Software Requirements Specification for Software Eng 4G06: subtitle describing software

Team 2, Parnas' Pals
William Lee
Jared Bentvelsen
Bassel Rezkalla
Yuvraj Randhawa
Dimitri Tsampiras
Matthew McCracken

April 5, 2023

Contents

Revision History

Date	Version	Notes
October 1, 2022	1.0	Volere template outline, Functional Requirements
October 5, 2022	1.1	Completed and submitted SRS document
April 4, 2023	2.0	Revision 1

1 Abbreviations and Acronyms

symbol	description
A	Assumption
API	Application Programming Interface
DD	Data Definition
GS	Goal Statement
LC	Likely Change
NFR	Non-Functional Requirement
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
T	Theoretical Model
UI	User Interface
UX	User Experience

2 Project Drivers

2.1 The Purpose of the Project

2.1.1 The User Business or Background of the Project Effort

With the increase in media consumption across the world, there has been a greater focus on several previously underrepresented or inaccessable niches, such as health and fitness. Despite the growth and presence of fitness in social media, there are still large barriers to entry that make it intimidating to get started or get accurate information that would aid individuals in their fitness journeys. A lot of the media available online is either behind a paywall, or structured in undigestible video and written formats. The lack of a free, centralized system that contains media generated and verified by fitness enthusiasts alike spurred the idea for this application. This application aims to bridge the gap that exists in the online fitness world by allowing individuals to create and track workouts of their own, search and share workouts created by others, and review and discuss what they find personally works and doesn't work for them. Creating a collaborative online fitness environment allows individuals to start or expand their fitness journey without looking in numerous locations or paying money for programs that may not work for them.

2.2 Stakeholders

1. Fitness Enthusiasts - Anyone interested in exploring other fitness routines, creating their own routines, and tracking their own personal progression towards goals.

2. Personal Trainers - Olympian provides the ideal platform for trainers to share routines and goals with their clients.

3. Fitness Advertisers - One avenue of monetization that Olympian could take is running advertisements. Although these advertisements could fall into any category, the largest stakeholders will be Fitness Advertisers, as the users of Olympian will be heavily involved with fitness, and thus most likely to buy fitness products.

4. Dr. Smith and Course Teaching Assistants - Dr. Smith and the course TAs (Teaching Assistants) are invested in the application in terms of tracking its progress and completion along with its scope and complexity.

3 Project Constraints

3.1 Mandated Constraints

3.1.1 Solution Constraints

1. **Description:** The product shall operate its back-end server with Node.js.
   Rationale: The product depends on the functionality provided by many libraries unique to Node.js.
   Fit Criterion: All back-end server libraries used are Node.js libraries, with a Node.js back-end.

3.1.2 Implementation Environment of the Current System

The product will be launched as a web-app on the internet, and as a mobile app. There is no hardware or otherwise physical integration of the product.

3.1.3 Partner or Collaborative Applications

N/A

3.1.4 Off-the-Shelf Software

N/A

3.1.5 Anticipated Workplace Environment

N/A

3.1.6 Schedule Constraints

The product timeline will follow the schedule as laid out in the course outline by Dr. Smith.

Team Formed, Project Selected	September 19	0%
Problem Statement, Development Plan	September 26	2%
Requirements Document Revision 0	October 5	5%†,‡
Hazard Analysis 0	October 19	3%†
V&V Plan Revision 0	November 2	5%†,‡
Proof of Concept Demonstration	November 14–25	5%*
Design Document Revision 0	January 18	5%†,‡
Revision 0 Demonstration	February 6–February 17	10%*
V&V Report Revision 0	March 8	5%†,‡
Final Demonstration (Revision 1)	March 20–March 31	20%*
EXPO Demonstration	April TBD	10%*
Final Documentation (Revision 1)	April 5	30%*,‡

- Problem Statement
- Development Plan
- Requirements Document
- Hazard Analysis
- Design Document
- V&V Plan
- V&V Report
- User's Guide
- Source Code

3.1.7 Budget Constraints

N/A

3.1.8 Enterprise Constraints

N/A

3.2 Naming Conventions and Terminology

Below is a glossary of terms, acronyms and abbreviations used by stakeholders involved in the product's scope:

- **Repetitions:** The number of times a motion will be repeated.
- **Exercise:** An entity describing a physical movement to be performed with optional descriptors and any combination of the following quantifiers: Repetitions, Sets, Weight, Distance, Time, and Rest Time.

