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

October 5, 2022

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

1 Project Drivers

1.1 The Purpose of the Project

1.1.1 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.

1.2 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.

2 Project Constraints

2.1 Mandated Constraints

2.1.1 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.

2.1.2 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.

2.1.3 Partner or Collaborative Applications

N/A

2.1.4 Off-the-Shelf Software

N/A

2.1.5 Anticipated Workplace Environment

N/A

2.1.6 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

2.1.7 Budget Constraints

N/A

2.1.8 Enterprise Constraints

N/A

2.2 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.

2.3 Relevant Facts and Assumptions

N/A

3 Functional Requirements

3.1 The Scope of the Work

3.1.1 The Current Situation

N/A

3.1.2 The Context of the Work

N/A

3.1.3 Work Partitioning

Event Name	Input and Output	Summary
Creating a Program	Program Creation *input*	Finalizing all program creation inputs from user
Leaving a review on a user's post	Writing a review *input*	Commenting and proving a numeric value for program value
Searching for a Program	Program Search *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 *input*	Tracking workout information by adding sets and reps performed during exercise
Registering an account	Creating a new account *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 account information *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
Providing user with statistics	Generating and displaying statistics *output*	This will show users statistics they wouldn't otherwise know about.

3.2 Business Data Model and Data Directory

N/A

3.3 The Scope of the Product

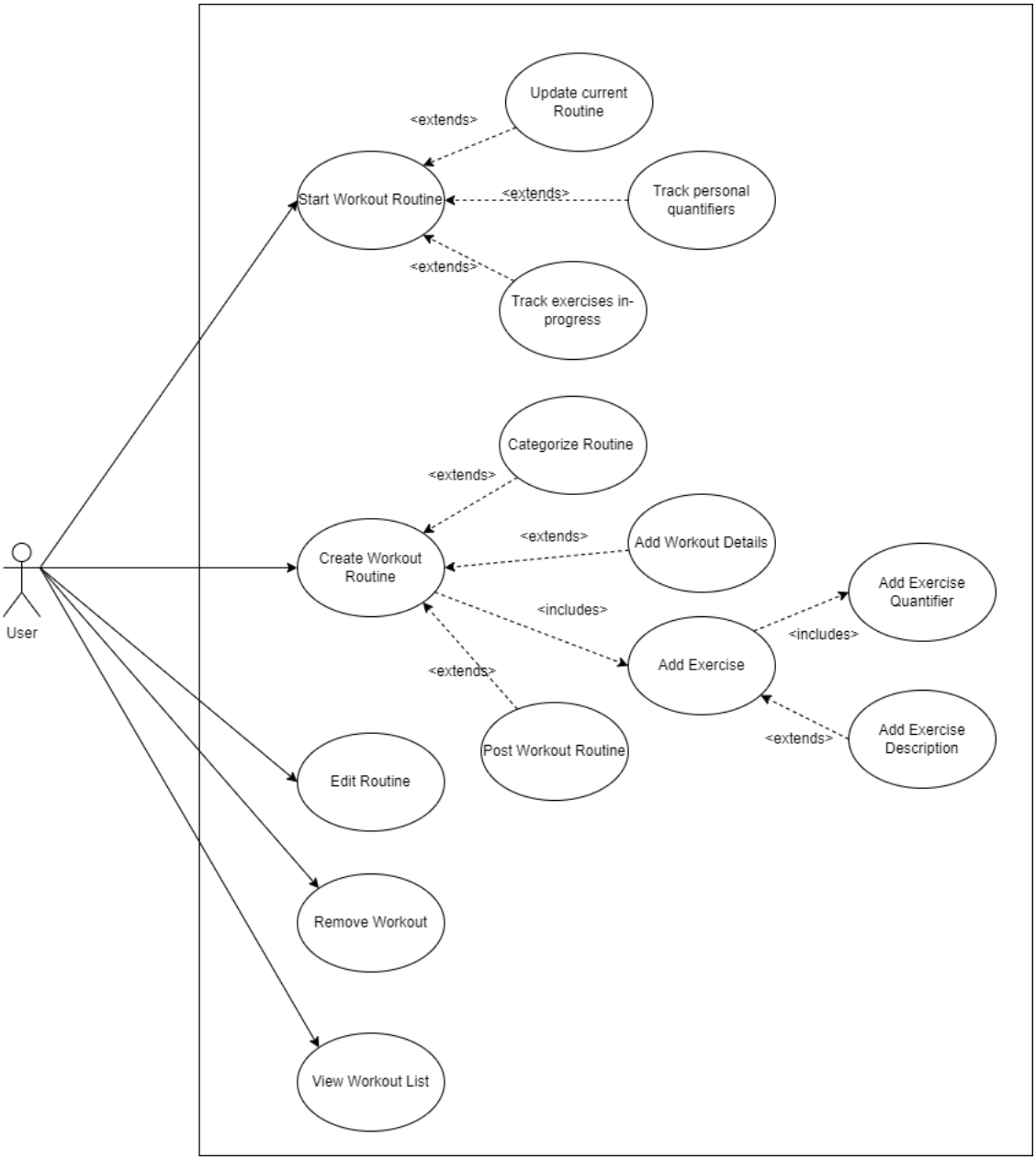
3.3.1 Product Boundary

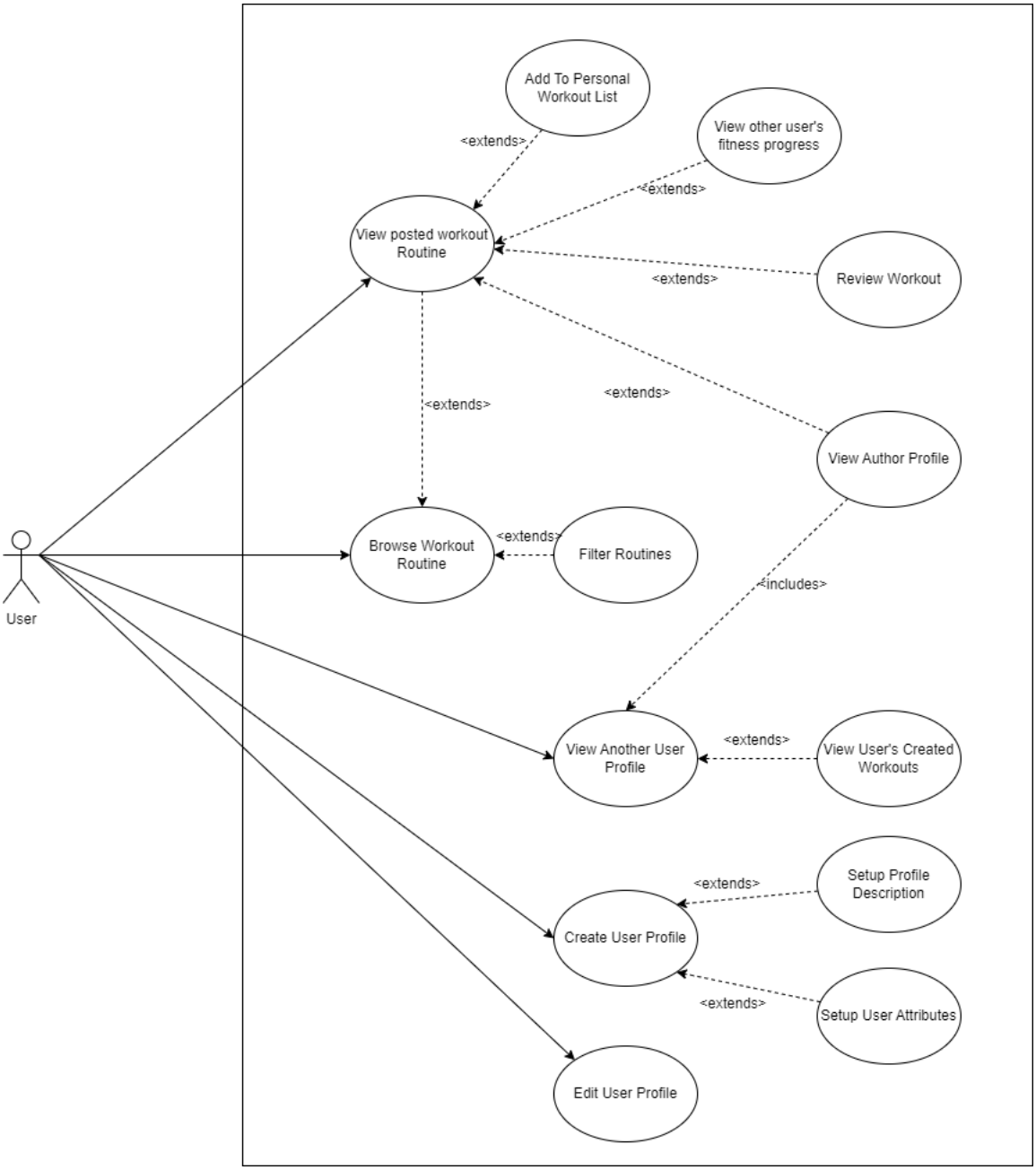
N/A

3.3.2 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

3.3.3 Use case Diagram





3.4 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.

