Project Name

Software Requirements Specification

PRJ566 – Winter 2024

PRJ566 – Team No: 2his

Name of Project: Project Exercise

Project Leader: Mara Perkons

Last updated: Apr. 12, 2024

Team Members:

1. Mara Perkons - 133599225
2. Devon Chan - 066869132
3. Andrew Sequeira - 055099063

Project Name

TABLE OF CONTENTS

Project Name

1- Introduction/Overview- Document Information

1.1 Document Authors

Mara Perkons

Devon Chan

Andrew Sequeira

1.2 Revision History

See document history through different weeks in the Teams deliverables folders or by looking at the version history through SharePoint.

1.3 Document Conventions

Any text in red indicates an exception or error.

Any text in blue is in-progress.

Any text highlighted in yellow is an important point.

Any text in green was recently added.

Any text *italicized* represents definitions.

Any text with ~~strike-through~~ is deleted.

1.4 Document Purpose

The purpose of this document is to serve as an outline and agreement of the plan for the development of a software application for Project Exercise. It is a living document, available to update as needed until client (faculty) sign off is acquired by the end of the Winter 2024 term. The document serves as a guide for the project plan.

1.5 Intended Audience

The intended audience for this document includes the individuals and teams involved in the development and management of this software project. Specifically, this includes Devon, Mara, and Andrew for both the project management and development. The class professor (Mohamed Kassim) is also involved in helping manage the project.

Project Name

1.6 Group Agreement

TEAM AGREEMENT

Team #: 2

Project Title: Project Exercise

Project Time Frame: January 9 2024 – August 16 2024

Team Members:

Mara Perkons, Devon Chan, Andrew Sequeira

Team Leadership:

Mara Perkons

Team Functions:

- *We will share information through MS Teams, OneDrive, e-mail and meetings.*
- *Provide opportunity for equal participation by each member*
- *Work together to complete tasks and meet deadlines*
- *Let the team lead know well in advance if a member is having trouble completing their task by a given deadline*
- *Strive to complete tasks well in advance of submission deadlines to allow for individual components to be unified and for any changes to be made*
- *Be open to new approaches and consider new ideas*

Team Meetings: We will

- Hold meetings every Tuesday after class, Thursday during class time
- Schedule additional meetings as needed
- Post a summary of action items decided upon during the meeting in the teams channel
- Agree to come prepared and allow each member opportunity to present their ideas
- Let the team know well in advance if a member has to miss a meeting, or come late

Team Problems: We will

- ***Strive t****o resolve any conflict with open and honest communication, while focusing on solving the problem instead of laying blame*
- *provide opportunity for a team member to fix issues before more concrete consequences are considered*
- *vote to remove members who are lacking in participation/communication if sufficient improvement is not made by the member after being notified of the issue(s)*

Team Commitment

Project Name

The undersigned members agree to work together on the project until the end of the PRJ666 next Semester. They recognize that as a team and individually they are responsible for the quality of all deliverables.

Name	**Date**
Mara Perkons	January 11, 2024
Devon Chan	January 11, 2024
Andrew Sequeira	January 11, 2024

2- Project Overview

2.1 Project Proposal

Project Background

This project aims to develop a brand new, customizable, and flexible exercise app to help users on their fitness journey in a more personalized way. As health and wellness plays a large role in our lives, many people look to apps to help guide them. However, most currently available apps are rather generalized and rigid in their available workout structures. While the intent to actively work on fitness may be there, many people encounter roadblocks such as a lack of variety and flexibility in their workout apps, and struggle to overcome motivational challenges to stick with their intent. Project Exercise aims to create an app that allows clients to tailor their workout to their needs and preferences, and that helps them stay motivated to improve their health and wellbeing by encouraging them while keeping them updated on their progress.

Problem Statement

The Problem of:	Current Fitness apps don't cater enough towards the individual but rather the masses and generalized workout plans
Affects:	People currently using other fitness apps or those who want to get into working out but don't know how to start/what to do
The impact of which is:	Many cannot personalize their routines and apps, and will not see the results they want and end up quitting working out

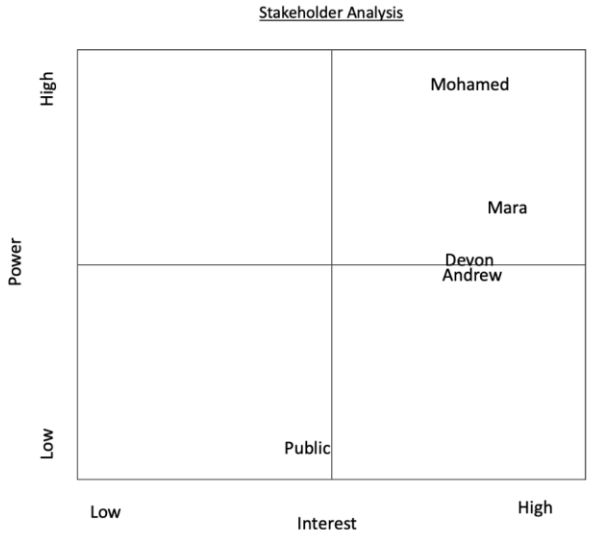
Project Name

A successful solution would:	Allow workouts and exercises to be personalized to users, while also showing them their progress to help keep them motivated

Product Vision

For	Individuals interested in fitness and health
Who	Value personalized workout planning, diversity of exercises, and real-time progress tracking
The (Product Name)	Fitness App
That	Will cater to each individual based on their goals
Unlike	Traditional fitness apps that provide static workout routines without adapting to individual progress or goals such as Progression
Our product	Will allow users to customize their experience in addition to providing more targeted exercises and plans to meet their desired goal

2.2 Stakeholders and Users



Name	Title	Project Role	Contact Info	Notes	Power	Interest	Ranking
Mohamed Kassim	Professor	Sponsor	mohamed.kassim@senecapolytechnic.ca	Weekly reports	H	H	1
Mara Perkons	Project Leader	Developer	mperkons1@myseneca.ca	Notify if absent	M	H	2
Devon Chan	Team Member	Developer	dchen80@myseneca.ca		M	H	2
Andrew Sequeira	Team Member	Developer	asequeira@myseneca.ca		M	H	2
General Public	User	End User			L	M	4

2.3 Functional Requirements

1. Create User Account

1.1. User indicates need to register or login.

1.1.1. System provides user with ability to log in with Apple or Google Authentication.

1.1.2. System direct users to Apple or Google's secure log-in portal and wait for access token to verify user login.

1.2. User indicates need to update their profile.

1.2.1. System prompts users for body information: weight, height, sex, age, and fitness levels to determine the users maintenance calories (amount of calories consumed to stay at current body-weight).

1.2.3. System prompts users on fitness goals, fitness levels, exercise preferences, available equipment, and health concerns.