- **Routine:** A routine (or workout routine) is composed of a sequence of exercises performed in order.

3.3 Relevant Facts and Assumptions

- Users understand how to operate a mobile device
- Users are able to search for interests (cycling, weightlifting, cardio, etc.)

4 Functional Requirements

4.1 The Scope of the Work

4.1.1 The Current Situation

N/A

4.1.2 The Context of the Work

Refer to Figure 1 in "System Design" document for System Context Diagram

4.1.3 Work Partitioning

Event Name	Input and Output	Summary
Creating a Program	Program Creation including Program Name (string), Program Tags, Program Viewing (public, private, friends) *input*	Finalizing all program creation inputs from user
Searching for a Program	Program name string/Program type string/Program creator string *input*	Searching and browsing for a specific program or type of program based on search parameters
Sorting search results	Results sorted and displayed *output*	Sorting search results in a relevant way that is useful to the user
Adding performed sets and reps for exercise	Entering rep and set information as integers *input*	Tracking workout information by adding sets and reps performed during exercise
Registering an account	Creating a new account with user's name, user's email, user's username, and user's password *input*	This creates a new user and appends it to the database
Adding program to profile	Appending a program to personal profile *input*	Users can add programs they find to their profiles to use a later time or immediately
Logging into account	Entering username and password as strings *input*	This authenticates and authorizes the user to access their account
Providing users with suggested programs and workouts	Generating and displaying programs and workouts tailored to individual users *output*	Using specific algorithms that find programs that suit the users needs