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 one 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.

4 Nonfunctional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

- N/A

4.1.2 Style Requirements

NFR1: The product shall be appear minimal and straightforward.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

NFR2: The product shall use fonts of readable size to the target user group.

4.2.2 Personalization and Internationalization Requirements

NFR3: The product shall allow the user to select a chosen language.

4.2.3 Learning Requirements

NFR4: The product shall be able to be used by untrained fitness enthusiasts and amateurs alike, who receive no training before using it.

4.2.4 Understandability and Politeness Requirements

NFR5: The product shall use symbols and pictures to provide users with an intuitive and efficient experience.

4.2.5 Accessibility Requirements

NFR6: The product shall be usable by users with hearing loss or partial blindness.

NFR7: The product shall make use of sufficiently contrasting colours.

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

NFR8: Users shall be able to input or update a program within 10 seconds.

4.3.2 Safety-Critical Requirements

- NA

4.3.3 Precision or Accuracy Requirements

NFR9: The average of the ratings per post shall be accurate to 2 decimal places.

4.3.4 Reliability and Availability Requirements

NFR10: The application shall achieve a 95 percent uptime.

4.3.5 Robustness or Fault-Tolerance Requirements

- N/A

4.3.6 Capacity Requirements

NFR11: The application shall support 100 simultaneous users per hour.

4.3.7 Scalability or Extensibility Requirements

NFR12: The application should be able to support 1000 users per hour within a year of its creation.

4.3.8 Longevity Requirements

- N/A

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

- N/A

4.4.2 Requirements for Interfacing with Adjacent Systems

NFR13: The application shall operate on iOS devices and Android devices.

4.4.3 Productization Requirements

- N/A

4.4.4 Release Requirements

- N/A

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

NFR14: The application must inform users when maintenance is taking place and must warn them at least 1 day in advance.

4.5.2 Supportability Requirements

- N/A

4.5.3 Adaptability Requirements

- N/A

4.6 Security Requirements

4.6.1 Access Requirements

NFR15: The application must not display other users private details to the user.

4.6.2 Integrity Requirements

NFR16: Passwords must be encrypted with SHA-256 when stored.

4.6.3 Privacy Requirements

NFR17: The application must use OAuth protocols to verify communication between the client and server.

4.6.4 Audit Requirements

NFR18: Data will be stored in a secure database. When data is deleted or edited a record of this data will be kept for up to 30 days.

4.6.5 Immunity Requirements

- N/A

4.7 Cultural Requirements

4.7.1 Cultural Market Requirements

NFR19: The application will filter out profanities and flag repeat offenders for suspension.

NFR20: The application will allow users to report offensive content and remove it from their feed.

4.7.2 Cultural Diversity and Inclusion Requirements

NFR21: Users can optionally specify their gender, age, and race. Gender and age will be used to cater the content for individuals feeds.

4.8 Legal Requirements

4.8.1 Legal Compliance Requirements

NFR22: The use of data will comply with the Data Protection Act.

4.8.2 Standards Compliance Requirements

- N/A

5 Traceability Matrices and Graphs

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11
NFR4	X										
NFR8	X	X	X								
NFR9							X				
NFR10							X	X			
NFR11							X	X			
NFR12							X	X			
NFR13	X					X	X				
NFR18	X				X		X		X	X	X
NFR20						X		X			
NFR22	X				X		X		X	X	X

Table 1: NonFunctional to Functional Requirement Dependency Matrix

6 Project Issues

6.1 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.

6.2 Off the Shelf Solutions

6.2.1 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.

6.2.2 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".

6.2.3 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".

6.3 New Problems

Not applicable.

6.4 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.

6.5 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.

6.6 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.

6.7 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.

6.8 User Documentation and Training

N/A

6.9 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.

6.10 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.

7 Reference Material

This section records information for easy reference.

7.1 Abbreviations and Acronyms

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
T	Theoretical Model

8 Reflection Appendix

8.1 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

8.2 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.