LiftingPal

Software Requirements Specification

1.0

10/8/2018

Andrew McLaren, Angel Chang, Drew Kozak, Robert Rozin, Abhi Inuganti
Software Engineer

Prepared for
The Best 1530 Group

Revision History

Date	Description	Author	Comments
10/7/18	<Version 1>	Angel Chang, Drew Kozak, Robert Rozin, Andrew McLaren, Abhi Inuganti	Requirements, Use Case, Diagrams

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

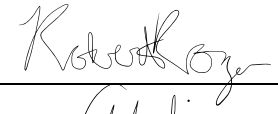
Signature	Printed Name	Title	Date
	Andrew McLaren	Software Eng.	10/8/2018
	Robert Rozin	Software Eng.	10/8/2018
	Abhi Inuganti	Software Eng.	10/8/2018
	Angel Chang	Software Eng.	10/8/2018
	Drew Kozak	Software Eng.	10/8/2018

Table of Contents

Revision History..... ii

Document Approval..... ii

1. Introduction..... 5

1.1 Purpose..... 5
1.2 Scope.....5
1.3 Definitions, Acronyms, and Abbreviations..... 6

2. General Description..... 6

2.1 Product Perspective..... 6
2.2 Product Functions..... 7
2.3 Constraints, Assumptions and Dependencies..... 7

3. Specific Requirements..... 7

3.1 External Interface Requirements..... 7
3.1.1 Software Interfaces..... 7
3.2 Functional Requirements..... 7
3.2.1 Input of 1 Rep Max..... 7
3.2.2 Desired Program Type..... 7
3.2.3 Equipment..... 7
3.2.4 Weaknesses..... 7
3.2.5 Injuries..... 7
3.2.6 Follow Up Questionnaire..... 7
3.2.7 LB to Kg converter..... 7
3.2.8 Macronutrients Calculator..... 8
3.2.9 Social Media..... 8
3.2.10 Block Length..... 8
3.2.11 Experience Level..... 8
3.2.12 Raw vs. Equipped Lifting..... 8
3.2.13 Peaking Cycle..... 8
3.3 Use Cases..... 8
3.3.1.1 ORMInput and Units Used..... 9
3.3.1.2 InputLifterLevel..... 10
3.3.1.3 InputProgramType..... 11
3.3.1.4 Input of Limitations/Injuries..... 12
3.3.1.5 Weakness Collection for Compound Lifts..... 13
3.3.1.6 Input Equipped or Raw.....15
3.3.1.7 Input Desired Lifting Frequency..... 16
3.3.1.8 Input Extra Possible Equipment..... 17
3.3.2.1 Input Weight on Macronutrient Calculator..... 18
3.3.2.2 Input Age on Macronutrient Calculator..... 19
3.3.3.1 Social Media Page..... 20

3.4 Classes / Objects..... 21

3.5 Non-Functional Requirements..... 23
3.5.1 Dictionary of Terms Used/Explanations..... 23
3.5.2 Memorization of User Input..... 23
3.5.3 Categorization of Users..... 23
3.5.4 Documentation Behind Lifter Types..... 24
3.5.5 Time Taken to Generate Program..... 24
3.5.6 Security.....24
3.5.7 Oracle Database..... 24
3.5.8 System Checks/Updates..... 25
3.6 Logical Database Requirements..... 25
3.7 Other Requirements..... 25

4. Analysis Models..... 25

4.1 Use Case Diagrams..... 25
4.2 Class Diagrams..... 27
4.3 Sequence Diagrams..... 28
4.4 Activity Diagrams (DFD)..... 29
4.5 State-Transition Diagrams (STD)..... 31

5. Change Management Process..... 35

A. Appendices..... 36

1. Introduction

This webapp is to help generate workout blocks for users who want to either get into a lifting routine, or for experienced lifters who want an workout block generator and system to keep track of their progress.

1.1 Purpose

To create tailored workout plans for beginner to advanced lifters who are looking for a place to track their progression, share their results with friends, and calculate their macro nutritional needs.
<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

LiftingPal is a web application that allows for new users to get an introduction into weight lifting and then continue their progression as they get acclimated to weight lifting. LiftingPal uses a questionnaire to help create workout blocks of the user's preferred length. It takes in their injuries, weaknesses, equipment available, and frequency of workouts to figure out a good lifting plan to follow. LiftingPal is accompanied with a macronutrient calculator to help track their physical health and make sure the user is getting the proper nutrients. The web app uses a database to store each user's information and track their progression, so that when they reach the follow up questionnaire at the end of the block, they can continue their physical development. This is a type of product that results in robust workout plans and can get the user into top physical strength. LiftingPal also implements a social media aspect to encourage their progression and show case to their friends when they hit a new Personal Record (PR) max.

1.2 Scope

This subsection should:

(1) LiftingPal

(2) What the software product will do and won't do (just high level, no details):

This product will collect information from a workout enthusiast regarding their weaknesses, injuries, and equipment. After collecting this information the product will produce a workout plan specifically designed for the workout enthusiast. This product will not force the user to follow through on any of the workout plans designed for them. The choice of whether or not to follow through on any of the plans will be up to the user to decide.

(3) Describe the application of the software being specified. Describe all relevant benefits, objectives, and goals as precisely as possible:

The main benefit of this product is to allow the user to live a healthy and productive life. This product will set objectives and specific goals in terms of the workouts that are assigned to

the user. The user should be able to follow the workout plans that were assigned to them, and fulfill each specific goal.

1.3 Definitions, Acronyms, and Abbreviations

Rep(s)- repetition(s)

2. General Description

The user affects the product and its requirements and is the primary source of what can affect the aforementioned items. Storing the user's data is another key factor in the creation of LiftingPal, if some bug happens to the Oracle Database holding the user's information, it may cause problems retrieving data and displaying correct results. A breach in security is another aspect that may cause future problems.

2.1 Product Perspective

There are other competing applications for Apple and Android that generate or suggest workouts for users like Fitness Point, Fitted Lifts and Jefit. These applications help make suggestions to how users should workout, while LiftingPal generates workout blocks for users and help them schedule when and how to workout. Our product goes beyond simply instructing users on how to do workouts, it helps generate workouts for the users needs while helping them stay on track by fitting into their schedule and avoiding any skips because of injurys or work obligations.

2.2 Product Functions

This product will perform functions that are related to the main questionnaire with regards to the creation of a workout plan. There will be functions that handle user input with regards to their favored workout programs, their specific level in general lifting, injuries, and weaknesses in terms of workouts. All of these specific functions deal with the user's workout plan, and furthermore there will be more functions that that the software will perform for the user's social media page. The product should post the user's accomplishments on their social media page upon successful completion of a workout.

2.3 Constraints, Assumptions and Dependencies

The project assumes the user will be putting in logistical numbers relevant to how much weight the user can actually lift. There is no true benefit for the user to input values that they cannot physically achieve while working out. There are various problems that may occur within the Oracle database. The Oracle database may go through a bug or error that does not update all of the data at some point, there may be a malfunction storing the data due to any bugs that may occur when Oracle has updates to their servers. The macronutrient calculator is an external software that is depending on the user to put reasonable height, weight, gender, and age and

the percentages they are looking to adjust their caloric intake that will ultimately increase their physical health. If there is a compromise in security of the Oracle database, it will compromise the information LiftingPal stores and can change the final result of workout blocks or other prior user information.

3. Specific Requirements

This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 3.2. Overall, this section 3 will capture the requirements that are used to guide the project's software design, implementation, and testing.

3.1 External Interface Requirements

3.1.1 Software Interfaces: Oracle Database, Macronutrient calculator, Social Media interface.

3.2 Functional Requirements

This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.