4.2 Business Data Model and Data Directory

N/A

4.3 The Scope of the Product

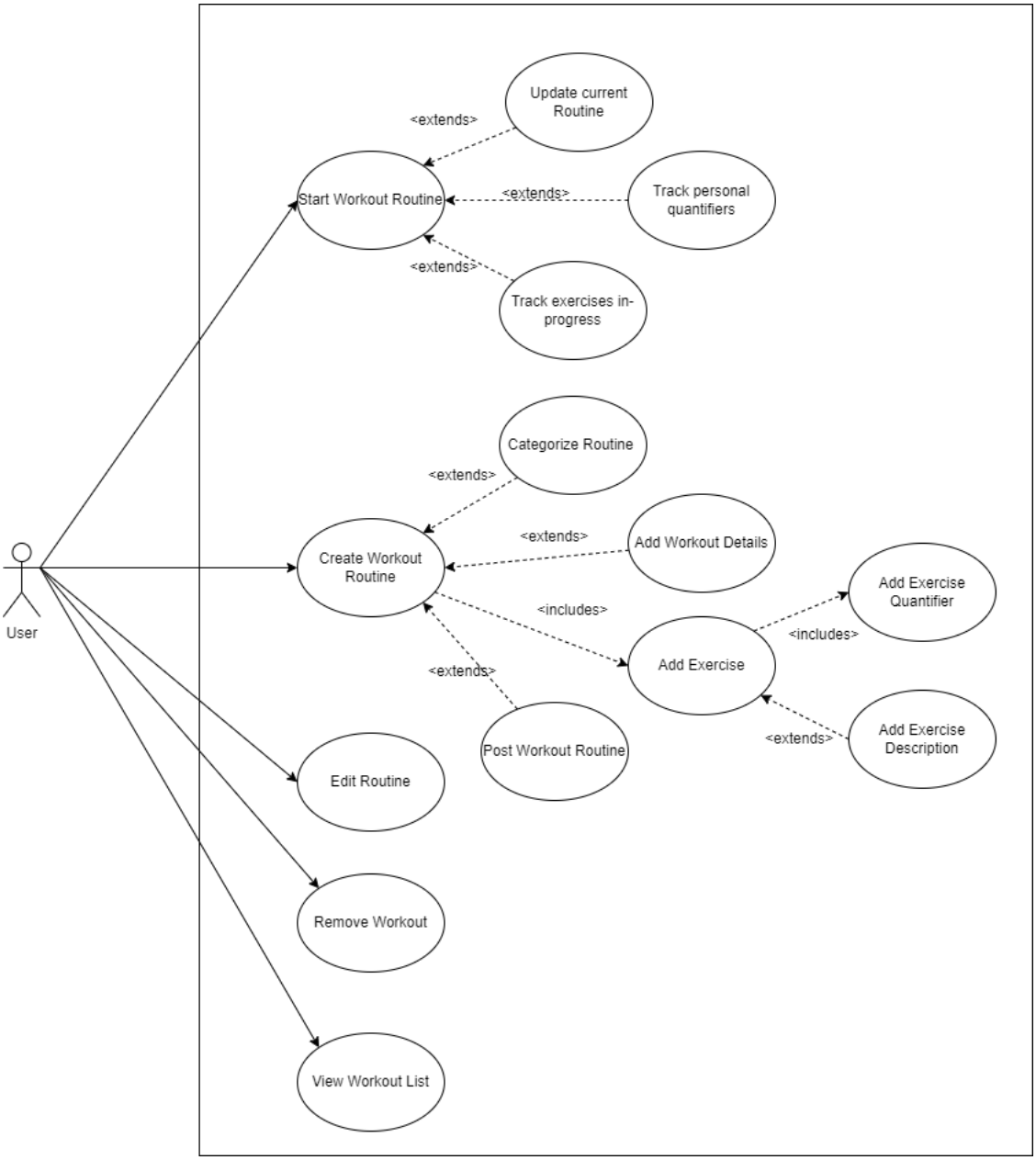
4.3.1 Product Boundary

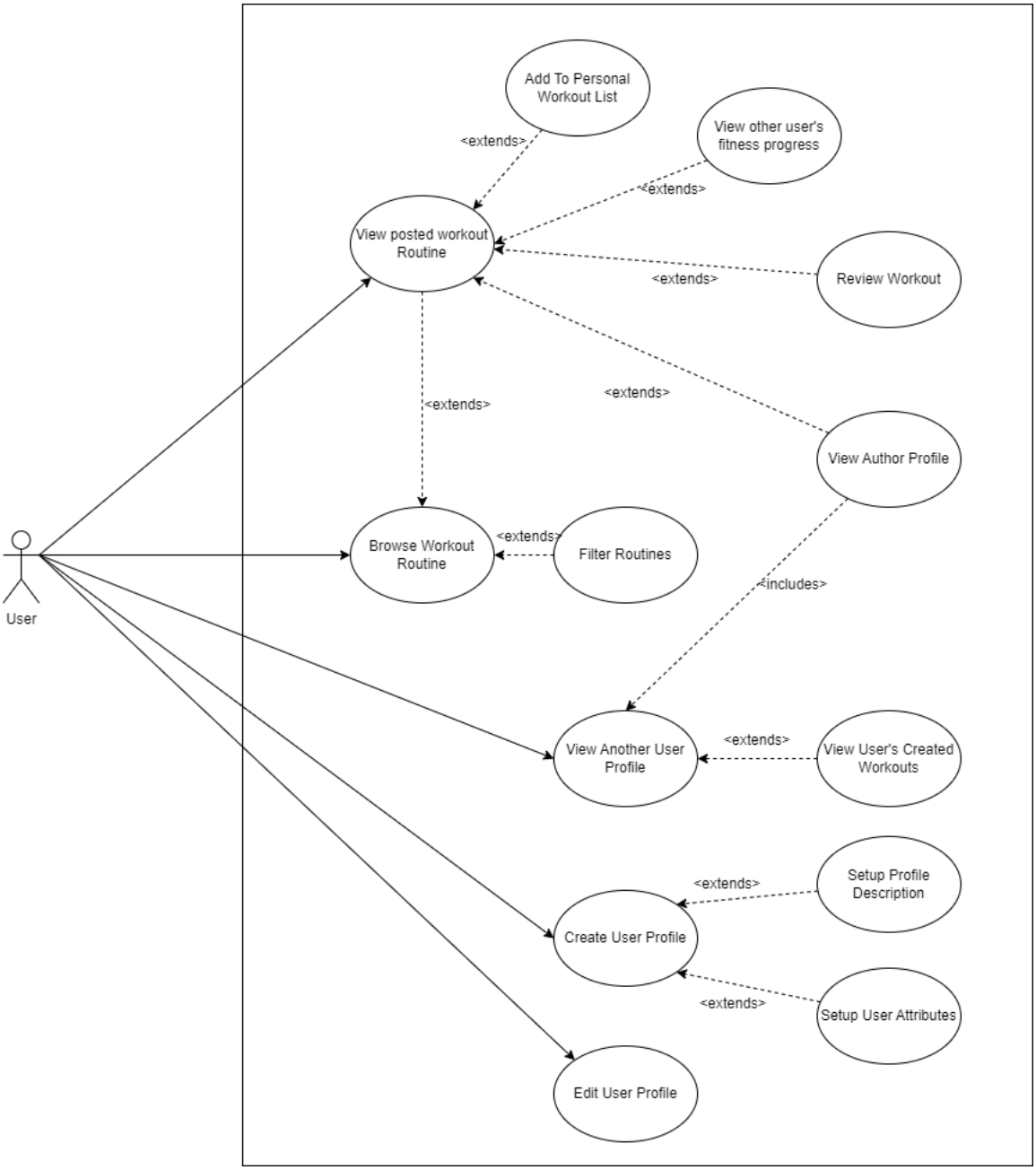
The application will allow users to create workout programs, follow other users, and track workouts.

