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
Date 1	1.0	Notes
Date 2	1.1	Notes

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

N/A

3.2 Business Data Model and Data Directory

N/A

3.3 The Scope of the Product

3.3.1 Product Boundary

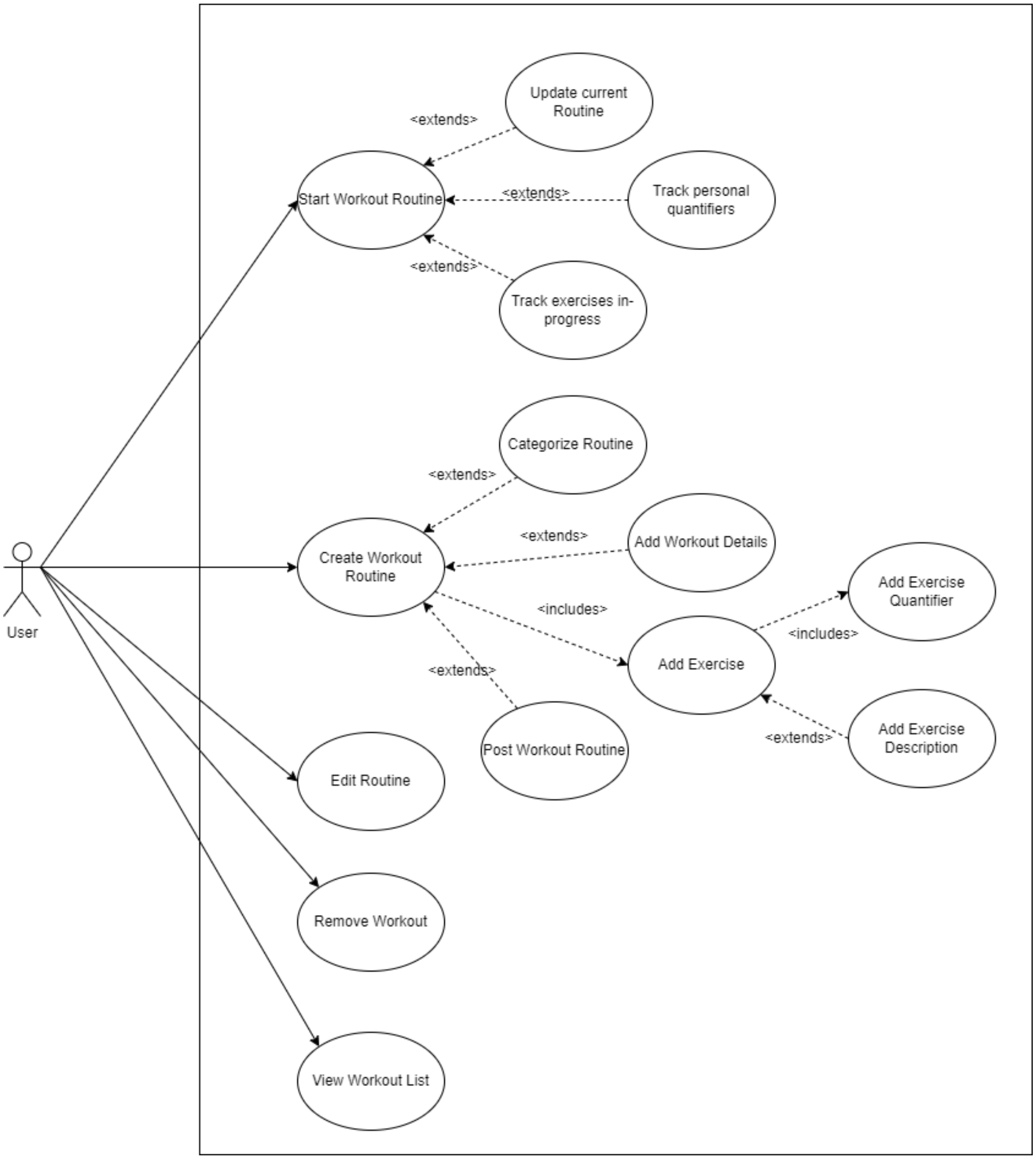
N/A

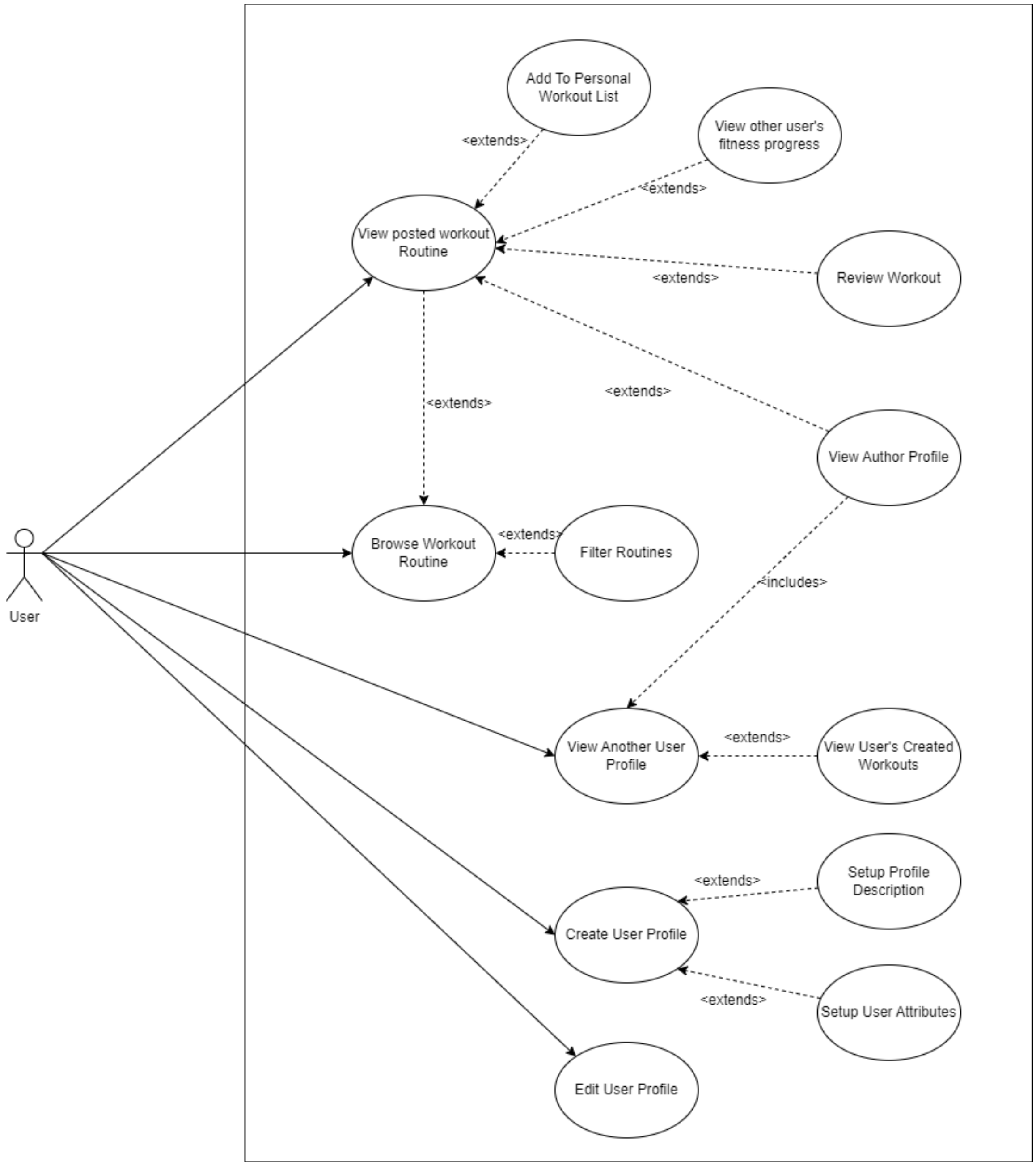
3.3.2 Use cases

- 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 add a "hint" section to a given exercise. A "hint" is strictly text less than 500 characters.
Rationale: The user should be able to add hints, tips, tricks, and finer workout descriptions to a given exercise.
Fit Criterion: The user is able to attach a hint to each exercise in a workout routine.

R5: Description: The system shall store a workout routine created by a user.
Rationale: To be able to store a routine to potentially share the routine with other users.
Fit Criterion: On creation of a workout routine, the data for the routine should be accessible by the system.

R6: 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.

R7: Description: The product shall allow a user to update a workout routine with the same capabilities granted during workout routine
Rationale: For users to be able to change quantifiers or routine properties after the creation time. Users may change their mind or make mistakes, so making updates to existing routines is an important requirement.
Fit Criterion: The user is able to update pre-existing workout routines.