3.2.1 Input of squat/bench/deadlift 1 rep maxes

Initial input of current max reps

3.2.2 Desired Program Type

Upper/Lower split
Full body
Hits a single muscle group a day
Push, Pull, Legs

3.2.3 Equipment

Any extra equipment the user might have
Ex: bench, types of bar, weights…

3.2.4 Weaknesses

What the user is weak at and would like to improve on

3.2.5 Injury checker

Any injuries that could restrict them from certain exercises

3.2.6 Follow up questionnaire

After each workout block finishes, fill out another questionnaire to update progress and capabilities

3.2.7 Lb to Kg converter

Converter for users who prefer one over the other

3.2.8 Macronutrients Calculator

Determines necessary caloric intake and macronutrient intake depending on user input and the user's age, weight, height, and sex.

3.2.9 Social Media

Will share with friends when a workout is completed or a personal record is reached

3.2.10 Desired Block Length

How long does the user want their block to be

3.2.11 Experience level

Beginner, Intermediate, Advanced Lifter selection

3.2.12 Raw vs. Equipped Lifting

Whether an advanced lifter will be using a squat suit with knee wraps, a bench shirt, and a deadlift suit and would like to improve their lifts with them.

3.2.13 Peaking Cycle

Creates a block that will shed fatigue and prepare the lifter to test their maxes in the best physical form possible

3.3 Use Cases

Use Case ID:	1.1
Use Case Name:	ORMInput and Units Used
Created By:	Team
Last Updated By:	Team
Dated Created:	September 13
Last Revision Date:	September 17
Actors:	Workout Enthusiast
Description:	Collect information on actor and base program percentages off the user input.
Trigger:	User starts to fill out the questionnaire.

Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab.
Postconditions:	1. Stores user input. 2. Question fades. 3. User case 1.2 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	5a. User enters number larger than 9999Lb or 9999Kg 1. Next button disabled, dialog box saying “Invalid input: must be less than or equal to 9999” 2. User changes number 3. User continues normal flow
Includes:	Register new User Sign in Questionnaire button pressed
Frequency of Use:	When the user first logs in, and then any time the user would like to change them (due to time off, or testing new maxes)
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.2
Use Case Name:	InputLifterLevel
Created By:	Team
Last Updated By:	Team

Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base program blocks and layout on lifter level
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. Completes user case 1.1
Postconditions:	1. Stores user input. 2. Question fades. 3. User case 1.3 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (Must select one of three levels: Beginner, Intermediate, or Advanced, to continue)
Includes:	Register new User Sign in Questionnaire button pressed One rep max and unit metric inputs Lifter experience level input
Frequency of Use:	When the user first logs in, and then any time the user would like to change it (due to progressing as a lifter)
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.3
Use Case Name:	InputProgramType
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base program percentages off the user input.
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. Complete use case 1.2
Postconditions:	1. Stores user input. 2. Question fades. 3. Use case 1.4 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1-2.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (Must choose a program type)
Includes:	Register new User Sign in Questionnaire button pressed One rep max inputs Lifter experience level input
Frequency of Use:	When the user first logs in, and then any time the user would like to change it.

Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.4
Use Case Name:	Input of Limitations/Injuries
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base program percentages off the user input.
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab 4. User completes use case 1.3
Postconditions:	1. Stores user input 2. Question fades. 3. User case 1.5 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1-3.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 4
Exceptions:	N/A (give them checkboxes)

Includes:	Register new User Sign in Questionnaire button pressed One rep max inputs Lifter experience level input Program Input
Frequency of Use:	When the user first logs in or if they want to change
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.5
Use Case Name:	Weakness Collection for Compound Lifts
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base programmed lifts off of user's weak points
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. User completes 1.1-1.4
Postconditions:	1. Stores user input. 2. Question fades. 3. User case 1.6 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time.

	4. User clicks on questionnaire button. 5. User fills out questionnaire questions 1 - 4.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (User must select a button to proceed)
Includes:	Register new User Sign in Questionnaire button pressed
Frequency of Use:	When the user first logs in, and then any time the user would like to change them (due to time off, or testing new maxes)
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.6
Use Case Name:	Input Equipped or Raw
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base programmed lifts
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. User completes 1.1-1.5
Postconditions:	1. Stores user input.

	2. Question fades. 3. User case 1.7 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1-5.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (User must select one to continue)
Includes:	Register new User Sign in Questionnaire button pressed One rep max and unit metric inputs Lifter experience level input Program type input Limitations/injury input Weakness input
Frequency of Use:	When the user first logs in, and then any time the user would like to change them.
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.7
Use Case Name:	Input Desired Lifting Frequency
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast

Description:	Collect information on actor and base programmed lifts
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. User completes 1.1-1.6
Postconditions:	1. Stores user input. 2. Question fades. 3. User case 1.8 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1-5.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (User must select one to continue)
Includes:	Register new User Sign in Questionnaire button pressed One rep max and unit metric inputs Lifter experience level input Program type input Limitations/injury input Weakness input
Frequency of Use:	When the user first logs in, and then any time the user would like to change them.
Special Requirements:	Requires connection to database
Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	1.8
Use Case Name:	Input Extra Possible Equipment

Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect information on actor and base programmed lifts
Trigger:	User starts to fill out the questionnaire.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on questionnaire tab. 4. User completes 1.1-1.7
Postconditions:	1. Stores user input. 2. Question fades. 3. User case 2.1 begins
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on questionnaire button. 5. User fills out questionnaire question 1-5.
Alternative Flows:	4a. While the user is in the questionnaire, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A (User must select one to continue)
Includes:	Register new User Sign in Questionnaire button pressed One rep max and unit metric inputs Lifter experience level input Program type input Limitations/injury input Weakness input
Frequency of Use:	When the user first logs in, and then any time the user would like to change them.
Special Requirements:	Requires connection to database

Assumptions:	They know answers to the question
Notes and Issues:	N/A

Use Case ID:	2.1
Use Case Name:	Input Weight on Macronutrient Calculator
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiast
Description:	Collect weight information on actor in order to calculate macronutrient recommendations.
Trigger:	User starts to fill out the calculator.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on on macronutrient calculator tab.
Postconditions:	1. Stores user input.
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on macronutrient calculator button. 5. User fills out first input box asking for weight.
Alternative Flows:	4a. While the user is in the calculator, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A
Includes:	Register new User Sign in
Frequency of Use:	Whenever the user wants to calculator their macronutrient recommendations.

Special Requirements:	Requires connection to database to store information.
Assumptions:	They know the answer to the question.
Notes and Issues:	N/A

Use Case ID:	2.2
Use Case Name:	Input Age on Macronutrient Calculator
Created By:	Team
Last Updated By:	Team
Dated Created:	October 7
Last Revision Date:	October 7
Actors:	Workout Enthusiast
Description:	Collect age information on actor in order to calculate macronutrient recommendations.
Trigger:	User starts to fill out the calculator.
Preconditions:	1. User registers for an account. 2. User logs in account for the first time. 3. User clicks on on macronutrient calculator tab.
Postconditions:	1. Stores user input.
Normal Flow:	1. Use case begins when user accesses the webapp 2. User registers for an account. 3. User logs in account for the first time. 4. User clicks on macronutrient calculator button. 5. User fills out input box asking for age.
Alternative Flows:	4a. While the user is in the calculator, they leave 1.App closes until user returns 2.When user returns continue normal flow from step 3
Exceptions:	N/A
Includes:	Register new User Sign in

Frequency of Use:	Whenever the user wants to calculator their macronutrient recommendations.
Special Requirements:	Requires connection to database to store information.
Assumptions:	They know the answer to the question.
Notes and Issues:	N/A