1.2.4. System displays users updated profile.

2. Workout Tracker

2.1. User initiates a workout routine.

2.1.1. For repetition-based exercises:

2.1.1.1. The system prompts users for number of reps completed.

2.1.1.2. The systems displays a countdown clock for rest periods or move on to the next exercise appropriately.

2.1.2. For time-based exercises:

2.1.2.1. The system displays a countdown clock for users when they start the exercise.

2.1.3. The system stores and records the following statistics:

2.1.3.1. Number of reps and sets completed.

2.1.3.2. Time required to finish each exercise.

2.1.3.3. Total time to complete the routine.

2.1.3.4. Estimated calories burned.

2.1.4. For any exercise, users may choose to substitute exercise with a similar exercise of their choice.

3. **Routine Builder**

3.1. User wishes to create a workout plan.

3.1.2. System allows structuring the workout plan by specific days or week cycle.

3.1.3. System allows inserting any type of exercises into a specific day.

3.1.3. System allows insertion of any exercise:

3.1.3.1. Time-based exercises must insert desired time.

3.1.3.2. Repetition-based exercises must insert desired repetitions.

3.1.4. System allows grouping exercises by training style, such as: circuit training or super-setting exercises.

3.1.5. User may choose to generate a workout plan based on user-preferences given inside their user profile page.

~~**4. Calorie Tracker**~~

~~4.1. User wishes to add a food item for today.~~

~~4.1.1. User enters food item and amount consumed.~~

~~4.1.2. System uses food API to determine calories, fat, carbs, and protein intake.~~

~~4.2. System displays user's maintenance calories, amount of calories burned, and total calories consumed today.~~

5. Fitness Progression Tracker

5.1. User wishes to check fitness progression.

5.1.1. The system displays statistics and trends based on past workout, weight, and goals.

5.1.2. The system uses AI to provide user feedback for improvements and suggestions based on past workout data and goals.

2.4 Nonfunctional Requirements

Operational Constraints:

- Availability:
  - app should be available to all users 24/7 and have very little downtime for maintenance and upgrades so as to not disrupt users workouts
  - Any downtime for maintenance should be communicated to users in advance
- Scalability:
  - Should be able to handle an increased number of users at any time, especially during peaks hours for going to the gym (ex. After 5pm on weekdays when everyone is off work)
  - Performance should be unchanged during increased user load

Performance Constraints:

- Fast Response Time
  - Loading time of app should be minimal (3 sec maximum)
  - Response times for all user interactions should be minimal, for example loading workout routines and checking on their individual progress
- Database Storage
  - Efficiently store and retrieve user data such as workout history
  - Queries should be optimized for performance

Security Constraints

- User Data Protection
  - Ensure user data such as login credentials and personal information are stored securely and protected against unauthorized access
- Authentication and Authorization
  - Should be implemented securely and include strong encryption to protect against bad actors

Usability and User Experience Constraints

- User interface consistency

- App should have nice and consistent user interface that is very intuitive and easy to use

2.5 Project Scope

Project Goals and Objectives

Project will deliver a working web app to browse workouts, create custom workouts, and create custom routines.

- Webapp must be able to run on any browser
- Webapp UI/UX must be optimized for mobile browsing
- Offers a library of preset workouts and routines with detailed information on required equipment, fitness levels, and goals
- Integration with Google/Apple Authentication
- Provide users with a user-friendly platform to initiate and track workouts and routines

Project Boundaries

Within Scope:

- Development of core features outlined in Functional Requirements
- Integration with external APIs for exercise database, and AI technology
- Testing and quality insurance to ensure app meets functional and performance requirements
- Deployment of app
- Documentation of app's functionalities

Out of Scope:

- Direct integration with fitness trackers and other wearable devices
- Connection to social networking platforms within the app for users to connect and share their progress
- Providing comprehensive meal planning and nutrition tracking
- Legal and regulatory compliance beyond basic data privacy requirements
- Hardware and server maintenance for app hosting

2.6 System Risks

Risk	Response
None of the team members have much experience with incorporating AI into programs	The team will research and learn individually how to utilize AI and will hold sessions to share information learned

Some team members have not used Apple and Google built in authentication, introducing security risks if implemented incorrectly	The members familiar with the systems will review or implement the Apple / Google log in functionality
Utilizing AI to provide users with improvement suggestions based on past progress introduces complexity which may lead to security risks	Data privacy requirements regarding AI will be reviewed and adhered to
Data privacy risks related to handling sensitive user data (especially health-related information) could lead to more sever repercussions in the event of a data leak	Team members will review and adhere to health data protection regulations
The amount of data stored in our database may begin to exceed the space available for a free cluster	Should the risk occur, the team will review possible actions such as purchasing increased capacity or clearing inactive accounts
Difficulty in attracting users away from established fitness apps could limit the project's success	Work to ensure our app's uniqueness

2.7 Operating Environment

Software Environment

- Cross platform compatibility on various operating systems (IOS, Android, windows) and various web browsers (Chrome, Edge, Firefox etc)
- Responsive design, providing users with the best possible viewing experience across various devices (ex ipad, iphone, android, tablet)

Hardware Environment

- Enough server capacity to handle high volumes of concurrent users specifically during peak workout hours (ex. right after end of workday, 5pm)
- Reliable and fast content delivery of variousno content such as images or videos to explain workouts/exercises

Security Environment

- Secure authentication mechanisms to protect user accounts
- Various methods for sign up and sign in
- Adhere to best practices for web applications to prevent common vulnerabilities like SQL injection
- Regular updates or patches in addition to security audits to address any possible security vulnerabilities

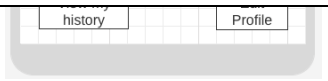
Compliance Environment

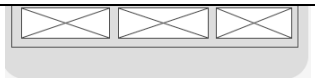
- Adhering to data protection laws for customer privacy
- Transparency with users about use of their data if any
- Compliance with health data protection regulations

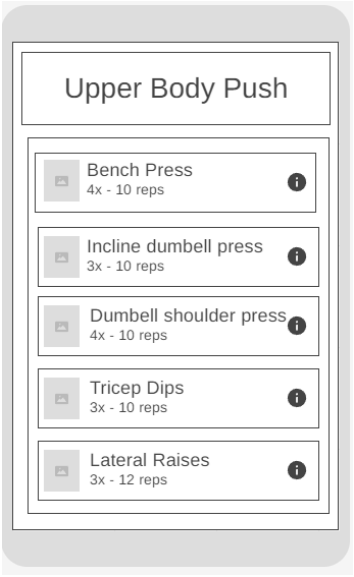
2.8 UI/UXD Interface Mock-ups

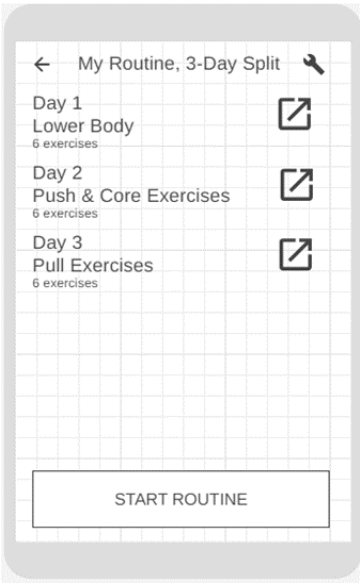
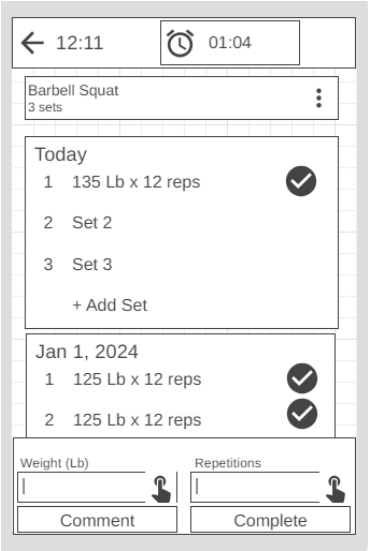
Commented [AS1]: add numbering