4.3.2 Use cases

- View posted workout routine
    1. View other user's fitness progress
    2. Add Personal Workout List
    3. View workout Author
    4. Review workout
- Browse Workout routines
    1. Filter routines
- View Another User's Profile
    1. View user's created routines
- Create User Profile
    1. Setup profile description
    2. Setup attributes
- Edit User Profile
- Start workout routine
    1. Track exercises in-progress
    2. Track personal Quantifiers
    3. Update current routine
- Create workout routine
    1. Post workout routine
    2. Categorize routine
    3. Add workout length details
    4. Add exercise
        (a) Add Quantifier
        (b) Add Workout Descriptions
- Edit Routine
- Remove Routine
- View Workout List

4.3.3 Use case Diagram





4.4 Functional Requirements

R1: Description: The system shall allow the user to create a workout routine.
Rationale: To allow the user to save their own workout routines.
Fit Criterion: The user created routines are accessible after creation.

R2: Description: The system shall allow the user to add and remove individual exercises in order to a created workout routine, with a maximum of 20 exercises per workout

routine.
Rationale: A workout routine is composed of a sequence of exercises performed in order, which the user should be able to add and remove as they wish to create the desired routine.
Fit Criterion: The user is able to add and remove exercises from a created workout routine.

R3: Description: The system shall allow the user to add or remove quantifiers to a given exercise.
Rationale: A single exercise requires quantifiers including but not limited to sets, reps, weight (light, medium, heavy), time, rest time to adequately specify how it is to be performed within a given routine.
Fit Criterion: The user is able to add quantifiers to a given exercise.

R4: Description: The system shall allow the user to publicly post a workout routine.
Rationale: To be able to display their workout routine to other users.
Fit Criterion: On publication of a routine, another user should be able to access and view the routine.

R5: Description: The system shall allow a user to save and view a performed workout.
Rationale: To allow a user to keep track of their current workout, and review their previous workouts when doing them again. This is especially helpful for ensuring progressive overload. That is, doing a little bit more than last time.
Fit Criterion: The user is able to save a performed workout and review that data in the future.

R6: Description: The product shall allow a user to browse and search for workout routines.
Rationale: To make publicly posted workout routines discoverable and for users to find routines that cater their fitness goals.
Fit Criterion: The user is able to browse and search for workout routines. The returned routines contain words that the user searched for.

R7: Description: The system should allow a user to create a profile, with a username between 1 and 25 characters.
Rationale: To display social, informational content to other users.
Fit Criterion: The user is able to create a profile. If the user does not meet the character requirements or includes symbols or characters that aren't allowed they will be informed of the username requirements through a text output.

R8: Description: The system shall allow a user to search for and view another user's profile.
Rationale: To allow a user to determine if another user has similar fitness goals or similar workout routines.
Fit Criterion: The user is able to view the profile of another user after searching for the target profile username.

R9: Description: The system shall allow the user to create and view a goal in the form of an exercise and a metric pair. For example, "Bench press - 100kg".
Rationale: This allows the user to set and witness progression towards their fitness goals.
Fit Criterion: The user is able to create and view goals.

R10: Description: The system shall allow the user to create progress points towards a specific goal. A progress point must be associated with a specific goal, and must exist as a date metric pair. For example, "09/04/22 - 96kg" under a "Bench Press - 100kg" goal. The metric type must match the goal metric type, for example kg.
Rationale: Being able to track progress towards set goals can help encourage more progression until the goal is reached.
Fit Criterion: The user is able to create progress points towards specific goals.

R11: Description: The product shall be able to visually display fitness progress towards set fitness goals.
Rationale: To help the user determine progress towards fitness goals.
Fit Criterion: The user is able to view progress points toward a set fitness goal.

