Fit X
Fall 2023
Group 9

Aidan Monahan
Alexander Tapia Tellez
Ari Key
Chase Blackwell-Robertson
Christopher Ramos

10/20/2023

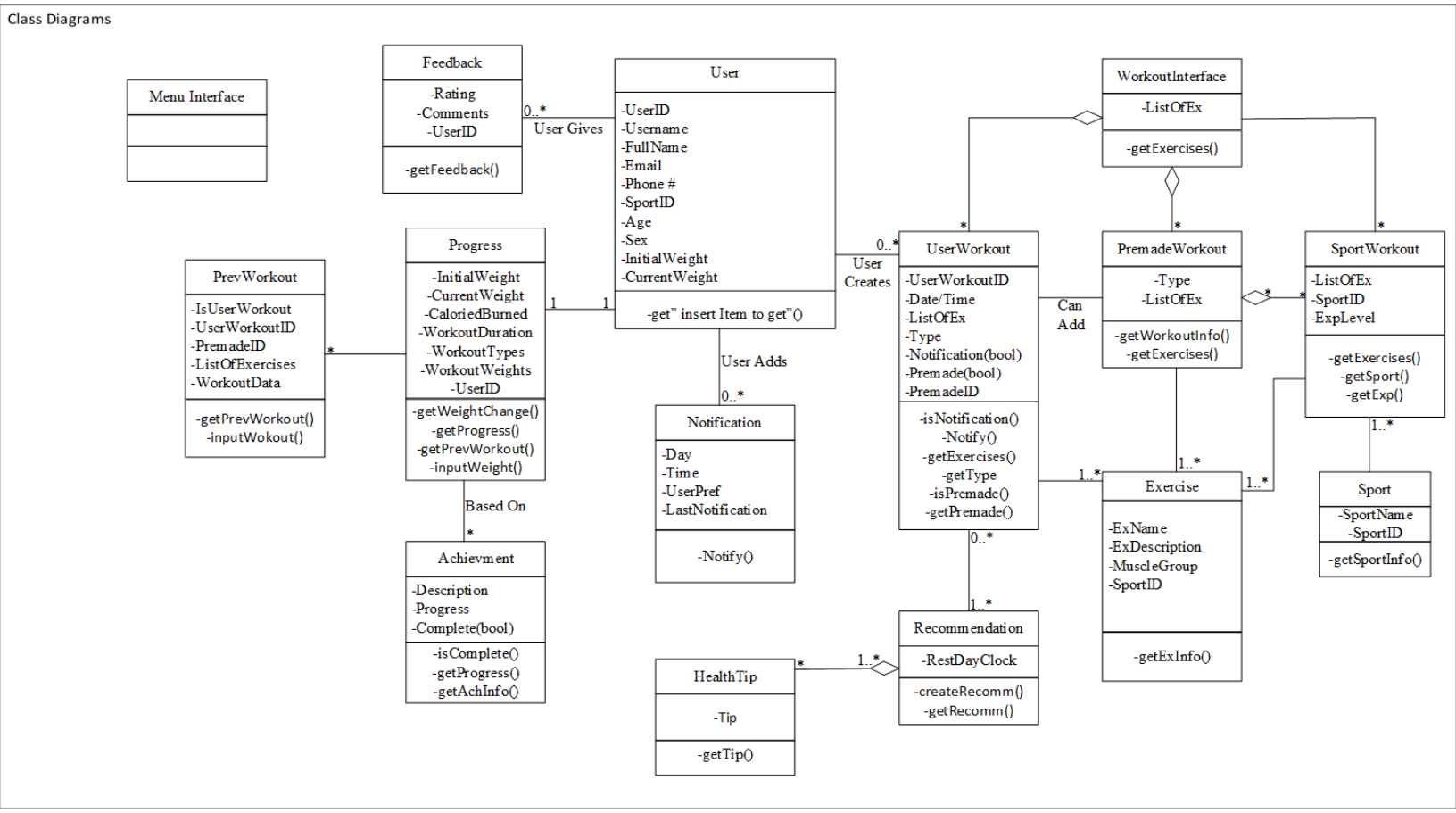
Planning Table

Name	Email	Task	Duration	Dependency	Due Date	Evaluation
Aidan	amonahan1@student.gsu.edu	Front end (Sport Workouts), Class diagram, and Refined problem statement	15hr	Account creation from Ari	10/20/2023	100%
Alex	atapiatellez1@student.gsu.edu	Backend Logic (Sport Specific Workouts) and Sequence Diagram(Sport Specific Workouts)	8hr	None	10/20/2023	100%
Ari	akey8@student.gsu.edu	Front end and logic(Account Creation)	8hr	None	10/20/2023	100%
Chase (Coordinator)	cblackwellrobertson1@student.gsu.edu	Backend Logic (Account Creation) and Sequence Diagram (Login/Registraion)	8hr	None	10/20/2023	100%
Chris	cramos14@student.gsu.edu	Test Plans	8hr	None	10/20/2023	100%

Refined Problem Statement:

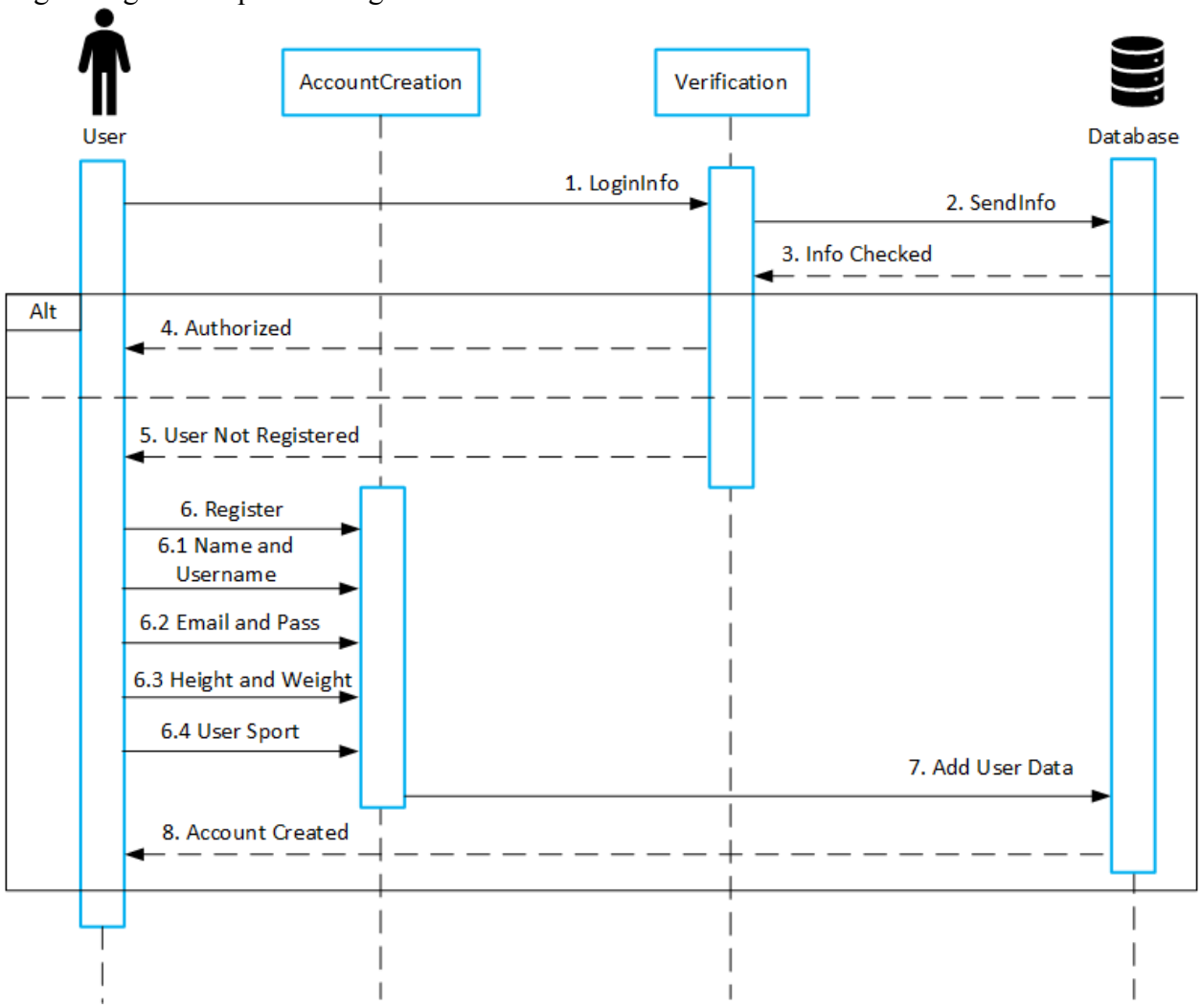
Our product is a fitness app for people who want to improve in their sport and/or their lifestyle. It is made to resolve the problem of people not having access to fitness content/resources or don't want to spend time researching various forms of workouts and/or healthy habits. The expensive alternative is having a private trainer which not many people can afford, which is why this app is worth creating as it allows the user to have a form of trainer in the palm of their hand. The objectives that we strive for are enabling easy access to workouts for specific sports and also allowing the regular person to use the app for their workout/wellness needs. We will ensure this by offering not only pre-planned workouts that will help people free up time trying to understand which exercises are best, but also by acting as a repository for a large number of exercises. To ensure anyone can easily pick and choose which exercises work best for them, FitX will also provide videos and/or descriptions on how to perform these exercises. We will also provide recommendations to people, to help maximize muscle growth and to ensure they are not overexerting themselves with lack of rest. It will send a notification to the users phone when they have a scheduled workout or when they need to take a rest day. Our competitors are similar apps such as FitOn, Adidas Training, and Burn. Fit and our approach to creating this app that is different from these is that our app will have specific workouts made for sports to ensure growth while also having a section for a regular person to use the app. What is interesting about this is that it is built on an Android system and uses various systems/modules such as a login system to create a simple yet efficient fitness app.