Use Case ID:	3.1
Use Case Name:	Social Media Page
Created By:	Team
Last Updated By:	Team
Dated Created:	September 18
Last Revision Date:	September 18
Actors:	Workout Enthusiasts
Description:	Shares workout accomplishments with friends
Trigger:	User clicks on the Social media tab/button
Preconditions:	1. User logs into account
Postconditions:	1. User goes back to main page
Normal Flow:	1. User Logs in 2. User accesses the social media page
Alternative Flows:	N/A
Exceptions:	2a. User tries to access page but has no connection 1a. User waits to reconnect 1b. User leaves app to reconnect
Includes:	Register new User Sign in
Frequency of Use:	Whenever the User wants to see his friend feed
Special	Requires connection to database

Requirements:	
Assumptions:	They have friends
Notes and Issues:	N/A

3.4 Classes / Objects

Class List

1. User (Lifter)
2. Program Maker
    a. Beginner Program Maker
    b. Intermediate Program Maker
    c. Unequipped Advanced
    d. Equipped Advanced
3. Questionnaire
4. Social Media
    a. Post
        i. PRs
        ii. Finishing a session
        iii. Finishing a block
    b. Add Friend (Function)
5. Helpful Info
    a. Dictionary
    b. Helpful Information
6. Macro Calculator

Class Details

1. User
    a. Stores
        i. Blocks Created
        ii. Answers to Questionnaire
        iii. Friend list
    b. Methods
        i. Retrieve current block
        ii. React to Post in feed
        iii. Add/Remove Friend
        iv. Getters
            1. Blocks Created
            2. Answers to Questionnaire
            3. Friend List

        v. Setters
            1. Blocks Created
            2. Answers to Questionnaire
            3. Friend List
2. Program Maker
    a. Stores
        i. Nothing
    b. Methods
        i. Create Block
3. Questionnaire Class
    a. Stores
        i. Questions
    b. Methods
        i. Retrieve Questions
        ii. Save Answers to User Class
4. Social Media Class
    a. Stores
        i. Posts made every week (Sunday -> Saturday)
        ii. Date Post was made
    b. Methods
        i. Make new Post
5. Post Class
    a. `Stores
        i. Data within post
        ii. Reactions to post
    b. Methods
        i. Like a post
        ii. Clear Posts
6. Helpful Information Class
    a. Stores
        i. Dictionary Of Terms
        ii. Links to helpful videos and websites
    b. Methods
        i. Getters
7. Macro Calculator Class
    a. Stores
        i. Macros
            1. Fats
            2. Carbohydrates
            3. Protein
        ii. TDEE
    b. Methods
        i. Getter

1. Macros
    a. Fats
    b. Carbohydrates
    c. Protein
2. Calories

ii. Setter

1. Macros
    a. Fats
    b. Carbohydrates
    c. Protein
2. TDEE

3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

3.5.1 Dictionary for Terms Used/Explanations:

The product will return a dictionary that has explanations regarding the specific terms related to workouts for users who are not aware of these respective terms. The dictionary with all of the relevant terms/explanations should be returned to the user under five seconds. This feature is something that should be entirely reliable for the user. The user should have readily available access to this feature. The security related to this feature is guaranteed via the same security that the product itself. This dictionary will automatically be maintained and have all the necessary terms preloaded. This feature will be portable as well in accordance to the portability of the product itself.

3.5.2 Memorization of User Input: This is an integral part of the product which involves the memorization of user input to perform specific tasks related to workout plans, or other particular features. This is a task that should be able to be processed under five seconds in a quick and efficient manner. This is a feature that is reliable as it is integral for the product to function. This feature will become available whenever there is an instance of user input. Security is essentially guaranteed via the product itself. This requirement will be maintained at every instance of user input as it is important for the product to function. This feature will support portability as it is a feature of the product which itself is portable.

3.5.3 Categorization of Users: This product is dependent on the lifting ability and general strength of the users themselves. At the very minimum the user must be able to physically move a barbell. The product will maintain this general information about the user to determine whether or not they are physically able to use this product for their workouts. The product will be able to maintain and process this data under five seconds. As this is another important feature for the product and user safety it will be reliable at all times. This is a feature that will be available

initially as it will be required for the user to enter this information. Once the information has been processed, and the user is acceptable then the product can move forward. This feature will remain secure as it has access to the security of the product itself. This feature will be maintained whenever this feature is in use. This feature has access to portability by extension of the main product's portability.

3.5.4 Documentation Behind Lifter Types: There will be extensive documentation behind each lifter type. This is a feature that will be fast and will be processed under five seconds. This feature will remain available when it is in use. It is a feature that will have security access by extension of the product itself. There will be no further security processes for this feature of the product. This feature will be maintained during its usage within the product. Portability is available by extension of the main product's portability.

3.5.5 Time taken to generate Program: This non-functional feature is about the total amount of time that will be required to generate this program. The general time to generate this program will be under five seconds. This performance will be reliable and replicable unless there is a specific error within the program itself. This will be an available feature as it will be encountered during every execution and build of the program. There will be no further implementation of security for this case. There will be no maintainability for this case unless the product spends more than the regular amount of time to generate the program. This is portability as the product is meant to be a portable product.

3.5.6 Security of the product: This is the non-functional feature that is related to the security of the product itself. This means that the product will implement a level of security to make sure that user data such as names, weights, and types of workouts, and other records will be safe. The security itself will be guaranteed via user accounts that the user must create to access their own data. The performance of the security will be under five seconds, and will stay a reliable feature of the product as securing user data is very important. This feature will be available by default once the user has logged into the product. The user's account will be the security which holds the data of each respective user. Security will be maintained during each run of the program. This will be a portable feature as security must be maintained via logins regardless of platform.

3.5.7 Oracle Database: Oracle Database will be utilized in this product to store data with regards to the user. This feature must remain fast and efficient and should be able to be processed under five seconds. This is a reliable process as a it is a third-party application that is known to be trustworthy and reliable. Security is already granted via the security of the product itself (user login). There will be no further security implementations for the database aside from the user's login. There will be no reason to maintain this feature. It will be used when the need specifically arises during the run of the program. This will be a portable feature as we must have a database to run integral parts of the program regardless of the platform.

3.5.8 System Checks/Updates: The system must check and update the server every time it is opened, and the device is connected to the internet. The requirement's performance will be processed in under five seconds. This will be a reliable feature as the system must update every time is opened unless there is some error within the process. This feature will be available whenever the device itself is connected to the internet, and whenever the system itself is opened. There will be no further implementation of security for this feature aside from the user's login. This process will be maintained every time this feature is in use. This will be a portable feature as it regardless of the platform there must be a constant update whenever the program is opened, or whenever the device itself is connected to the internet.

3.6 Logical Database Requirements

We will run a MySQL Relational Database that will store information on users that will be used to generate their workout block. We will keep the tables to a minimum, we will have three tables. The first will store usernames, passwords and friends, the second will contain the equipment that each user has access to, and the third will contain Personal Records, their current maxes for different workouts and benchmarks they reached in pervious block.