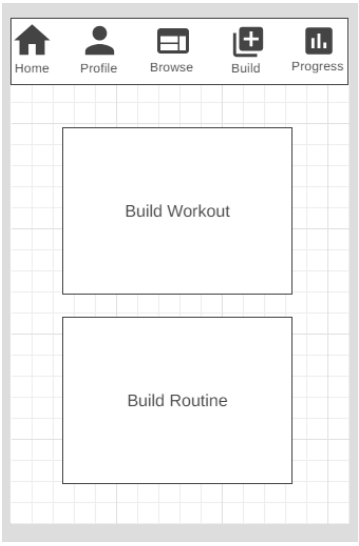
MU01		- Automatically redirected to this page if the user is not logged in - https://wireframe.cc/K99WvV

MU02		- The landing page for the app once logged in - Includes the user's "favourited" routines and workouts - Able to filter by routine, workout, category, etc. - https://wireframe.cc/2UW4yn
MU03		- Pressing the "Profile" menu item in the nav bar sends the user here - Includes the profile details the user has entered - Allows for editing of these details to reflect any updates - User can navigate to their workout history through this page - https://wireframe.cc/VuCjm0

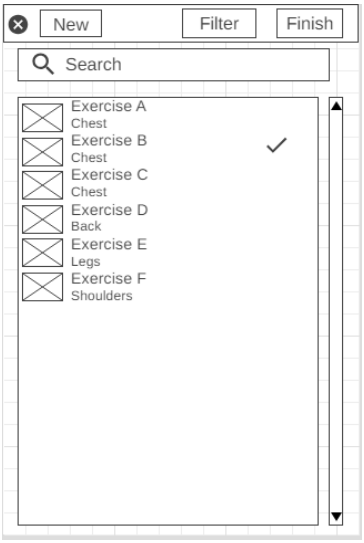
MU04		- Pressing the “view my history” button from the profile page will send the user here - Includes a list of previously completed workouts/routines - https://wireframe.cc/sEoUud
MU05		- Ability to display various forms of data - Graph and stats - See various metrics like weight gained/lost, weight progression of exercises - https://wireframe.cc/bABE5e

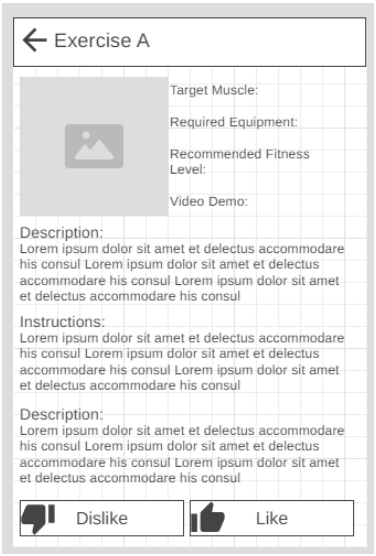
MU06	Browse workouts and routines page 	- Able to browser either preset workouts or routines and sort by various filters such as time, exercise type etc - https://wireframe.cc/np8gdh
MU07	Workout Details Page 	- Display exercises within a workout - Ability to get more information about an exercise (such as muscles worked) - https://wireframe.cc/R3h4Tb

MU08	Routine Details Page 	- Displays the selected routine - Displays all the workouts that a routine contains - Ability to modify the routine - https://wireframe.cc/JHxBJX
MU09	During Workout Page 	- Displays the current exercise to perform, and the past workouts done with this exercise - Able to set weight, reps, or time and complete a repetition - Displays workout's total time on top left - Displays rest timer on top right - Able to add extra sets or press the settings to change the - https://wireframe.cc/XSfmQA

MU10	Build Page 	- Pressing the "Build" icon in the navigation bar sends users to this page - Can select to build a workout or build a routine - https://wireframe.cc/PdlUyI
MU11	Routine Builder Page 	- This page allows users to build a routine from scratch - Pressing the edit button for "Routine Name" allows users to change the routine name - Pressing the edit button for a particular Workout allows users to edit the workout and transfers users to the workout editor. - Pressing Add Workout allows users to go to the browse workout page to add a workout to the routine. o Can add an empty workout - https://wireframe.cc/haM88N

MU12	Workout Editor Page 	- Lists all exercises inside the workout - Able to remove, change, or add exercises to a workout - Able to group exercises (for circuit training or super setting) - https://wireframe.cc/wpw4n5

MU13	Exercise Selector Page
	 
		- Lists all exercises
		- User can select exercises to add to a workout
		- User can search or filter exercises
		- Filter Options:
		  ○ By muscle group
		  ○ By exercise type (plyo, weightlifting, etc)
		- https://wireframe.cc/EQWfok

MU14	Exercise Details Page 	- Displays information on a particular exercise - Users may like or dislike the exercise which adds it to the users preference - https://wireframe.cc/IdsJtU
MU15	Workout Summary Page 	- After workout complete, shows summary - Shows information like calories burned, time taken, and number of exercises completed. - https://wireframe.cc/pWONlM

