages the concept of gamification. For example, users earn points by participating in challenges. The more challenges a user completes, the more points they earn. Users compete against other users in the local area by earning enough points to move further up the leaderboard. The platform functions as a tool to empower users to take control of their health by tracking their progress.

B Timeline Comparison

Our group had planned to complete a total of six user stories over two sprints. We aimed to complete three user stories in each sprint. During the first sprint, which lasted from January 28 to March 17, we completed the top two user stories. In the second sprint, which lasted from March 18 to April 22, we worked on the third top user story and two additional user stories. In total, we completed five user stories.

C Time Comparison

Wellness Quest users wanted to be able to create an account so that they could participate in weekly/ daily challenges. We facilitated this by implementing the functionality according to the use case, Create Account (UC_001). New users could log into their account after completing registration via the "Register" button on the Login screen.

Wellness Quest users wanted to be able to view their profile after logging in so that they could know their current standing. We facilitated this by implementing the Profile screen according to the use case, View Profile (UC_002). The software displays the user's profile information, such as the username, level, and active challenges when the user submits a login request from the Login screen.

Wellness Quest users wanted to be able to view all completed challenges so that they could showcase their efforts. We facilitated this by implementing the functionality according to the use case, Record Challenges (UC_003). The software displays a list of the user's completed challenges when they select the History tab.

Wellness Quest users wanted to be able to use their phone's GPS to accurately track the distance traveled during a working out. We facilitated this by implementing the functionality according to the use case, Track Distance (UC_004). The software displays the total distance the user has traveled after a workout when they initiated the workout tracking feature with their GPS location enabled on their phone.

Wellness Quest users wanted to view a challenge's potential points to be motivated to participate. Initially, we planned for the software to display the potential points earned by completing the challenge when the user selected the challenge on the Challenge Screen according to the use case, View Challenge's Potential Points (UC_006). Instead, we elected to display the potential points they could earn alongside the challenge name.

D Software Evaluation

D.1 Functionality

D.1.1 Testing methods, tools, and steps:

a. Methodology: Adopted a comprehensive testing approach involving both manual and automated testing methods.

b. Tools:

- Backend/API: using Postman
- Frontend/UI: using a web browser (Google Chrome) and an Android emulator (Android Studio).

- Others: GitHub, VSCode, Docker, MS Word for documentation, Google Drive for sharing files, and Discord for communication.

c. Steps:

- Develop test cases for both frontend and backend.
- Conduct backend testing using Postman.
- Conduct frontend testing using a Web browser (Google Chrome) and an Android emulator (Android Studio)/an Android phone.
- Regression testing to ensure the integrity of existing features.
- Inform teammates about test results and update testing documents.