R8: 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.

R9: 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.

R10: Description: The product should allow a user to save another user's public workout routine, such that the other user's routine is accessible from the saving user's profile.
Rationale: To allow a user to save a workout routine for later use.
Fit Criterion: The user is able to save another user's workout routine to their profile.

R11: 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.

R12: Description: The system shall allow a user to edit their profile.
Rationale: To update any information regarding the user profile.
Fit Criterion: The user is able to make changes to their profile after it has been created.

R13: 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.

R14: 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.

R15: Description: The system shall allow the user to create progress points towards a specific goal.
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.

R16: Description: 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: Progress points must be simple to enter so that the resultant data can be easily visualized.
Fit Criterion: Progress points can only exist as a date-metric pair with the metric type matching the associated goal metric type.

R17: 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.

[Every IM should map to at least one requirement, but not every requirement has to map to a corresponding IM. —TPLT]

4 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —TPLT] [The goal is for the nonfunctional requirements to be unambiguous, abstract and verifiable. This isn't easy to show succinctly, so a good strategy may be to give a "high level" view of the requirement, but allow for the details to be covered in the Verification and Validation document. —TPLT] [An absolute requirement on a quality of the system is rarely needed. For instance, an accuracy of 0.0101 % is likely fine, even if the requirement is for 0.01 % accuracy. Therefore, the emphasis will often be more on describing now well the quality is achieved, through experimentation, and possibly theory, rather than meeting some bar that was defined a priori. —TPLT] [You do not need an entry for correctness in your NFRs. The purpose of the SRS is to record the requirements that need to be satisfied for correctness. Any statement of correctness would just be redundant. Rather than discuss correctness, you can characterize how far away from the correct (true) solution you are allowed to be. This is discussed under accuracy. —TPLT]

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

NFR1: The product shall be attractive to a young audience (teenagers, adults in their 20s and 30s).

4.1.2 Style Requirements

NFR2: The product shall be appear minimal and straightforward.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use Requirements

NFR3: The product shall be easy for teenagers to use.

NFR4: The product shall be unintrusive and nondisruptive to use while exercising.

4.2.2 Personalization and Internationalization Requirements

NFR5: The product shall allow the user to select a chosen language.

NFR6: The product shall retain the user's selected exercise preferences.

4.2.3 Learning Requirements

NFR7: 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

NFR8: The product shall use symbols and pictures to provide users with an intuitive and efficient experience.

4.2.5 Accessibility Requirements

NFR9: The product shall be usable by users with hearing loss or partial blindness.

NFR10: The product shall make use of sufficiently contrasting colours.

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

NFR11: Search results shall be displayed within 10 seconds.

NFR12: User login shall support 100 users per hour providing a response time of less than 10 seconds.

NFR13: Application pages shall support 100 users providing a response time of less than 10 seconds.

NFR14: 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

NFR15: The average of the ratings per post shall be accurate to 2 decimal places.

4.3.4 Reliability and Availability Requirements

NFR16: The application shall achieve a 95 percent uptime.

4.3.5 Robustness or Fault-Tolerance Requirements

NFR17: The application shall operate offline and online.

4.3.6 Capacity Requirements

NFR18: The application shall support 100 simultaneous users per hour.

4.3.7 Scalability or Extensibility Requirements

NFR19: The application should be able to support 1000 users per hour within a year of its creation.

NFR20: The application should not lose any functionality or efficiency as it grows.

4.3.8 Longevity Requirements

NFR21: The application should be able to operate without maintenance for a minimum of six months.

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

NFR22: The application shall operate on a computer and phone.

4.4.2 Requirements for Interfacing with Adjacent Systems

NFR23: The application shall operate on iOS devices, Android devices, and web browsers.

4.4.3 Productization Requirements

NFR24: The application shall be distributed on the corresponding iOS and Android stores, as well as on the web.

4.4.4 Release Requirements

NFR25: The application will have maintenance releases every six months.

NFR26: The maintenance releases will improve features and cause no previous features to fail.

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

NFR27: The application must inform users when maintenance is taking place and must warn them at least 1 day in advance.

NFR28: The application must display informative error and warning messages when problems occur.

4.5.2 Supportability Requirements

- N/A

4.5.3 Adaptability Requirements

NFR29: The application must run on mobile devices. It will be supported on Android and IOS.

4.6 Security Requirements

4.6.1 Access Requirements

NFR30: The application must allow users to access the application and their data using private credentials

NFR31: The application must not display other users private details to the user.