Project Name

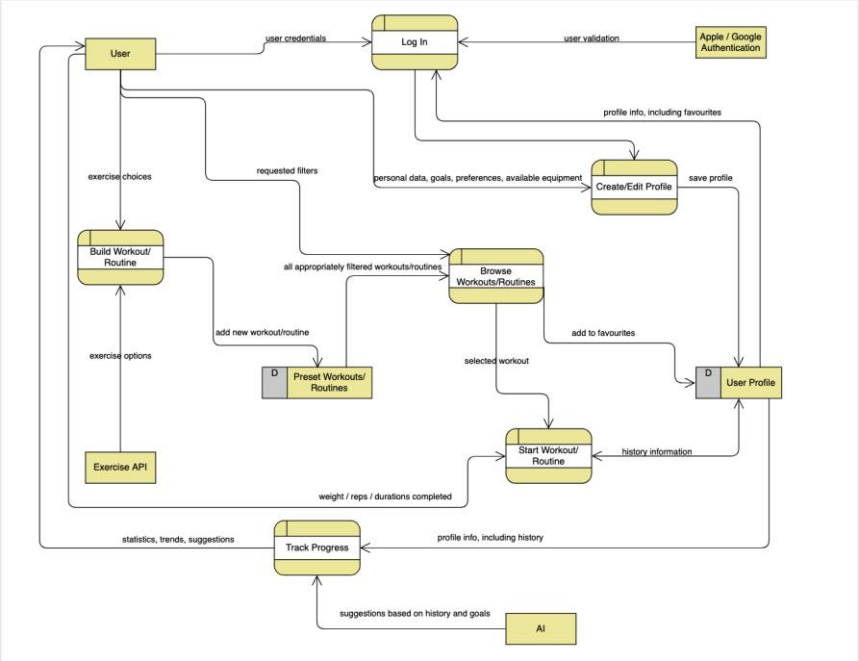
Project Name

3. Process and Data Modeling

3.1 UML/DFD Modeling and Data Modeling

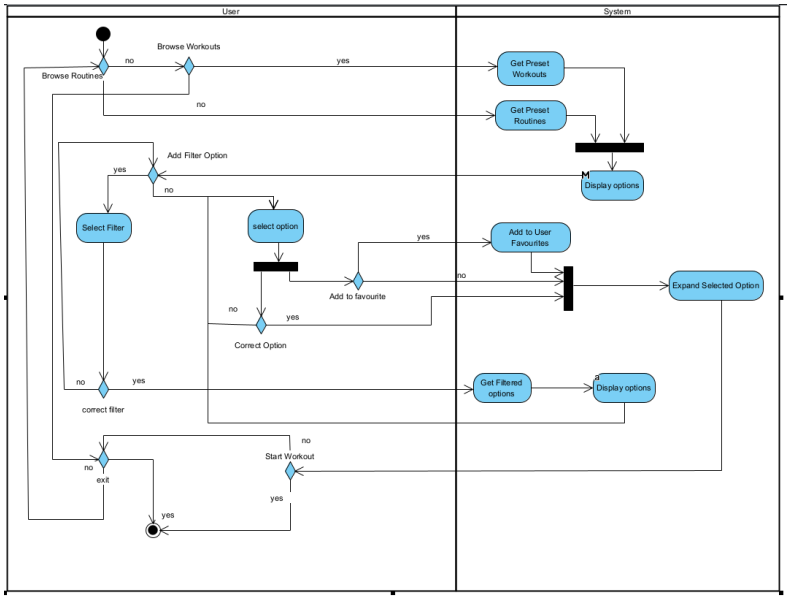
Activity Diagrams and Data Flow diagram

Data Flow Diagram for major processes:

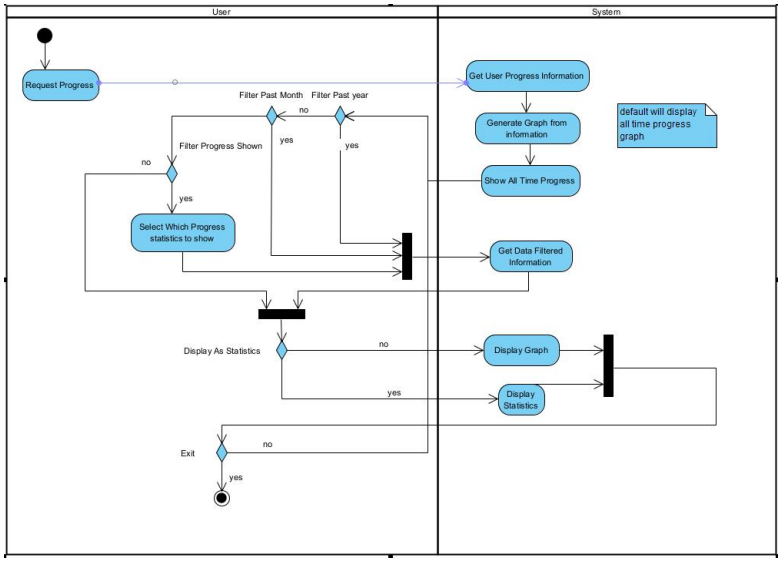


Activity Diagram for the browser feature (browsing routines and workouts) - UI mockups:

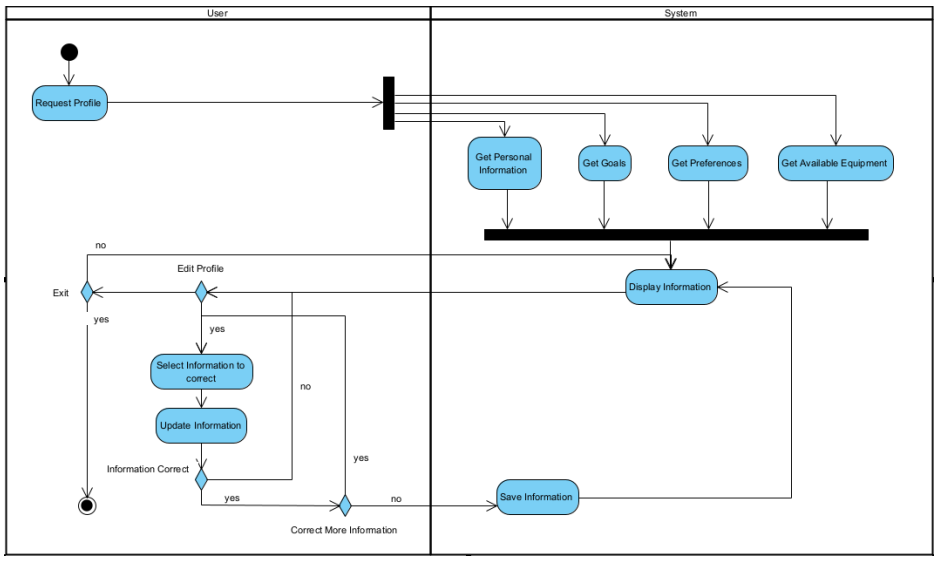
Commented [AS2]: add UI numbers after implemented