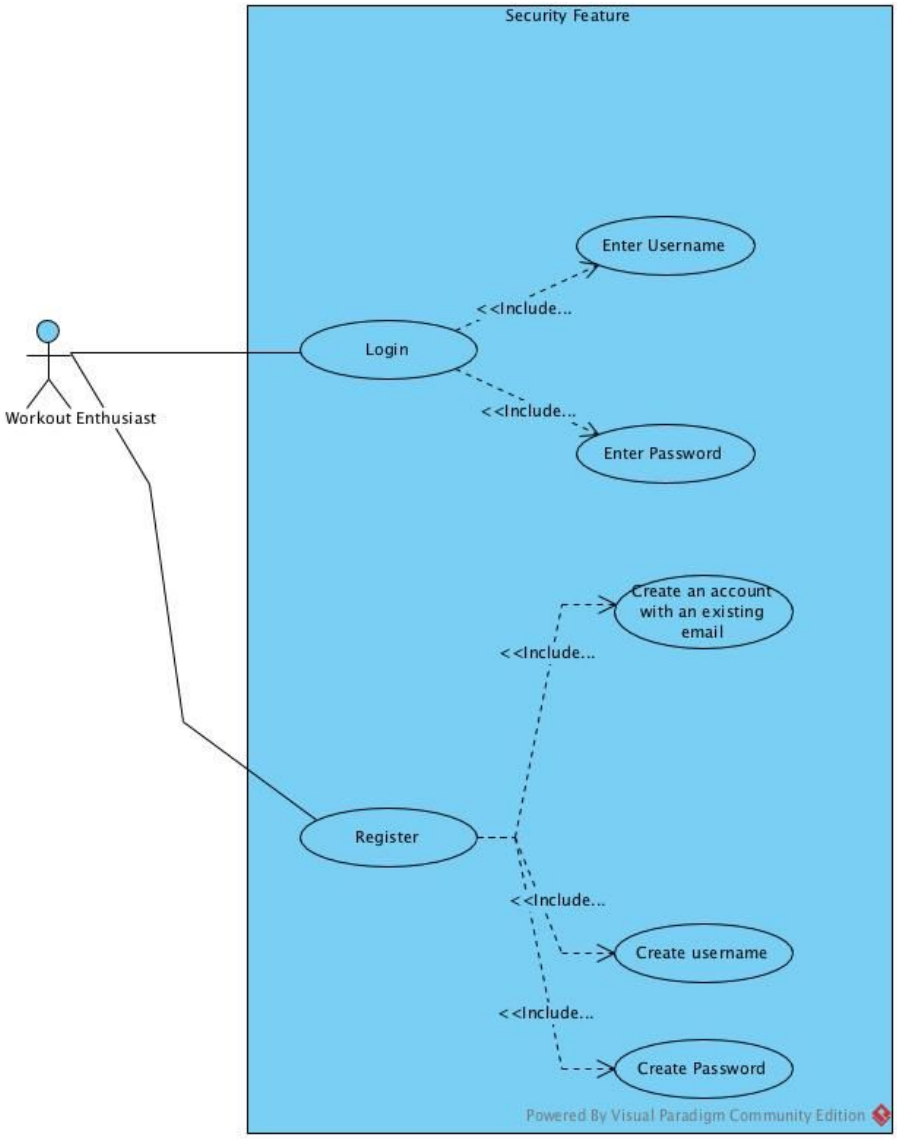
3.7 Other Requirements

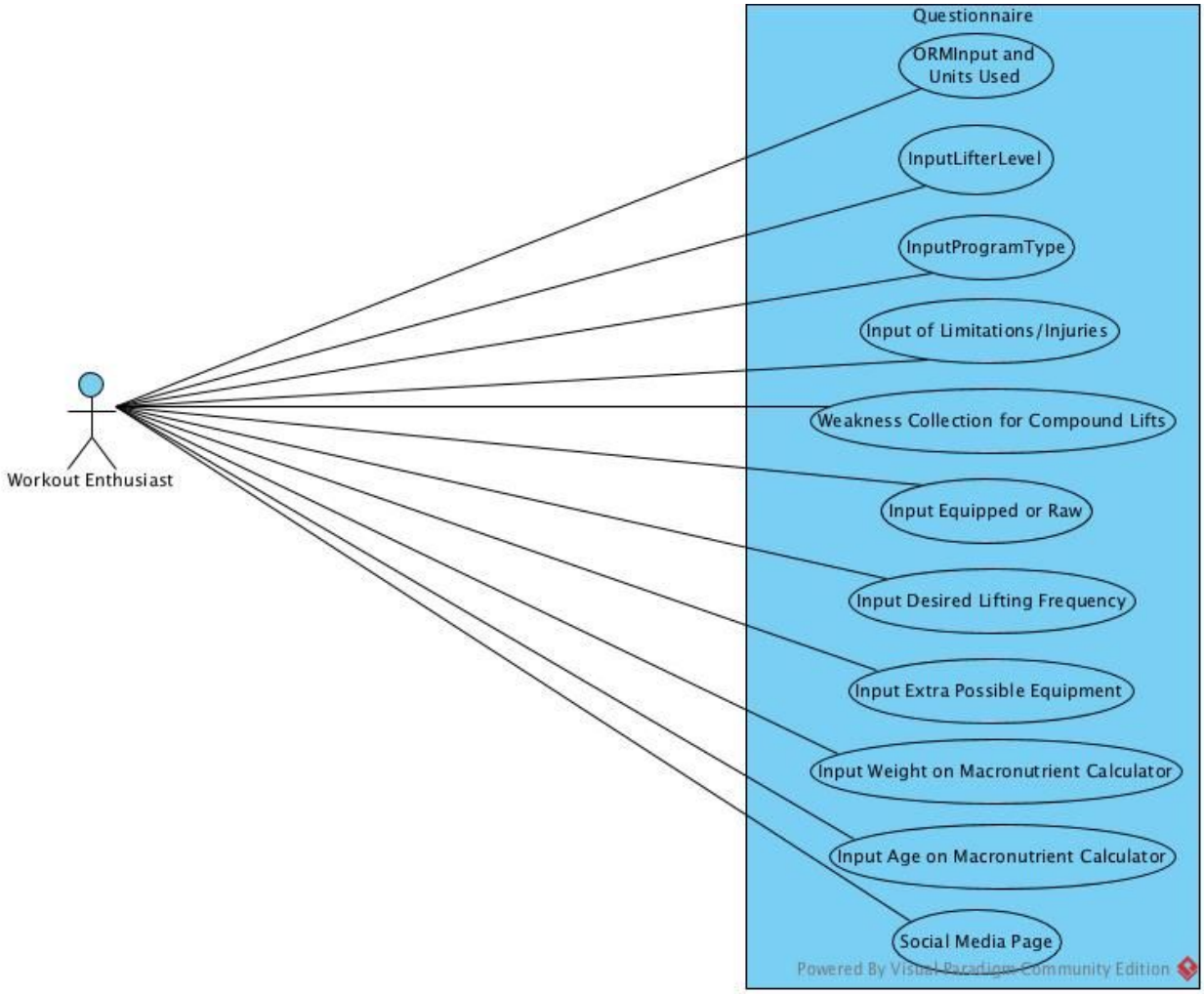
Catchall section for any additional requirements.