Class Diagrams:

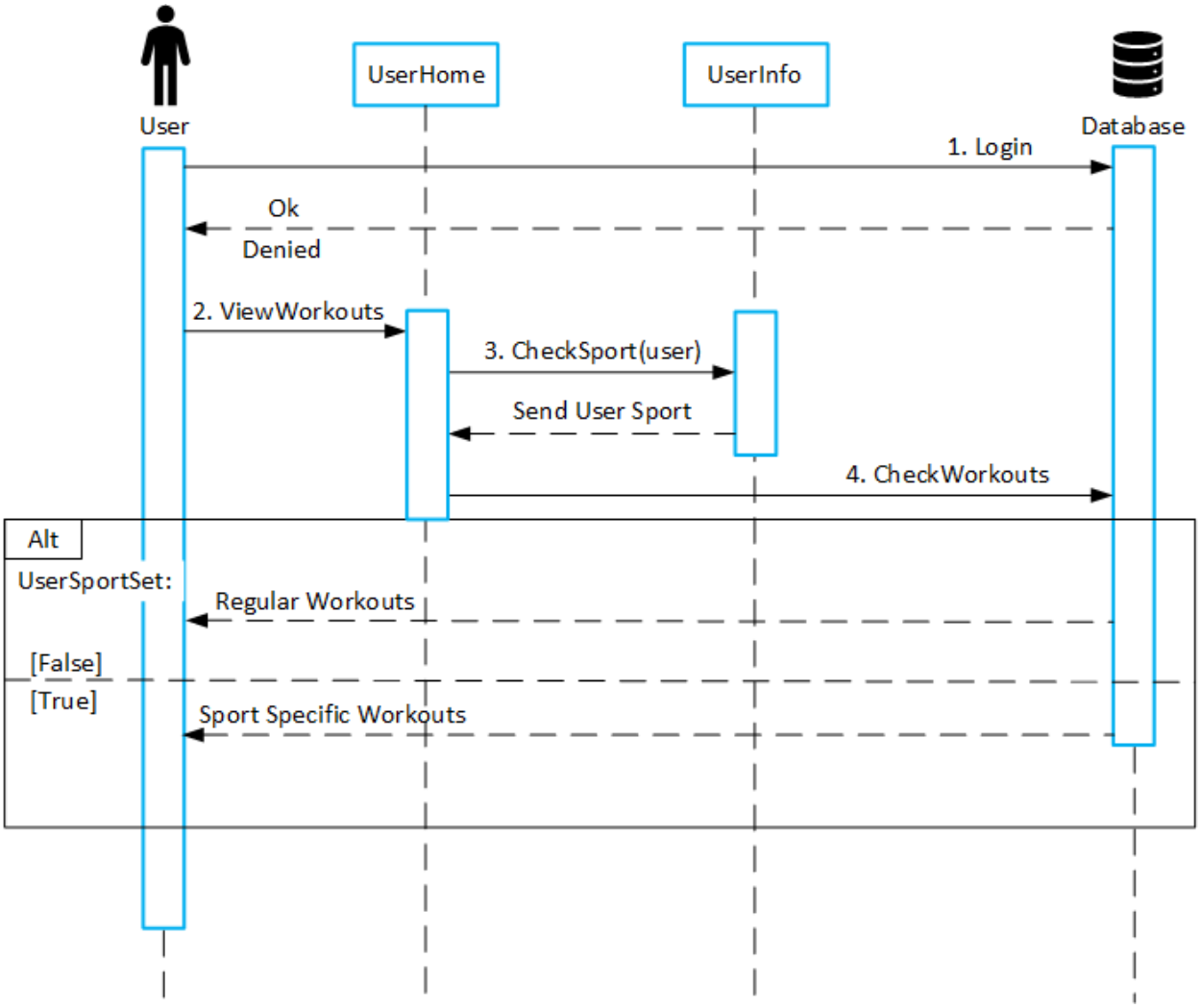


Behavioral Modeling:

Login/ Register Sequence Diagram:



Sport Workouts Sequence Diagram:



Implementation

Backend

We utilized Firebase Firestore for data storage and retrieval, focusing on two major use cases: "Account Creation" and "Workout for Specific Sports." This database utilizes a NoSQL structure.

Database Design:

Authentication User Databse:

- This contains the User's Emal, Password, and they are assigned an ID.

User Collection:

- The User collection stores user information, including age, experience level, first name, last name, height, weight, and sport ID. User ID is attached to the authenication user database.

Sports Collection:

- The Sports collection contains a list of sports, each assigned a unique sport ID.

- Workouts Collection:

  This collection is stored within the sports collection. Workouts are stored in this collection with details like the number of reps and exercise ID. Workouts are associated with specific sports.

Exercises Collection:

- The Exercises collection holds exercise information, including exercise name, exercise ID, and muscle group. It relates to workouts.

Frontend and Logic

Two Use Cases Chosen:

- Use Case 4: Account Creation
- Use Case 1: Workouts for Specific Sports
+ Use Case 6: Find workout videos

We crafted FitX's front-end and logic using a combination of Kotlin and XML. Leveraging Android Studio as our integrated development environment, we've harnessed its capabilities to seamlessly run and test our app on virtual Android devices, enabling us to evaluate functionality in real-time. Our logic is meticulously implemented in Kotlin, while our XML code takes care of styling, layout, and visual presentation.

Steps to compile FitX code base:

1. Download or clone repo from our GitHub
2. Connect an Android device to your computer using a USB-to-USB-C cable.
3. Ensure that the device has developer options enabled for app installation
4. Open your preferred terminal.
5. Navigate to the downloaded repo location.
6. Enter the FitX folder.