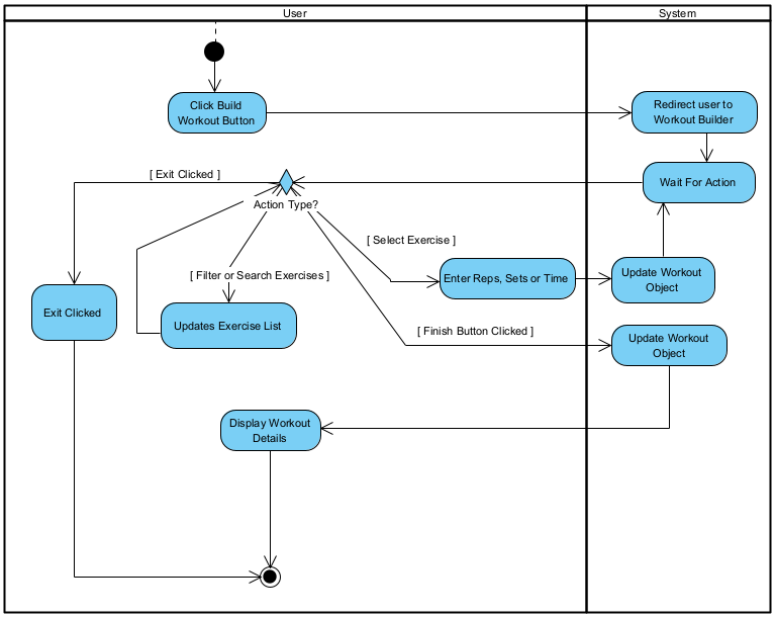
Activity Diagram for tracking progress



Activity Diagram for checking/updating profile

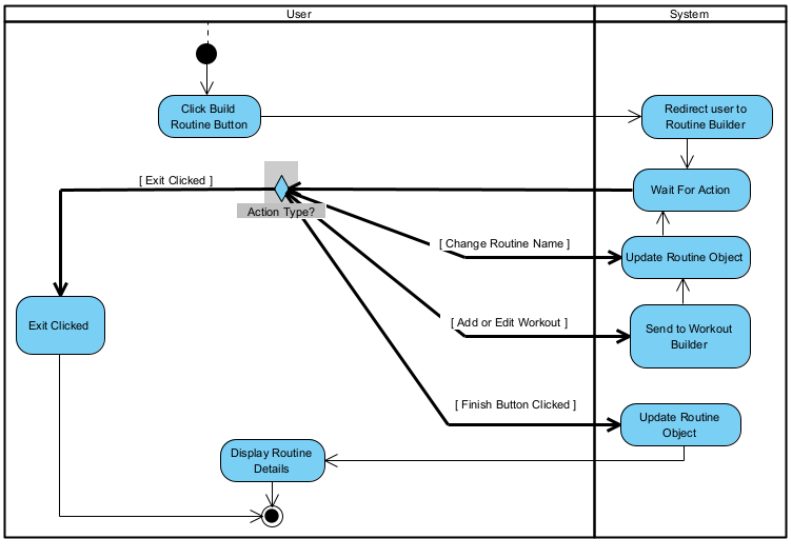


Activity Diagram for Building Workout:

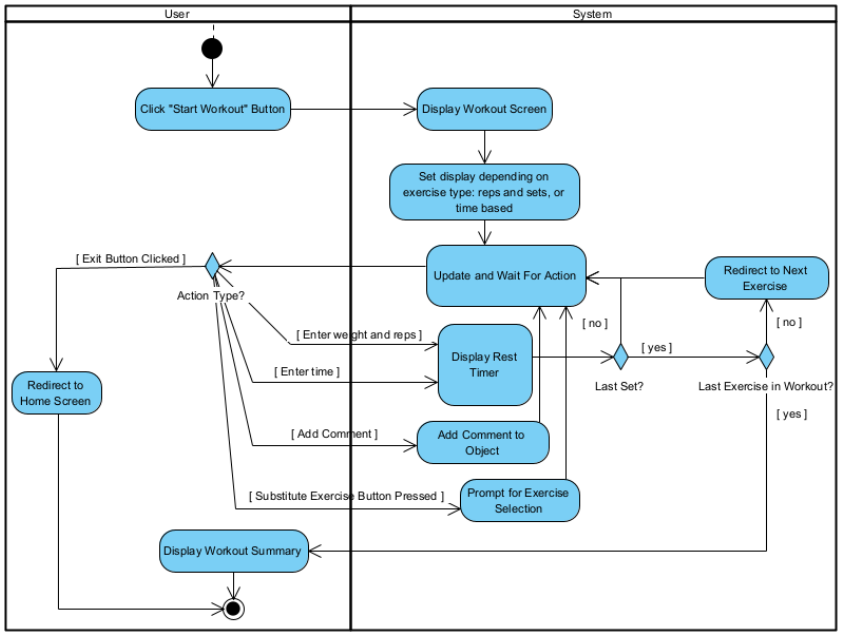


Ca

Activity Diagram for Building Routine:



Activity Diagram for Starting Workout:



3.2 Business Rules

Business Rule Number	Business Rule Description	Related UC
BR01	User must be signed in to access app functionality	UC01
BR02	User's workout history must track completion dates.	UC01
BR03	Routine must have a routine name and at least one workout.	UC03
BR04	Routine name can be no longer than 12 characters	UC03
BR05	Workout must have at least one exercise	UC04
BR06	All exercises must have repetition or time specified as well as number of sets.	UC04
BR07	Changes in the user's profile information for variable stats (such as weight) should be kept as a history instead of overwriting the previous value.	UC01
BR08	Exercises should be able to be swapped for a different exercise within the same category during a workout.	UC02
BR09	Exercises must include categorization for related exercise swapping capability.	UC02
BR10	Users must agree to the terms of service & privacy policy during account registration	UC01
BR11	Duplicate email addresses are not allowed (each email can be associated with only one user account)	UC00
BR12	User data must be encrypted	UC01
BR13	Users must have the option to delete their accounts and associated data permanently	UC00
BR14	A user can create a maximum of 15 workouts to avoid overloading the database	UC04, UC05

3.3 Use Case Specifications with corresponding interface mockups:

Each use case needs to have the following:

1- Business Rules.

2- System Use Case Diagrams.

3- Use Case Descriptions.

4- Corresponding Mockups

Use Case: Tracking Progress

Use Case Name: Tracking Progress			ID: UC01	Importance Level: Medium
Primary Actor: User		Use Case Type: Detail, Essential		
Stakeholders and Interests: User – wishes to view how their fitness journey has progressed				
Brief Description: This use case describes how the user might investigate their progress				
Trigger: User wishes to learn more about their fitness progression Preconditions: User has at least one completed workout. Type: External				
Relationships: Association: User Includes: Extends: Generalization:				
Normal Flow of Events:	ID	Actor	System	
	1	User presses “Progress” navigation menu item.	Redirects user to Progress Tracking Page (MU05).	

	2		Performs S-1 gather progress info subflow. By default, displays all time progress graph.
	3	User selects any desired filters to apply.	Retrieves filtered data (by performing S-1 gather progress info subflow) and presents results to the user.
Repeats from step 3 until the user has viewed the information they desire and is satisfied with their investigation	4	User reviews results including feedback and suggestions for improvement	
	5	User chooses to navigate to another page in the app.	Begins the appropriate use case based on navigation selection.
SubFlows: S-1: Gather Progress Information 1. The system fetches user's workout history and profile information 2. The system gathers the relevant data and processes it to produce graphs, trends, stats, etc. 3. The system uses AI to produce suggestions based on the data and user's goals			
Alternate/Exceptional Flows: 3a: User selects timeframe filter, the results should only include the data from the requested time frame (past month, year, etc.) 3b: User selects filters based on exercise type, the results should highlight the relevant information			
Corresponding Mockup Interfaces: MU02, MU05			

Use Case: Starting Workout

Use Case Name: Starting a Workout	ID: UC02	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: User – wishes to begin an existing workout		
Brief Description: This use case describes how the user interacts with the app during a workout, including documenting the reps/times for each exercise completed.		
Trigger: User wishes to start a workout Type: External		
Relationships: Association: User		

Includes: Extends: Browsing Workouts and Routines Generalization:			
Normal Flow of Events:	ID	Actor	System
	1	User presses “Start Workout” button	Redirects user to Workout Screen (MU09). Updates display based on exercise type (reps and sets, or time based)
	1b	If the user chooses to substitute an exercise	Executes the Swap Exercise subflow (S-1)
	2	User completes the exercise	Prompts for either weight/reps or time depending the exercise
	3	Enters completed details, as well as any additional comments	Updates the workout, and displays the rest timer
Repeat from 1b until all exercises have been completed	4		Updates the display with the next exercise
	5	Upon completion of all exercises in workout	Saves the workout in the system and to the user’s history. Displays the Workout Summary Page (MU15).
SubFlows: S-1: Swap Exercise 1. The system fetches related exercise options 2. The user selects one of the exercise options 3. The system removes the old exercise and replaces it with the selected option, while updating the workout S-2: Cancel Workout 1. The user decides not to complete the workout. Presses the exit/back button.			
Alternate/Exceptional Flows: 2a: The exercise is reps/set based, the system prompts for weight and number of reps completed 2b: The exercise is time based, the system prompts for duration 3a: The user enters additional comments for the exercise, and the workout is updated with the comments 5a: If at any point the user decides not to complete the workout, performs the S-2 cancel workout subflow			
Corresponding Mockup Interfaces: MU07, MU09, MU13, MU15			