4. Analysis Models

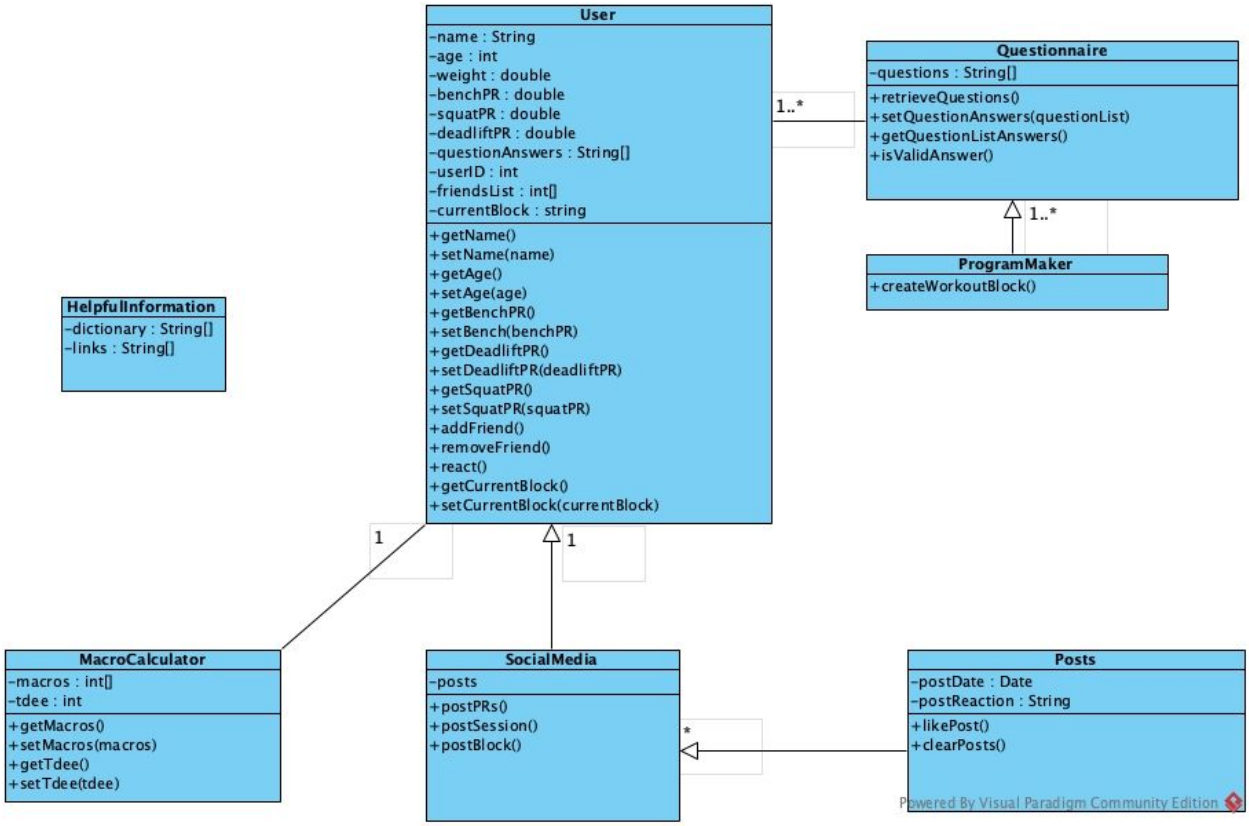
List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS's requirements.

4.1 Use Case Diagrams

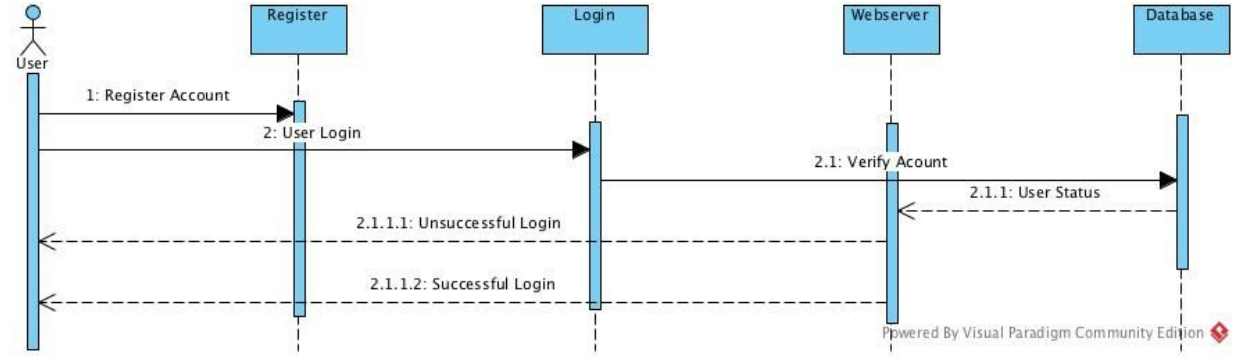

Security Feature
Enter Username
<<Include...
Login
<<Include...
Enter Password
Workout Enthusiast
Create an account with an existing email
<<Include...
Register
<<Include...
Create username
<<Include...
Create Password
Powered By Visual Paradigm Community Edition