7. Run the appropriate build script: ./script/build-script.sh for macOS, or .ps1 for Windows.
8. The build and install process will initiate, and FitX will be installed on the connected Android device.
9. 

This is what the app should look like
10. Now you should be able to successful launch FitX

Testing

Use case 1: work out for specific sports

Test case 1.1:

Description: User will choose a specific sport during account creation

Test input: specific sport

Expected result: app displays workouts related to sport

Dependence:

Initialization: there is a data base that stores sports

Test steps: submit specific sport

Test case 1.2:

Description: User will not choose a specific sport during account creation

Test input: don’t choose specific sport

Expected result: app displays generic workouts

Dependence:

Initialization: there is a data base that stores sports

Test steps: user doesn’t choose sport

Use case 3: planning workout

Test case 3.1:

Description: user will select muscle group for workout plan

Test input: select muscle group

Expected result: app will display workouts related to muscle group for user to add to workout plan

Dependence:

Initialization: there is a data base for workout/muscle group

Test steps: user selects muscles groups

Test case 3.2:

Description: user will not select muscle group for workout plan

Test input: dont select muscle group

Expected result: app will display generic workouts for user to add to workout plan

Dependence:

Initialization: there is a data base for workout/muscle group

Test steps: user doesn't selects muscles groups

Test case 3.3:

Description: user will select days for workout plan

Test input: select days

Expected result: app will display workout plan on main page on given day

Dependence:

Initialization: work out plan is created

Test steps: user selects days

Test case 3.4:

Description: user will not select days for workout plan

Test input: don't select days

Expected result: app will not display workout plan on main page on given day, it will be hidden until user selects it from workout catalogue.