5 Nonfunctional Requirements

5.1 Look and Feel Requirements

5.1.1 Appearance Requirements

- N/A

5.1.2 Style Requirements

NFR1: The product shall be minimalistic in appearance.
Rationale: Users will have an easier time learning how to utilize the app. The minimalistic look will aid users in navigating the app effectively.
Fit Criterion: The application shouldn't contain more than five clashing colours on one page. The application shouldn't contain large amounts of text scattered throughout pages (text should be coheisve and grouped).

5.2 Usability and Humanity Requirements

5.2.1 Ease of Use Requirements

NFR2: The product shall use fonts of readable size (no less than size 12) to the target user group.

Rationale: Users will have an easier time navigating the application and reading instructions/information displayed.
Fit Criterion: Application will utilize at minimum fonts of size 12.

5.2.2 Personalization and Internationalization Requirements

- N/A

5.2.3 Learning Requirements

NFR3: The product shall be able to be used by untrained fitness enthusiasts and amateurs alike, who receive no training before using it.
Rationale: Users of differing athletic backgrounds will be able to utilize the app for their required purposes.
Fit Criterion: Application will ensure a small learning curve, enforced by the minimalistic look and provided text prompts.

5.2.4 Understandability and Politeness Requirements

NFR4: The product shall use symbols, text, and picture prompts and instructions to provide users with an intuitive and efficient experience.
Rationale: Users will be able to follow text prompts and images to navigate the application for their required purposes.
Fit Criterion: Application will increase understandability through the usage of text, symbols, and images.

5.2.5 Accessibility Requirements

- N/A

NFR5: The product shall make use of sufficiently contrasting colours.
Rationale: Contrasting colours will make it easier to navigate through the application and identify different pages and prompts.
Fit Criterion: Applicaiton will utilize colours that are easily distinguishable from one another.

5.3 Performance Requirements

5.3.1 Speed and Latency Requirements

- N/A

5.3.2 Safety-Critical Requirements

- N/A

5.3.3 Precision or Accuracy Requirements

- N/A

5.3.4 Reliability and Availability Requirements

- N/A

5.3.5 Robustness or Fault-Tolerance Requirements

- N/A

5.3.6 Capacity Requirements

- N/A

5.3.7 Scalability or Extensibility Requirements

- N/A

5.3.8 Longevity Requirements

- N/A

5.4 Operational and Environmental Requirements

5.4.1 Expected Physical Environment

- N/A

5.4.2 Requirements for Interfacing with Adjacent Systems

NFR6: The application shall operate on iOS devices
Rationale: Users on i0S devices will be able to utilize the application.
Fit Criterion: The application will be completely tailored to i0S devices.

5.4.3 Productization Requirements

- N/A

5.4.4 Release Requirements

- N/A

5.5 Maintainability and Support Requirements

5.5.1 Maintenance Requirements

- N/A

5.5.2 Supportability Requirements

- N/A

5.5.3 Adaptability Requirements

- N/A

5.6 Security Requirements

5.6.1 Access Requirements

NFR7: The application must not display other users private details to the user.
Rationale: Private user information such as the account password must be kept private to prevent account hacking.
Fit Criterion: Application will not display sensitive user information to other users.

5.6.2 Integrity Requirements

NFR8: Passwords must be encrypted with SHA-256 when stored.
Rationale: Passwords must be encrypted to ensure they are not easily hackable or discoverable.
Fit Criterion: Passwords will be encrypted when they are stored in the database.

5.6.3 Privacy Requirements

- N/A

5.6.4 Audit Requirements

- N/A

5.6.5 Immunity Requirements

- N/A

5.7 Cultural Requirements

5.7.1 Cultural Market Requirements

- N/A

5.7.2 Cultural Diversity and Inclusion Requirements

- N/A

5.8 Legal Requirements

5.8.1 Legal Compliance Requirements

- N/A

5.8.2 Standards Compliance Requirements

- N/A

6 Traceability Matrices and Graphs

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11
NFR4	X										
NFR7								X			
NFR8							X				

Table 1: Non-Functional to Functional Requirement Dependency Matrix

7 Project Issues

7.1 Open Issues

- **Issue 1:** Users may have discomfort using an app to track their workouts over the traditional pen and paper method. The business event of tracking personal quantifiers is of concern with this issue.
- **Issue 2:** It may be unfavourable to users with existing programs and workout data to migrate to a new application.
- **Issue 3:** Users' personal methods of working out - various quirks and features - may not be covered by the app's program creation capabilities.
- **Issue 4:** There is a concern with the app's compatibility with various devices. Integral UI/UX features of the app such has haptics, pop-ups, and dragging features may not work with various devices - hindering the overall experience for some users.