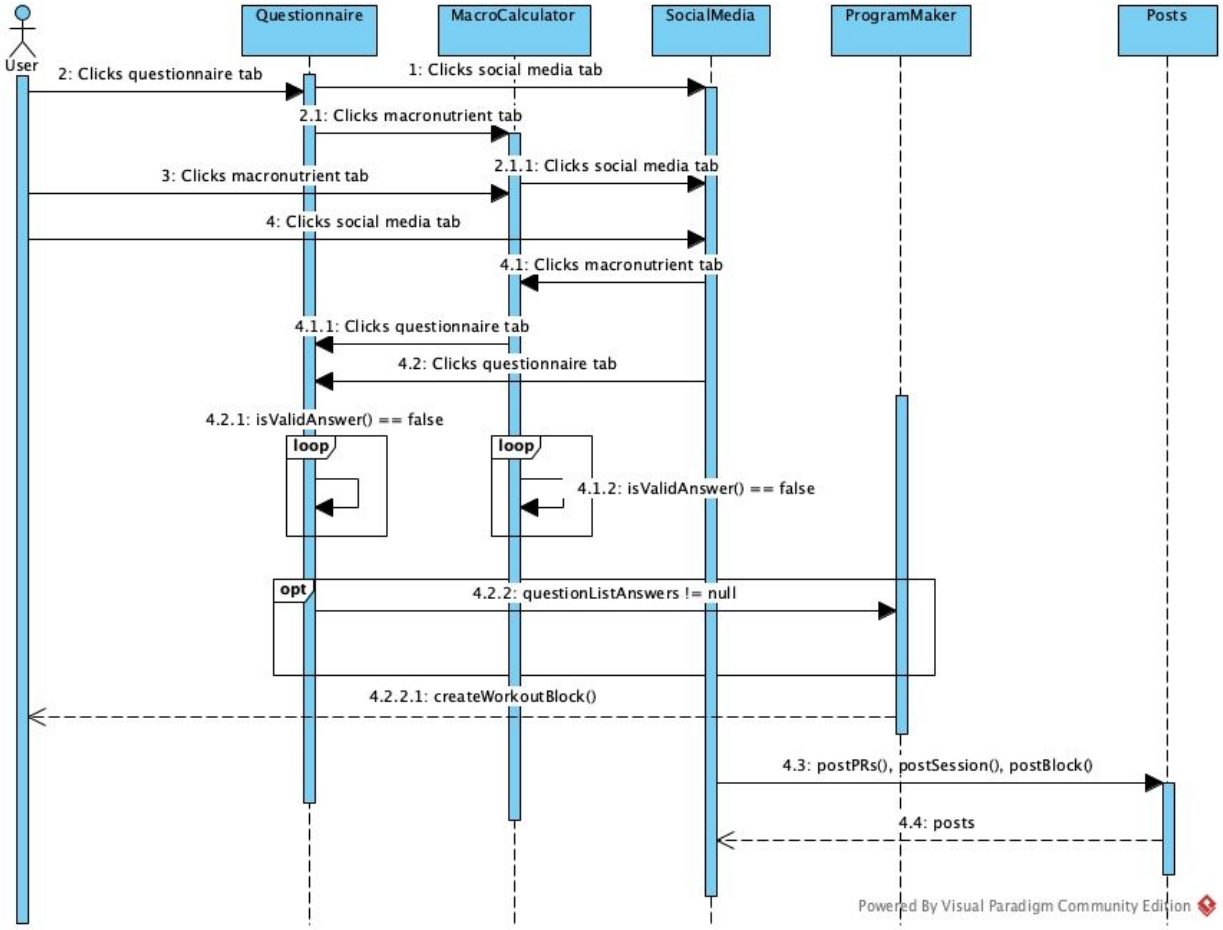


4.2 Class Diagrams

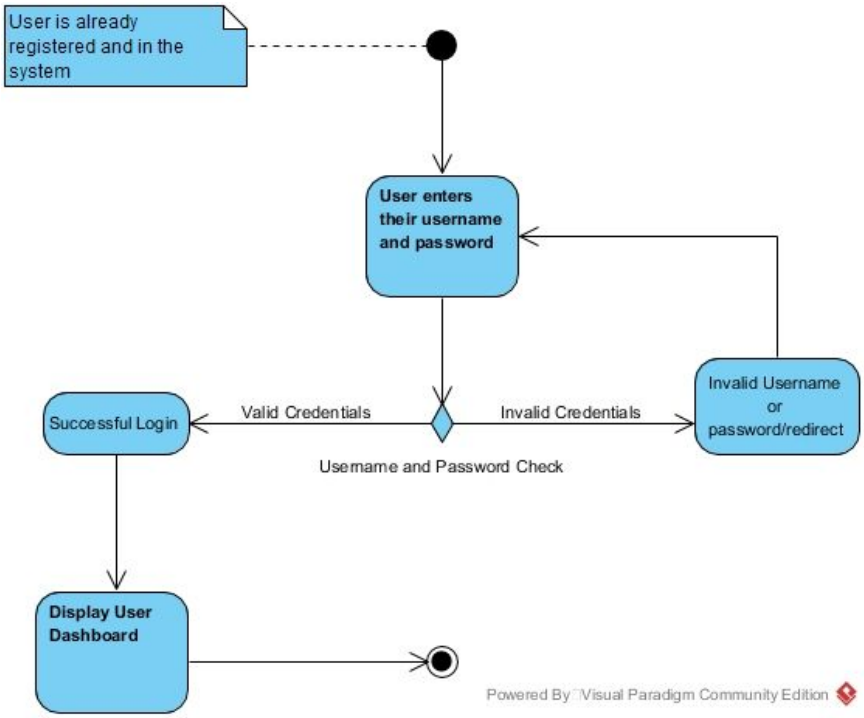


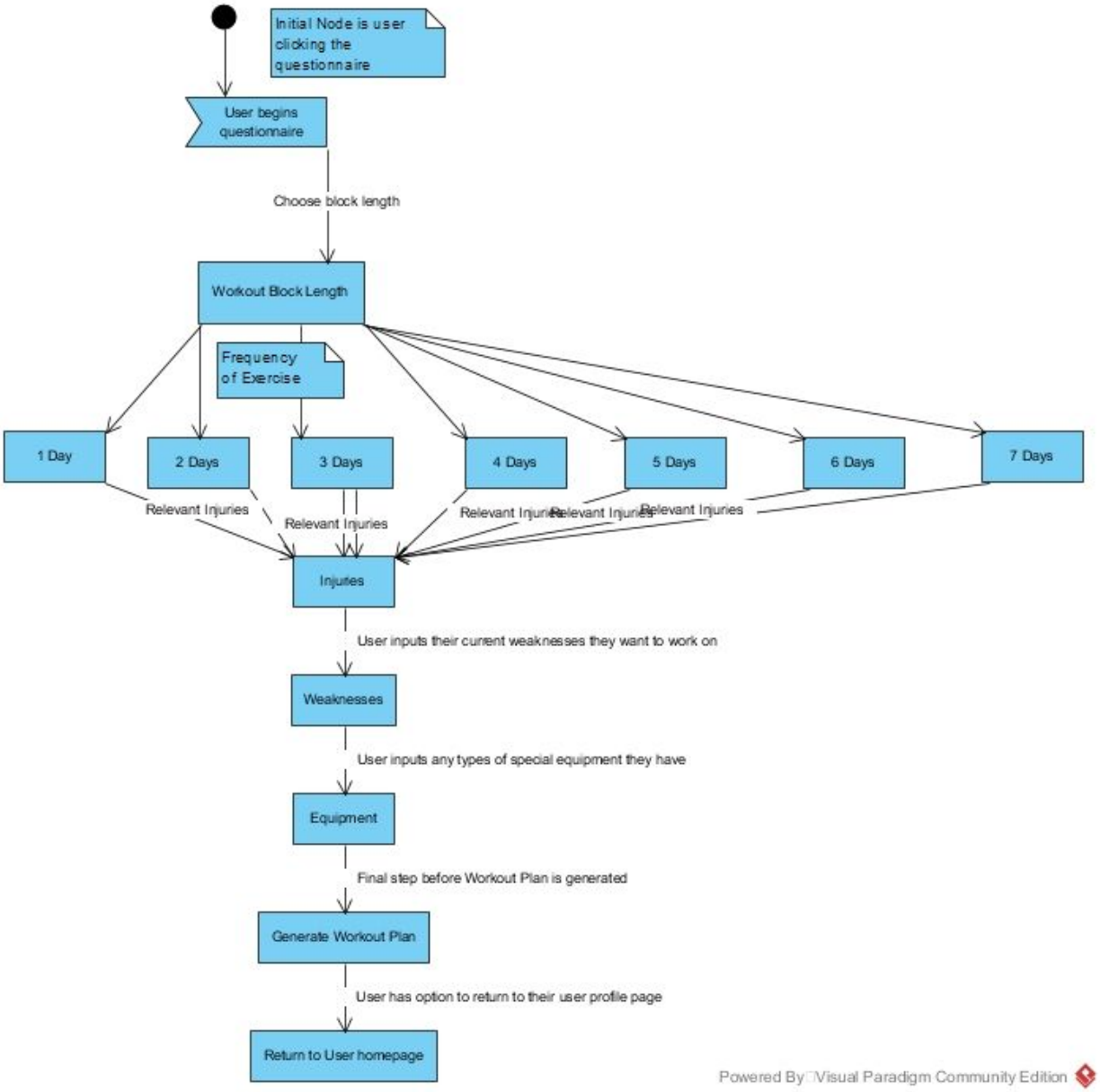
4.3 Sequence Diagrams



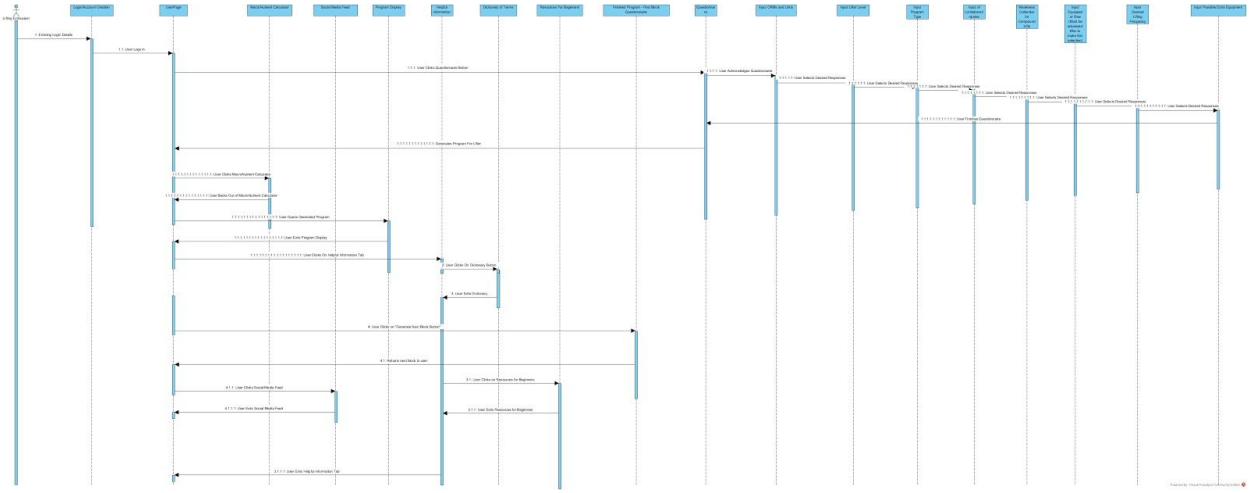


4.4 Activity Diagrams

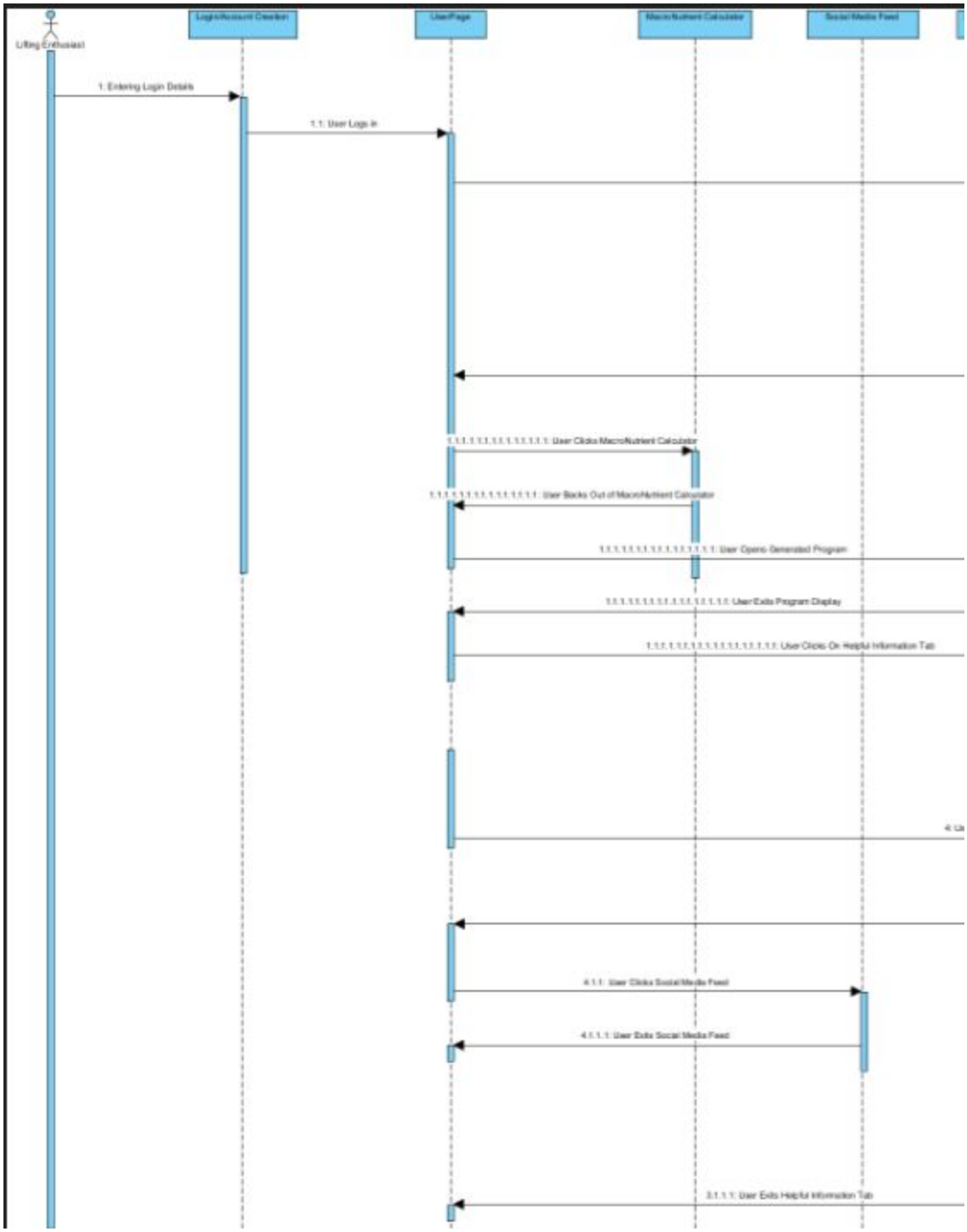

User is already registered and in the system
User enters their username and password
Invalid Username or password/redirect
Valid Credentials
Invalid Credentials
Successful Login
Username and Password Check
Display User Dashboard
Powered By Visual Paradigm Community Edition