Use Case: Building a Routine

Use Case Name: Building a Routine			ID: UC03	Importance Level: High
Primary Actor: User		Use Case Type: Detail, Essential		
Stakeholders and Interests: User – wishes to create a new workout				
Brief Description: This use case describes how to make a new routine in the app				
Trigger: User wishes to create a new routine inside our app Type: External				
Relationships: Association: User Includes: Extends: Generalization:				
Normal Flow of Events:	ID	Actor	System	
	1	User presses “Build Routine” button	Redirects user to Routine Builder Page.	
	2	**If:** User presses change routine name.	Prompt for new Routine Name.	
	2b	Input routine name	Update routine object	
	3	**If:** User presses Add Workout button	Redirects user to Workout Builder Page, executing “Build a Workout Use Case”	
	4	**If:** User presses Edit Workout button	Redirects user to Workout Builder page, along with selected workout object, and executes “Build a Workout Use Case”	
	5	Repeat above steps until user is satisfied with routine.		
	6	User presses Finish Button	Redirects user to the routine details summary. Saves the routine into the system.	
SubFlows:				
Alternate/Exceptional Flows: 4a: User chooses not to create a routine. Presses exit button.				
Corresponding Mockup Interfaces: MU10, MU08, MU11				

Use Case: Building a Workout

Use Case Name: Building a Workout			ID: UC04	Importance Level: High
Primary Actor: User		Use Case Type: Detail, Essential		
Stakeholders and Interests: User – wishes to create a new workout				
Brief Description: This use case describes how to make a new workout in the app				
Trigger: User wishes to create a new workout inside our app Type: External				
Relationships: Association: User Includes: Extends: Building a Routine Generalization:				
Normal Flow of Events:	ID	Actor	System	
	1	User presses “Build Workout” button	Redirects user to Workout Builder Page.	
	2	If user searches or filters exercise list	Updates exercise list with query results	
	3	User selects an exercise	Prompts user for number of reps, sets, or time for the exercise.	
	4	Repeat above until all exercises are selected.		
	5	User presses Finish Button	Redirects user to the workout details summary. Saves the workout into the system.	
SubFlows:				
Alternate/Exceptional Flows: 4a: User chooses not to create a workout. Presses exit button.				
Corresponding Mockup Interfaces: MU10, MU12. MU13, MU14, MU07				

Use Case: Browsing Workouts and Routines

Use Case Name: Browsing Workouts and Routines			ID: UC05	Importance Level: High
Primary Actor: User		Use Case Type: Detail, Essential		
Stakeholders and Interests: User – wishes to browse workouts and routines				
Brief Description: This use case describes how to browse				
Trigger: User wishes to browse workouts and routines Type: External				
Relationships: Association: User Includes: Extends: Generalization:				
Normal Flow of Events:	ID	Actor	System	
	1	User presses “Browse” button	Redirects user to browsing page	
	2	User chooses to browse by workouts or exercises	If user makes no selection – default option, S-1 subflow is performed. If user chooses routines, S-2 subflow is performed	
	3	User selects filter criteria	Updates browsing page based on filter criteria	
	4	User selects one of the options	Displays information for selected workout	
	5	User chooses to start workout	Displays Workout information along with prompts for weights and reps for the first exercise	
SubFlows: S-1: Default option of browsing by workouts is selected - system displays list of prebuilt workouts on browsing page S-2: Selected Routine browsing - system displays list of prebuilt routines on browsing page				

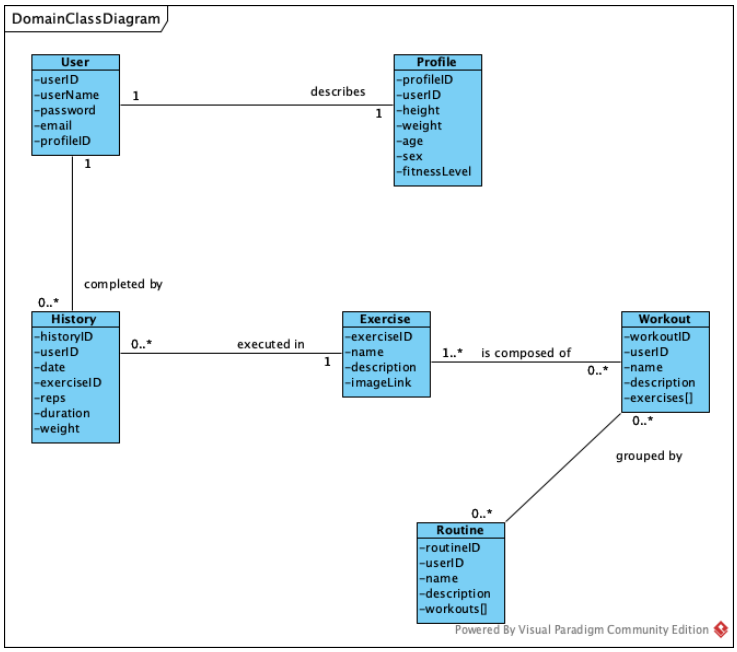
Alternate/Exceptional Flows: 3b: user chooses not to select a filter, moves to step 4 5a: User wants more information on specific exercise in chosen workout 5b User chooses not to do a workout and exits
Corresponding Mockup Interfaces: MU06, MU07, MU09, MU14

Use Case: Updating Profile

Use Case Name: Updating Profile		ID: UC06	Importance Level: Medium
Primary Actor: User	Use Case Type: Detail, Essential		
Stakeholders and Interests: User – wishes to update their profile			
Brief Description: This use case describes how to update a user's profile			
Trigger: User wishes to update profile Type: External			
Relationships: Association: User Includes: Extends: View profile Generalization:			
Normal Flow of Events:	ID	Actor	System
	1	User presses “Profile” button	Redirects user to Profile Page and displays all personal information
	2	User chooses to edit profile	Prompts for which information to update

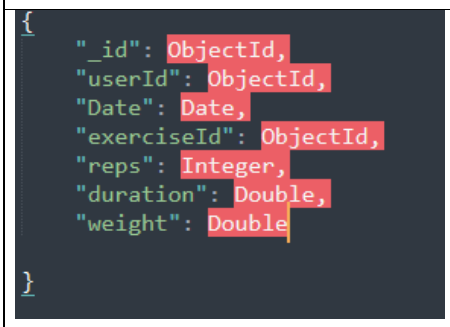
	3	User selects information to update and enters updated information	Prompts for confirmation about new information entered
	4	User confirms information is correct	Saves edited information and displays
	5	Repeat steps 3,4 until all information to be updated has been updated	
	6	User chooses to exit	Displays profile page
SubFlows:			
Alternate/Exceptional Flows: 5a: user choose not to edit information and chooses to exit			
Corresponding Mockup Interfaces: MU03			