7.2 Off the Shelf Solutions

7.2.1 Ready Made Products

TeamBuildr

- TeamBuildr is an exercise programming app for large groups used at a professional level. Used for teams, specifically at a high level. There is no present social aspect. It is also a paid service.
- Program creation is a very complex process. This app has a solution that has been tried and tested by multiple users at a high level.

JEFIT

- JEFIT lets you create and track workouts. It does not feature user-generated content but has paid template programs. It has functionality allowing users to post progress pictures.
- JEFIT could be used as a foundation for adding required features if acquired.

7.2.2 Reusable Components

One Rep Max Calculator

- The npm library one-rep-max, can be used to satisfy the requirements of providing user statistics. The library offers multiple recognized methods of calculating users' maximum weight for one repetition.

- Event: Track Personal Quantifiers

Content-Based Recommender

- The npm library content-based-recommender can help display recommended content for the user based on user inputted parameters.
- This addresses the business event "filter routines".

7.2.3 Products that can be copied

Exercise Directory

- Exercise Directory provides a bulk directory of various exercises that can be directly copied as there is no legal ownership of an exercise.

HyperHuman API

- HyperHuman API offers pre-built workout programs that can be used as basic templates. This is especially useful in the beginning stages of the app where user generated content is at a low.
- This addresses the business event "browse programs".

7.3 New Problems

N/A

7.4 Tasks

- Determine which technologies will be used for the creation of this project (frontend framework, server language).
- Design UX wire-frames for the application.
- Design UI design mockups for the application.
- Design UI components and testing for the application with front-end technology.
- Create skeleton of API.
- Revise requirement documents.
- Implement test report.

7.5 Migration to the New Product

The product will be deployed to the Apple App store first, then it will be added to the Google Play store once it has run without issue on the app store. This project is new, so no transition period is relevant.

7.6 Risks

- User data breach. If users' data is leaked, there would be damaging ramifications on our reputation and our userbase.
- Failure of content moderation. Olympian is a social media application and as such is susceptible to being used for nefarious and hateful purposes. This is why content moderation will be vital in running this application.

7.7 Costs

- The cost of keeping a server that is always responsive to client requests. This has an estimated cost of $25 per month.
- The cost of utilizing a database that can handle many reads and writes quickly. This has an estimated cost of $10 per month.
- The cost of an Individual Developer Account needed to host all apps on the Apple App Store. This will cost USD$99.

7.8 User Documentation and Training

A user guide will be provided so that users are able to understand the intricacies of the application and how to navigate it.

7.9 Waiting Room

- The product must allow users to track their diet information and recommend diets based on calory intake.
- The product must algorithmically produce a "Recommended" feed filled with workouts selected for users based on their habits and interests.
- The product must include an optional networking feature where users can view the growth of similar users and track the steps taken to produce those improvements.

7.10 Ideas for Solutions

The product should be built using React Native along with an Express.js backend and utilizing the Google Firestore suite for app management including database and storage. Pricing was collected with these services in mind and they fullfil the needs of the product.

8 Reference Material

This section records information for easy reference.

9 Reflection Appendix

9.1 Required Skills and Knowledge

- Skill: **Team management**
  Rationale: In order to organize deadlines, work distribution, project progression and team communication, having good team management principles will guide the team towards a positive and productive environment.
  Team member: William Lee

- Skill: **Data & Database management**
  Rationale: Many components of this project will require data tracking and storage in a secure fashion.
  Team member: William Lee

- Skill: **Mobile UI Development**
  Rationale: As a mobile application, this product will need to have a fluid user interface.
  Team member: Dimitri Tsampiras

- Skill: **Mobile Functionality Development**
  Rationale: As a mobile application, this product will require knowledge in mobile development techniques and functions.
  Team member: Matthew McCracken

- Skill: **Full Stack Development**
  Rationale: As an application utilizing a client side, a server, and an online database, this product will require knowledge of how to integrate different layers of full stack applications.
  Team member: Matthew McCracken

- Skill: **Software Communications**
  Rationale: Data distribution and transferring will be required for communication between software layers. Having knowledge towards software communicaton will be required to have efficent communication.
  Team member: Jared Bentvelsen