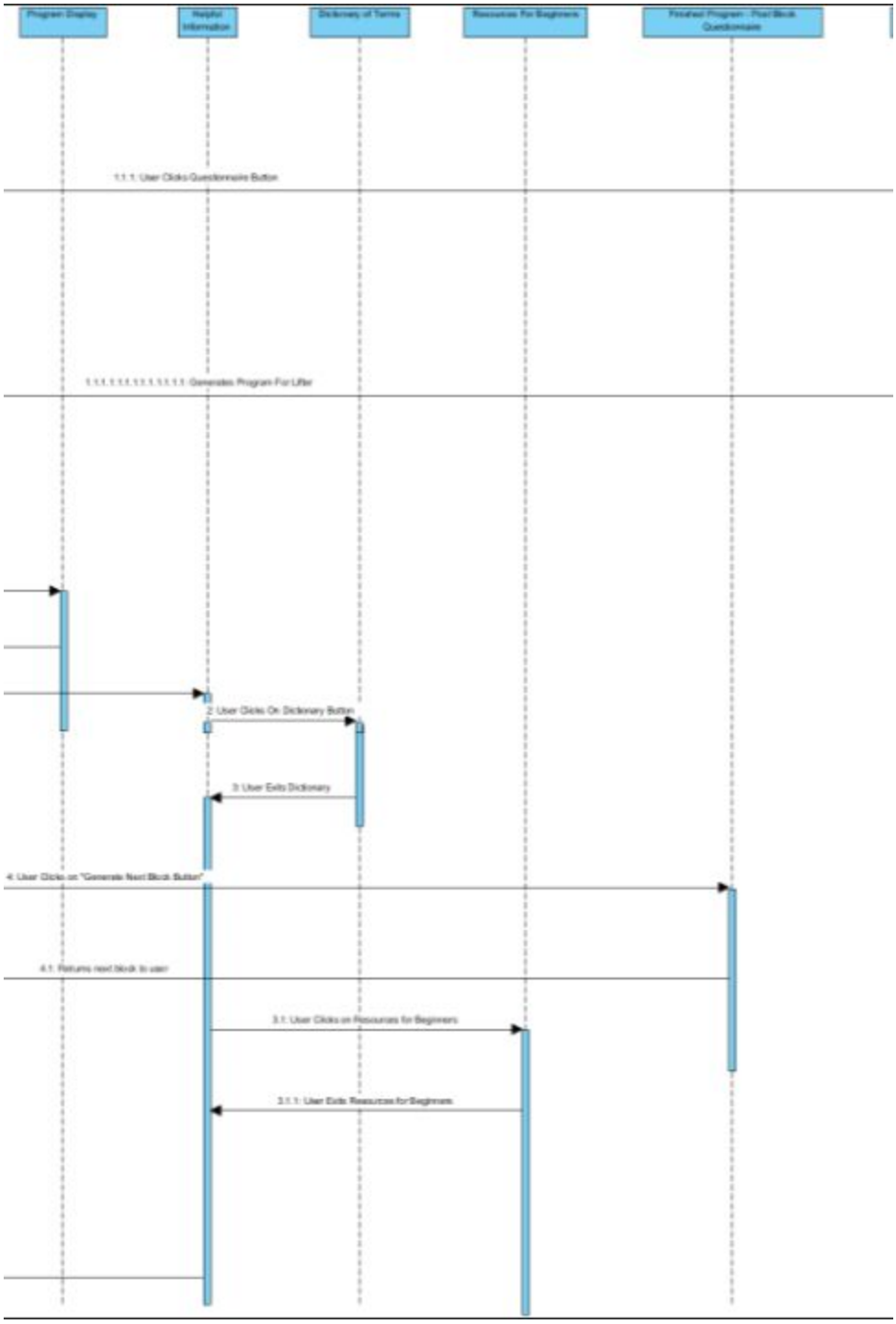


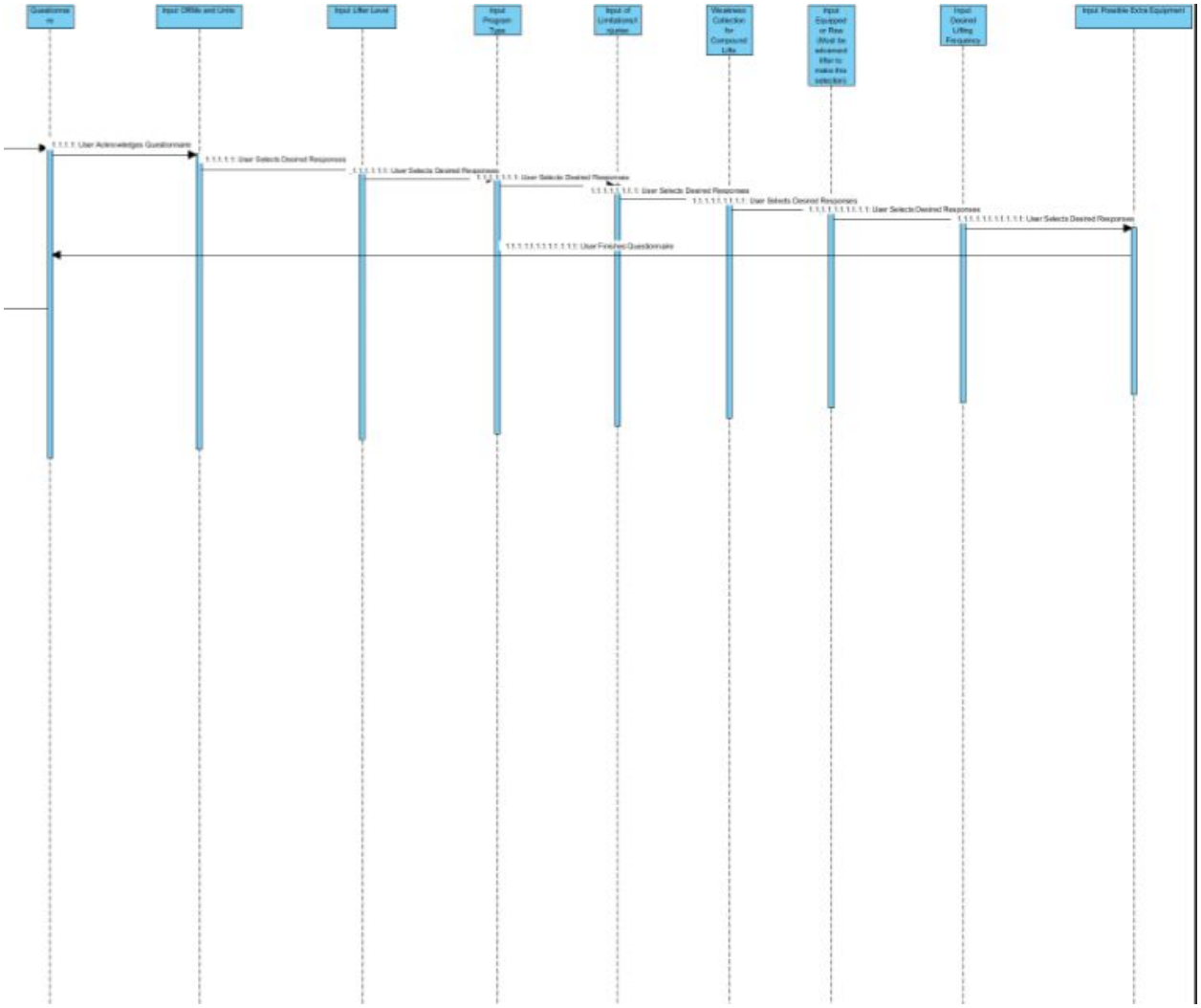
4.5 State-Transition Diagrams (STD)



Log/Account Creation
UserPage
Mechanical Calculator
Social Media Feed
Program Display
Helpful Information
Dictionary of Terms
Resources For Beginners
Finished Program - Post Block Questionnaire
Questionnaire
Data Office and Clerk
Input Chief Level
Input Program Type
Input of Lumpsum Loans
Mechanical Calculator for Lumpsum Life
Input Chief Level for Best Chief Level (Mechanical Calculator Life Lumpsum)
Input Chief Level for Long Pressure
Input Possible Cost Equipment
1: Entering Login Details
1.1: User Login In
1.1.1: User Clicks Questionnaire Button
1.1.1.1: User Acknowledges Questionnaire
1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1.1.1.1.1: User Selects Desired Responses
1.1.1.1.1.1.1.1.1.1.1.1.1: User Finished Questionnaire
1.1.1.1.1.1.1.1.1.1.1.1.1.1: Generates Program Post Offer
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Clicks Mechanical Calculator
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Bailes Out of Mechanical Calculator
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Opens Generated Program
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Exits Program Display
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Clicks On Helpful Information Tab
2: User Clicks On Dictionary Button
3: User Exits Dictionary
4: User Clicks on "Generate Next Block Button"
4.1: Refuse mechanical to user
3.1: User Clicks on Resources for Beginners
4.1.1: User Clicks Social Media Feed
4.1.1.1: User Exits Social Media Feed
3.1.1: User Exits Resources for Beginners
2.1.1: User Exits Help for Information Tab


Lifting Enthusiast
Login/Account Creation
User/Page
Macro Nutrient Calculator
Social Media Feed
1: Entering Login Details
1.1: User Logs in
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Clicks MacroNutrient Calculator
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Backs Out of MacroNutrient Calculator
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Opens Generated Program
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Exits Program Display
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: User Clicks On Helpful Information Tab
4: Cl
4.1.1: User Clicks Social Media Feed
4.1.1.1: User Exits Social Media Feed
3.1.1.1: User Exits Helpful Information Tab


Program Display
Helpful Information
Dictionary of Terms
Resources For Beginners
Finished Program - Post Block Questionnaire
1.1.1: User Clicks Questionnaire Button
1.1.1.1.1.1.1.1.1.1.1.1.1.1.1.1: Generates Program For Lifter
2: User Clicks On Dictionary Button
3: User Exits Dictionary
4: User Clicks on "Generate Next Block Button"
4.1: Returns next block to user
3.1: User Clicks on Resources for Beginners
3.1.1: User Exits Resources for Beginners



5. Change Management Process

To implement ad hoc changes and ensure that all changes meet all standards, we will incorporate the following strategy:

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.
If there is a need for any sort of changes in project scope or requirements it will need to be approved by the unanimous approval of the entire team. Anyone can submit changes to the project if they feel it is necessary. This can be anyone on the team itself, or perhaps a customer who wishes to add some sort of requirement to the product. There will be a variety of means to submit the changes. This can be via formal letter, phone call, email, or even direct interaction

with any one of the team members to consider a change in the program. After receiving the proposed changes, the group will need to discuss the changes and reach a consensus to apply the changes. Otherwise the recommendation for a particular change will be discarded.

A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS's overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.