4. Domain Class Diagram



5. Database

History
- **_id: ObjectId** - **userId: ObjectId** - Date: Date - ExerciseId: **ObjectId** - Weight: double - Reps: integer - Time: double

History Collection


Excercise
- **excerciseId: ObjectId** - **Name: String** - **Description: String** - **ImageLink: String**
Exercise Collection

```
{
    "excerciseId": ObjectId,
    "Name": String,
    "Description": String,
    "ImageLink": String
}
```

Profile

- ProfileId: ObjectId
- UserId: ObjectId
- Height: Double (ex 176cm)
- Weight: Double
- Age: Integer
- Sex: String
- FitnessLevel: String

Profile Collection

```
{
    "profileId":ObjectId,
    "UserId": ObjectId,
    "Height": Double,
    "Weight": Double,
    "Age": Integer,
    "Sex": String,
    "FitnessLevel": String
}
```

Workout

- _id: ObjectId
- userId: ObjectId
- name: String
- description: String
- exercises: Array

Workout Collection

```
{
    "_id": ObjectId,    // Workout ID
    "userId": ObjectId,  // Reference to the user who created the workout
    "name": String,
    "description": String,
    "exercises":
    [
      {
        "exerciseId": ObjectId,  // Reference to the exercise document
        "sets": Number,
        "reps": Number,
        "duration": Number,  // Time duration in seconds (if time-based exercise)
        "weight": Number,  // Weight in kilograms (if weight-based exercise)
        "notes": String
      }
    ]
  }
```

Commented [MP3]: the array of exercises in a workout is equivalent to a "WorkoutExercise"

Routine

- _id: ObjectId
- userId: ObjectId
- name: String
- description: String
- workouts: Array

Routine Collection

```
{
    "_id": ObjectId,    // Routine ID
    "userId": ObjectId,  // Reference to the user who created the routine
    "name": String,
    "description": String,
    "workouts":
    [
      {
        "workoutId": ObjectId,  // Reference to the workout document
        "dayOfWeek": String  // Day of the week for scheduling
      }
    ]
  }
```

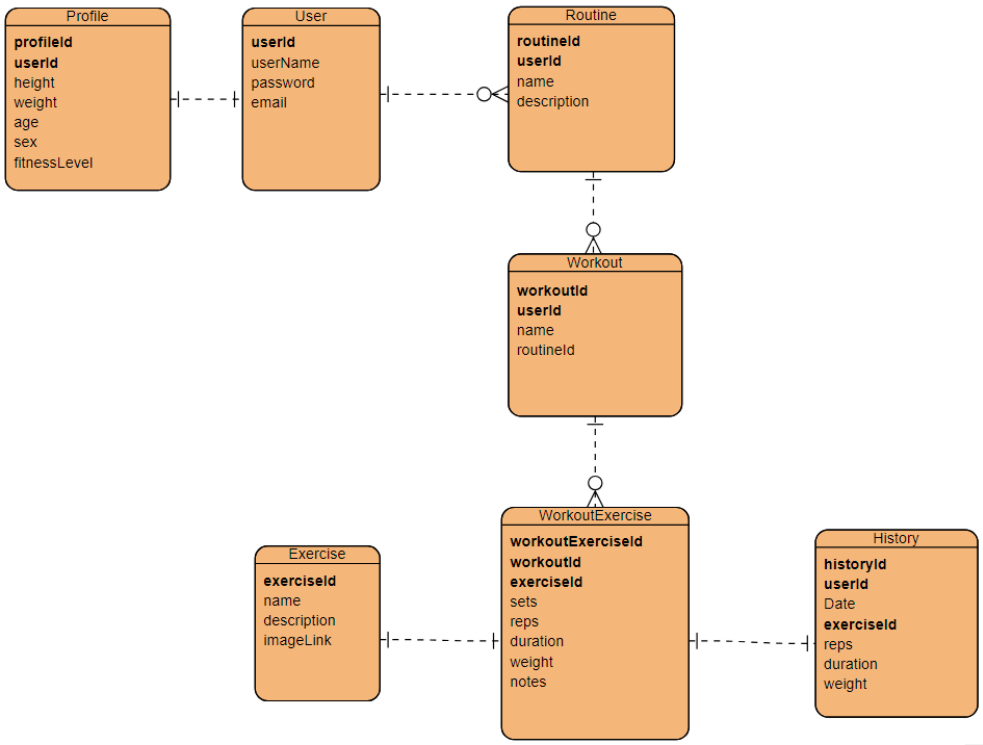
User
- _id: ObjectId - profileID: ObjectId - userName: String - password: String - email: String

User Collection

```
{
  "_id": ObjectId,    // User ID
  "profileId": ObjectId,  // Reference to the profile created by the user
  "userName": String,
  "password": String,
  "email": String
}
```

NoSQL ERD Diagram

Project Name



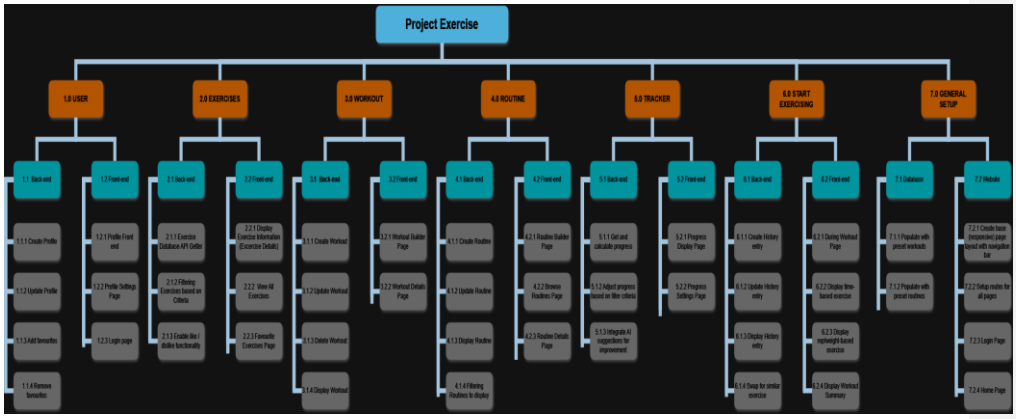
Entities Chart

Entity	PK	SK
User	User#<userId>	Profile#<userId>
Profile	Profile#<profileId>	User#<profileId>
Routine	User#<userId>	Routine#<routineId>
Workout	User#<userId>	Workout#<workoutId>
WorkoutExercise	Workout#<workoutId>	WorkoutExercise#<workoutExerciseId>
Exercise	Exercise#<exerciseId>	WorkoutExercise#<workoutExerciseId>
History	WorkoutExercise#<workoutExerciseId>	History#<historyId>

Project Name

6. Work Breakdown Structure (WBS)

6.1 Work Breakdown Structure



7. Milestones and Acceptance Criteria

7.1 User Milestone

Definition: This milestone involves creating the front and back-end for Users. For the back-end, this means creating a User class to manage account creation and logging in, a Profile class to manage and control users' profiles, and routes to interact with our database for CRUD operations and queries. For the front-end, this means allowing users to log in/out, view their profile information (such as age, height, weight, etc.), and adjust their settings, including making note of any injuries or muscle groups they would like to target or avoid.