- Skill: **Document Organization and Writing**
  Rationale: Having descriptive and effective writing will be required to increase usability and decrease the learning curve for users. Document organization will be instrumental to carrying out project principles and guiding the growth and development of the project.
  Team member: Yuvraj Randhawa

- Skill: **Software Architecture and Structure**
  Rationale: Having a sound software structure will increase development efficiency,

simplicity and organization. With topic knowledge in this area, software management and testing will be made easy, saving time and effort for developers.
Team member: Bassel Rezkalla

9.2 Approaches for Acquiring Skills and Knowledge

- Skill: **Team management**
  Approach 1: Attend group synergy meetings where we discuss goals for the project and weekly meetings to discuss issues.
  Approach 2: Utilize Discord and GitLab's PR functionality to achieve organization and solid communication for all project deliverables.
  Verdict: William will use Approach 1 to improve his Team Management ability. He has chosen this approach because it will bring him face to face with his team and align team goals.

- Skill: **Data & Database management**
  Approach 1: Make use of the many online tutorials, articles and existing solutions for database management to refer to and learn from.
  Approach 2: Utilize notes, projects, and past tests from SFWRENG-3DB3, a class that all members of the group took, to achieve good database management practices.
  Verdict: William will use Approach 2 to improve his database management skills. He has chosen this approach because he has a vast collection of documentation from SFWRENG-3DB3 and did well in the course.

- Skill: **Mobile UI Development**
  Approach 1: Most mobile UI have well documented libraries to refer and learn from. Having design and prototyping sessions with the team before development will help improve good domain knowledge.
  Approach 2: Figma is the resource we will use to mock up the frames of our application. It also has extensive documentation and tutorials we will complete.
  Verdict: Dimitri will use Approach 2 to improve his Mobile UI Development skills. Figma is a valuable resource and familiarizing himself with it will improve his ability to provide quality mobile designs. By performing these tutorials, Dimitri will be exposed to other mobile designs to learn what works and what doesn't.

- Skill: **Mobile Functionality Development**
  Approach 1: We can explore the Touch API and its documentation to learn about the events emitted by the actions of mobile users.
  Approach 2: Practice by converting existing applications to make them mobile responsive. Our group has experience making web applications but could practice mobile development by making these apps responsive.
  Verdict: Matthew will use Approach 1 to improve his Mobile Functionality Development skills. He has chosen this strategy because the Touch API is essential for making mobile applications and he already has experience with responsive web development.

- Skill: **Full Stack Development**
  Approach 1: Complete the Google Cloud Certified - Associate Cloud Engineer Certification. This will provide skills on deploying web applications and integrating Google Cloud Services.
  Approach 2: Set up dummy clients and servers and practice hosting active client applications with simple UI.
  Verdict: Matthew will use Approach 1 to improve his Full Stack Development skills. He has chosen this strategy because this certificate offers a more extensive and professional training than Matthew can achieve on his own. Additionally, Google's certificate will introduce him to many GCP services that the project will need to utilize.

- Skill: **Software Communications**
  Approach 1: Perform Github beginner to intermediate tutorials.
  Approach 2: Take a Coursera Course on Software Specification writing.
  Verdict: Jared will use Approach 2 to improve his Software Communications skills. He has chosen this strategy because the Coursera course will show examples of correct specifications and teach him best practices which he can then pass on to the team.

- Skill: **Document Organization and Writing**
  Approach 1: Practice writing in a journal to practice descriptive and clear writing.
  Approach 2: Review notes from SFWRENG-3I03 to improve communication skills in an engineering context.
  Verdict: Yuvraj will use Approach 2 to improve his Document Organization and Writing skills. He has chosen this strategy because this course provided lessons on writing in LaTeX and how to write clear software documents.

- Skill: **Software Architecture and Structure**
  Approach 1: Referring to common software structures and architectures will help create guidelines for making a catered solution for this product. Many university courses such as McMaster's SFWRENG-2AA4 and SFWRENG-3A04 discuss topics surrounding software architecture.
  Approach 2: Hold group meeting to discuss common architecture styles and come to an agreement on how all developers will design this architecture. Synergy is important with multiple developers designing a core structure.
  Verdict: Bassel will use Approach 2 to improve his Software Architecture and Structure skills. He has chosen this strategy because it is important for all developers to be on the same page regarding software architecture strategies.