NFR32: The application must allow users to securely reset their password when they forget it.

4.6.2 Integrity Requirements

NFR33: The application must store data in a secure database.

NFR34: The application must verify the users email address when they first sign up.

NFR35: Passwords must be at least 8 characters, with an uppercase letter, a number, and a symbol.

NFR36: Passwords must be encrypted when stored.

4.6.3 Privacy Requirements

NFR37: The application must offer the user the option to keep all of their data private from other users.

NFR38: The application must use OAuth protocols to verify communication between the client and server.

4.6.4 Audit Requirements

NFR39: 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

NFR40: The application will filter out profanities and flag repeat offenders for suspension.

NFR41: The application will allow users to report offensive content and remove it from their feed.

4.7.2 Cultural Diversity and Inclusion Requirements

NFR42: 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

NFR43: The use of data will comply with the Data Protection Act.

4.8.2 Standards Compliance Requirements

- N/A

NFR44: **Accuracy** [Characterize the accuracy by giving the context/use for the software. Maybe something like, "The accuracy of the computed solutions should meet the level needed for <engineering or scientific application>. The level of accuracy achieved by Software Eng 4G06 shall be described following the procedure given in Section X of the Verification and Validation Plan." A link to the VnV plan would be a nice extra. —TPLT]

NFR45: **Usability** [Characterize the usability by giving the context/use for the software. You should likely reference the user characteristics section. The level of usability achieved by the software shall be described following the procedure given in Section X of the Verification and Validation Plan. A link to the VnV plan would be a nice extra. —TPLT]

NFR46: **Maintainability** [The effort required to make any of the likely changes listed for Software Eng 4G06 should be less than FRACTION of the original development time. FRACTION is then a symbolic constant that can be defined at the end of the report. —TPLT]

NFR47: **Portability** [This NFR is easier to write than the others. The systems that Software Eng 4G06 should run on should be listed here. When possible the specific versions of the potential operating environments should be given. To make the NFR verifiable a statement could be made that the tests from a given section of the VnV plan can be successfully run on all of the possible operating environments. —TPLT]

- Other NFRs that might be discussed include verifiability, understandability and reusability.

5 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 1 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 2 shows the dependencies of instance models, requirements, and data constraints on each other. Table 3 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

	T??	T??	T??	GD??	GD??	DD??	DD??	DD??	DD??	IM??	IM??	IM??	IM
T??													
T??			X										
T??													
GD??													
GD??	X												
DD??				X									
DD??				X									
DD??													
DD??								X					
IM??					X	X	X				X		
IM??					X		X		X	X			X
IM??		X											
IM??		X	X				X	X	X		X		

Table 1: Traceability Matrix Showing the Connections Between Items of Different Sections

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

	IM??	IM??	IM??	IM??	??	R??	R??
IM??		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R??	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R??			X	X			
R??		X					
R??		X					

Table 2: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
T??	X																		
T??																			
T??																			
GD??		X																	
GD??			X	X	X	X													
DD??							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM??											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 3: Traceability Matrix Showing the Connections Between Assumptions and Other Items

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.
- **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, 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

- 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. 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

- App that lets you create and track workouts. No user generated content but has paid template programs. Social aspect that lets you post progress pictures.
- 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 one rep max.
- 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

- 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)
- app UX wire-frames
- app UI design mockups
- app UI components and testing using front end technology
- Create skeleton of API
- Revise requirement documents.
- Implement test report

6.5 Migration to the New Product

The product will be rolled out to the Apple App store first and then added to the Google play store after it has run without issue on the app store. This project is new and so no transition period is relevant.

6.6 Risks

- User data breach. If users data is leaked then this would have damaging ramifications for 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 expressjs 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
Software Eng 4G06	[put an expanded version of your program name here (as appropriate) —TPLT]
T	Theoretical Model

[Add any other abbreviations or acronyms that you add —TPLT]

References

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]
[Grammar, flow and LaTeXadvice:

- For Mac users `*.DS_Store` should be in `.gitignore`
- LaTeX and formatting rules
  - Variables are italic, everything else not, includes subscripts (link to document)
    - Conventions
    - Watch out for implied multiplication
  - Use BibTeX
  - Use cross-referencing
- Grammar and writing rules
  - Acronyms expanded on first usage (not just in table of acronyms)
  - "In order to" should be "to"

—TPLT]
[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be "not applicable" for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- "Referenced by" implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

2. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?