Dependence:

Initialization: work out plan is created

Test steps: user doesn't select days

Use case 4: account creation.

Test case 4.1:

Description: user will make an account

Test input: user email, password, user information

Expected result: account will be created from email and password.

Dependence:

Initialization: user data base is present

Test steps: user inputs email, user inputs password, verify user is not already made

Test case 4.2:

Description: user will try to make account will already used information

Test input: invalid user email, password, user information

Expected result: account will NOT be created from email and password since it already exists

Dependence:

Initialization: user data base is present

Test steps: user inputs email, user inputs password, verify user is not already made

Test case 4.3:

Description: user will input password with length of 8 or more in account creation

Test input: password with length or 8 or more

Expected result: account will be created.

Dependence:

Initialization: user data base is present

Test steps: user inputs valid password

Test case 4.4:

Description: user will input password with length less than 8

Test input: password with length less than 8

Expected result: app will prompt user to change password, account will not be created

Dependence:

Initialization: user data base is present

Test steps: user inputs invalid password / verify user inputs valid password

Use case 6: find workout video

Test case 6.1

Description: user will be shown workout videos which will show the user how to perform the given workout based on workout they are trying to perform

Test input: workout

Expected result: video related to workout is presented to user

Dependence:

Initialization: workout data base

Test steps: user inputs workout

Test case 6.2

Description: user will be not be shown workout videos since they didn't choose a workout

Test input: no workout

Expected result: no video will be shown to the user

Dependence:

Initialization: workout data base

Test steps: user inputs no workout

Use case 7: select workout.goo

Test case 7.1

Description: user will be able to select exercises and workouts based on specifications

Test input: muscle group, sport, type of exercise

Expected result: user is presented will a list of exercises/workouts related to the information provided (muscle group, sport, type of exercise)

Dependence:

Initialization: workout data base , muscle data base, sport data base

Test steps: user inputs workout, muscle group, exercise

Test case 7.2

Description: user will be able to select from generic exercise/workouts

Test input: no input

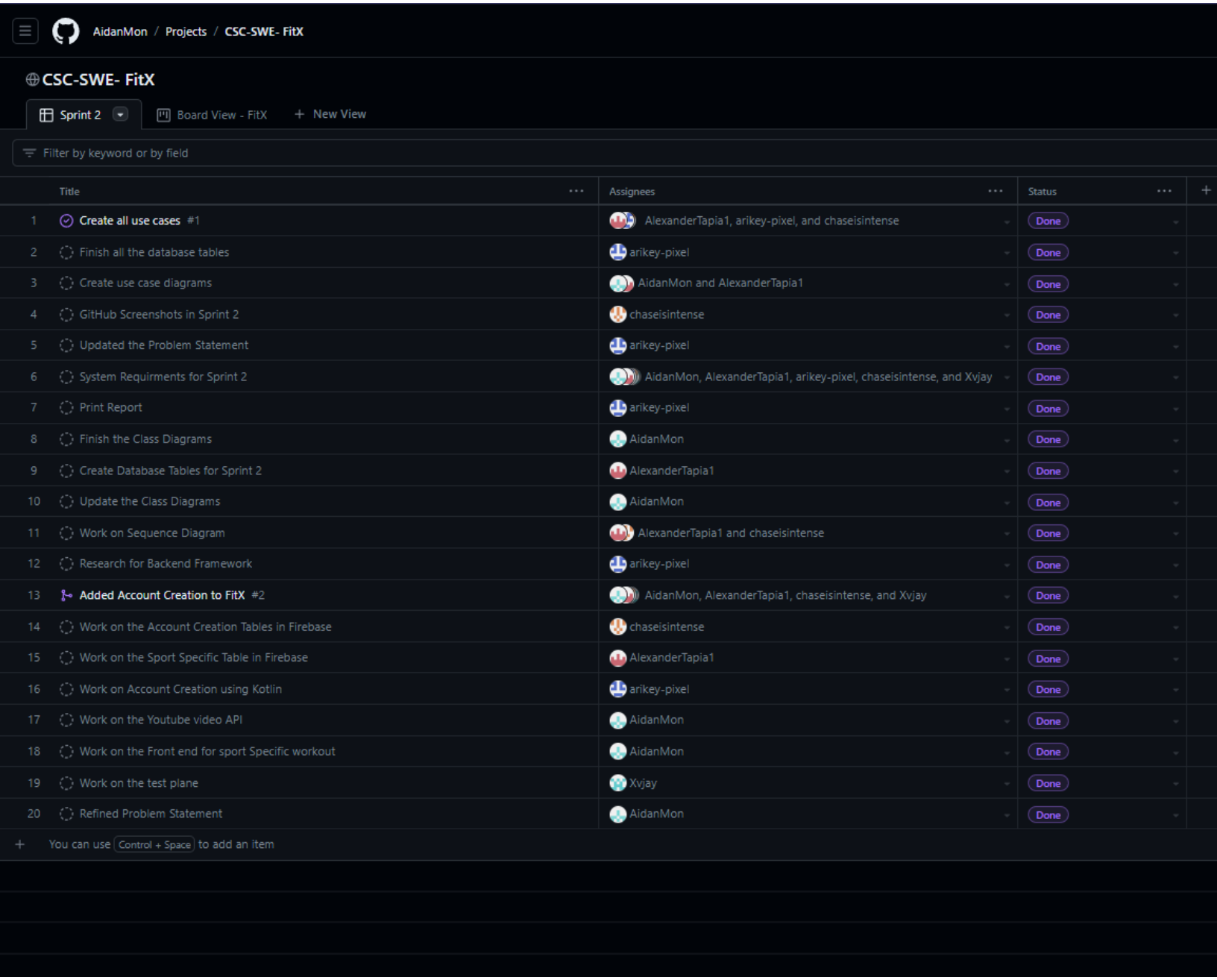
Expected result: user is presented will a generic list of exercise/workouts

Dependence:

Initialization: workout data base , muscle data base, sport data base

Test steps: user inputs no workout or muscle group or exercise

Project Screenshot:



	Title	Assignees	Status
1	Create all use cases #1	AlexanderTapia1, arikey-pixel, and chaseisintense	Done
2	Finish all the database tables	arikey-pixel	Done
3	Create use case diagrams	AidanMon and AlexanderTapia1	Done
4	GitHub Screenshots in Sprint 2	chaseisintense	Done
5	Updated the Problem Statement	arikey-pixel	Done
6	System Requirments for Sprint 2	AidanMon, AlexanderTapia1, arikey-pixel, chaseisintense, and Xvjay	Done
7	Print Report	arikey-pixel	Done
8	Finish the Class Diagrams	AidanMon	Done
9	Create Database Tables for Sprint 2	AlexanderTapia1	Done
10	Update the Class Diagrams	AidanMon	Done
11	Work on Sequence Diagram	AlexanderTapia1 and chaseisintense	Done
12	Research for Backend Framework	arikey-pixel	Done
13	Added Account Creation to FitX #2	AidanMon, AlexanderTapia1, chaseisintense, and Xvjay	Done
14	Work on the Account Creation Tables in Firebase	chaseisintense	Done
15	Work on the Sport Specific Table in Firebase	AlexanderTapia1	Done
16	Work on Account Creation using Kotlin	arikey-pixel	Done
17	Work on the Youtube video API	AidanMon	Done
18	Work on the Front end for sport Specific workout	AidanMon	Done
19	Work on the test plane	Xvjay	Done
20	Refined Problem Statement	AidanMon	Done

Link to our Github: https://github.com/AidanMon/CSC-SWE-FitX
Steps to compile FitX code base:

11. Download or clone repo from our GitHub
12. Connect an Android device to your computer using a USB-to-USB-C cable.
13. Ensure that the device has developer options enabled for app installation
14. Open your preferred terminal.
15. Navigate to the downloaded repo location.
16. Enter the FitX folder.
17. Run the appropriate build script: ./script/build-script.sh for macOS, or .ps1 for Windows.
18. The build and install process will initiate, and FitX will be installed on the connected Android device.
19. 

This is what the app should look like
20. Now you should be able to successful launch FitX