Acceptance Criteria

- Back-end must be able to retrieve Users and their Profiles from the database
- Users must be able to authenticate and log in with Apple and Google services
- Profile Settings page must be able to perform all CRUD operations
- Home page must be able to accurately display and update user's favourites

7.2 Exercises Milestone

Definition: This milestone involves creating the front and back-end for Exercises. For the back-end, this means creating routes to interact with our external exercise API for CRUD operations and queries, enabling filtering based on criteria such as muscle groups, exercise type, etc., and enabling the like/dislike functionality to help with user exercise browsing and suggestions. For the front-end, this includes developing a user interface for displaying a selected exercise's details, allowing the users to view all exercises, as well as their preferred exercises.

Acceptance Criteria

- Back-end must be able to retrieve Exercises from the external API
- Users must be able to filter Exercises based on criteria including their Profile's settings
- Users must be able to like or dislike Exercises to help with future suggestions

7.3 Workouts Milestone

Definition: This milestone involves creating the front and back-end for Workouts. For the back-end, this means creating a Workout class to manage and control workouts, and routes to interact with our database for CRUD operations and queries. For the front-end, this means allowing users to browse workouts, view workout details page, and workout builder page.

Acceptance Criteria

- Back-end must be able to retrieve Workouts from the database
- Workout Builder must be able to perform all CRUD operations
- Workout Details page must show all workout specifications accurately

7.4 Routine Milestone

Definition: This milestone involves creating the front and back-end for Routines. For the back-end, this means creating a Routine class to manage and control routines, and routes to interact with our database for CRUD operations and queries. For the front-end, this means allowing users to browse their routines, show the routine details page, and perform CRUD operations with the routine builder page.

Acceptance Criteria

- Back-end must be able to retrieve Routines from the database
- Routine builder must be able to perform all CRUD operations
- Routine Details page must show all of the routine's specifications accurately

7.5 Tracker Milestone

Definition: This milestone involves the development of the progress tracking system. For the back-end this will require functionality to gather all required information and calculate and adjust the user progress data based on filtering criteria. AI will also be integrated to add suggestions based on the progress and user goals. On the front end, this is the development of a user interface for displaying this progress in a nice and easy to understand way such as line graph or some other sort of chart

Acceptance Criteria

- Back end must be capable of retrieving and calculating progress
- Must be functionality that allows progress calculations and display to be updated based on user filter criteria such as time period or exercise type (ex strength increase of certain muscle)
- AI must be able to analyze user progress and provide suggestions for improvement for adjust how to proceed based on user historical data and their performance goals
- Progress display page must provide visually appealing and easy to understand representation of user stats

7.6 Start Workout Milestone

Definition: This milestone involves creating the front and back-end support for starting a workout session. The back-end involves the creation and management of this workout history including the ability to create, update and display workout and exercise history in addition to having the ability to swap certain exercises for other similar ones. For the Front end this includes development of a user interface for the active workout sessions which will display exercise information based on what type of exercise it is

Acceptance Criteria

- System must be able to retrieve and display the users workout history after completing workout
- Functionality must be implemented to swap out exercises during a workout
- Front end must display time based exercises with a timer interface
- Front end must display rep/weight based exercises with input fields with validation
- API route set up to access and update the history

7.7 General Setup Milestone

Definition: This milestone involves creating the foundation of the webapp, the base page with Navbar where each other route will be adding onto, in addition to setting up the database. The Web app must be responsive and be able to adapt to different device screens especially phone screens and must also provide a navigation bar that allows users to easily navigate between pages.

Project Name

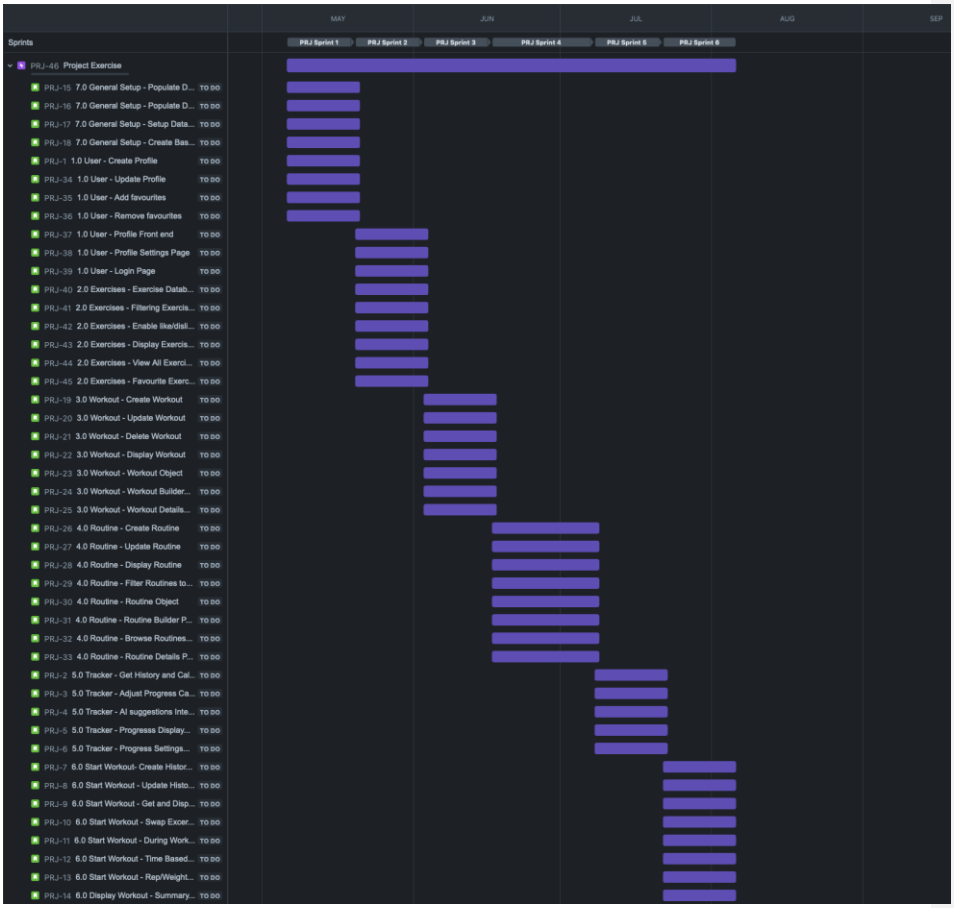
Acceptance Criteria:

- Base layout must be fully responsive ensuring compatibility on all different phone screen sizes
- Nav bar should be easily accessible and intuitive to use
- Cross browser and cross device compatibility
- Database schemas designed to effectively manage all data and manage the relationship of data
- Must be populated with comprehensive set of preset workouts and routines
- Security in place to protect from SQL injection if applicable
- Documentation of database schema and setup procedures provided for future maintenance and reference

8. Implementation Schedule

SEE: https://mperkons1.atlassian.net/jira/software/projects/PRJ/boards/6/backlog

Project Name



9. Client / Faculty Sign-off

Date: _____

Project Name

X_____.
Name of Client/Rep/Professor