D.1.2 Testing timeline

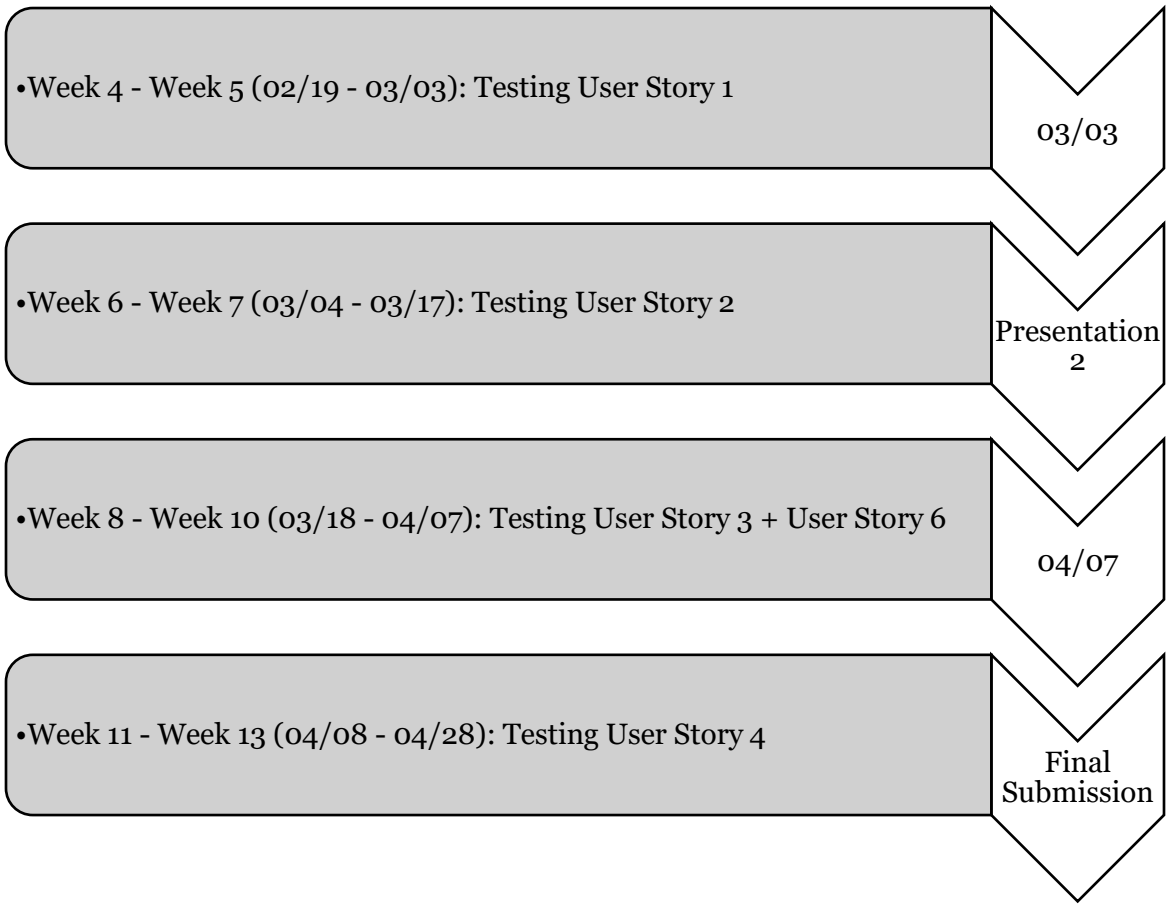

Figure 1 - Testing timeline

D.1.3 Open Issues/Functionalities

a. For the semester:

- GPS Tracking (User Story 4): While GPS tracking functionality is implemented, it requires further refinement to achieve optimal performance and user experience.
- Weekly Challenge (User Story 5): Due to prioritizing development efforts on other features, we haven't been able to allocate sufficient time to implement the Weekly Challenge functionality.

b. For the whole project:

- Set Higher Goal

- Log Weight and Height
- Seasonal Badge Challenge
- Find Friends

D.1.4 Testing Documentation

a. Test Plan (see Appendix A): Comprehensive document outlining the testing approach, objectives, scope, and tools for the project.

b. Frontend Test Cases (see Appendix B): Detailed test cases designed to verify the functionality of the frontend/UI of the application.

c. Backend Test Cases (see Appendix C): Detailed test cases focusing on verifying the functionality of the backend/server components.

d. Requirements Traceability Matrix (RTM) (see Appendix D): Matrix mapping requirements to test cases to ensure all project requirements are tested and fulfilled.

D.1.5 Functionality Summary

Currently, our group has implemented the following functionalities, each with varying degrees of completion and refinement:

a. User Registration and Login: Users can register accounts and log in securely.

b. Challenge Management: Users can view, add, and delete challenges from their profiles.

c. Challenge Progress Tracking/GPS Tracking: Automated tracking of challenge progress when the challenge is started, although further refinement is required for accuracy.

d. View Completed Challenges: Users can access and view their completed challenges via the History Page.

D.2 Security

D.2.1 Security methods and tools:

a. Methods:

- Security Requirements and Misuse Cases.
- Security Testing
- Threat Modeling

b. Tools:

- NPM Audit
- ZAP Proxy
- Retire.js
- Microsoft Threat Modeling Tool
- GitHub

D.2.2 Relevant Observations:

With our app being relatively simple, most of the security measures concerned the in and out flow of data into our application. With all data being handled through Sequelize, most issues involving injections are filtered out. Also, with HTTPS being in the workflow, the user's data should be secure from attackers listening on the connection.

D.2.3 Open Security Issues:

Right now, the only other security issue open is HTTP security headers which were not fully implemented due to time restraints. While these are likely not necessary, they can further safeguard the app. Also, anti-brute-forcing measures could be added to the login process. Other than that, most of the security documents could be updated to more closely cover the new features added.

D.2.4 Detailed Security Documentation:

a. Threat Identification

To identify the threats to our application, we used a threat model document template by OWASP. This document allowed us to take inventory of the interfaces, dependencies and levels of trusts used in our application. Along with this document, Windows Threat Modeling Tool was used to create threat modeling diagrams and to give suggestions of possible vulnerable points in the application.

Documents:

- Threat model.docx (see Appendix E): Threat model document using the OWASP template.
- Threat_Report3.htm (see Appendix F): Threat Report generated from Microsoft Threat Modeling tool on the login/registration use case.

b. Security Requirements

A security requirements document was established to demonstrate the security needs of the application. This document gave the requirements along with some example use/misuse cases. Also provided with this document were misuse case diagrams that demonstrated possible attack scenarios and methods to mitigate the attack. Most of these requirements were fully implemented with all being handled in some way.

Documents:

- Security Requirements.docx (see Appendix G): Contains security requirements for the project with each having an ID, example use case, example misuse case.
- Misuse[SecurityRequirmentID].png (see Appendix H): Misuse case example diagrams for each of the security requirements.

c. Data Protection Methods (Encryption, Hashing) Details

For the hashing of the user's passwords, bcrypt was used with the bcryptjs package. As for encryption, the communication between the client and server is secured using TLSv1.3.

d. Secure Coding Standards

OWASP Secure Coding standards has been listed as our coding standards, although with time constraints, not every measure has been used.

e. Code Verification

Most of the code was reviewed by the security lead to check for security errors.

f. Security Testing

In order to test the security of the application, both manual and automatic methods were used. I initially manually checked the security of the application; however later on ZAP Proxy's plugins were used to automatically scan the application. Using this, I found very few security issues that mostly consisted of CSP and information leak alerts.

g. Summary

Overall, the security of our application is in a good state. With the few security issues being minor and should be easy to fix. With HTTPS being implemented, all user data should be secure in transit and with the simplicity of the application, there are very few entry points that can be maliciously utilized.

E Work to Be Done

a) User Stories:

- Height and Weight Tracking
- Bonus XP for going past the scripted goal.
- Friend Networking
- Seasonal Badge
- Achievements with trophies

b) Bugs:

- GPS refinement on location accuracies.
- GPS not updating challenge status from ‘not started’ to ‘started’.

Appendix A: Test Plan

For detailed information regarding our testing plan, please refer to the Test Plan located in the Testing folder of our GitHub repository.

Appendix B: Frontend Test Cases

To access the Frontend Test Cases, please navigate to the Testing folder in our GitHub repository.

Appendix C: Backend Test Cases

To access the Backend Test Cases, please navigate to the Testing folder in our GitHub repository.

Appendix D: Requirements Traceability Matrix (RTM)

To access the Requirements Traceability Matrix (RTM), please navigate to the Testing folder in our GitHub repository.

Appendix E: Threat model.docx

To access the Threat model.docx, please navigate to the ‘Security Documents’ folder in our GitHub repository.

Appendix F: Threat_Report3.htm

To access the Threat_Report3.htm, please navigate to the ‘Security Documents’ folder in our GitHub repository.

Appendix G: Security Requirements.docx

To access the Security Requirements.docx, please navigate to the ‘Security Documents’ folder in our GitHub repository.

Appendix H: Misuse[SecurityRequirementID].png

To access the Misuse[SecurityRequirementID].png, please navigate to the ‘Security Documents’ folder in our GitHub repository.

Appendix I: 2024-04-12-ZAP-Report-localhost.html

To access the Threat model.docx, please navigate to the ‘Security Documents’ folder in our GitHub repository.

Appendix J: Security_Assessment_030824.docx

To access the Security_Assessment_030824.docx, please navigate to the 'Security Documents' folder in our GitHub repository.

Appendix K: ZAP Active Scan (3_25_24).docx

To access the ZAP Active Scan (3_25_24).docx, please navigate to the ‘Security Documents’ folder in our GitHub repository.