

### **PROJECT COMPONENT 3**

#### **GROUP- DNW**

<b><u>GROUP MEMBER</u></b>	<b><u>STUDENT ID.</u></b>
<b><u>DAVID MULINDWA</u></b>	<b><u>306555m</u></b>
<b><u>MOHAMMED NABIL</u></b>	<b><u>0307020n</u></b>
<b><u>WELTON TOME</u></b>	<b><u>0303719t</u></b>

#### **Scope Refinement from Component 1 and 2**

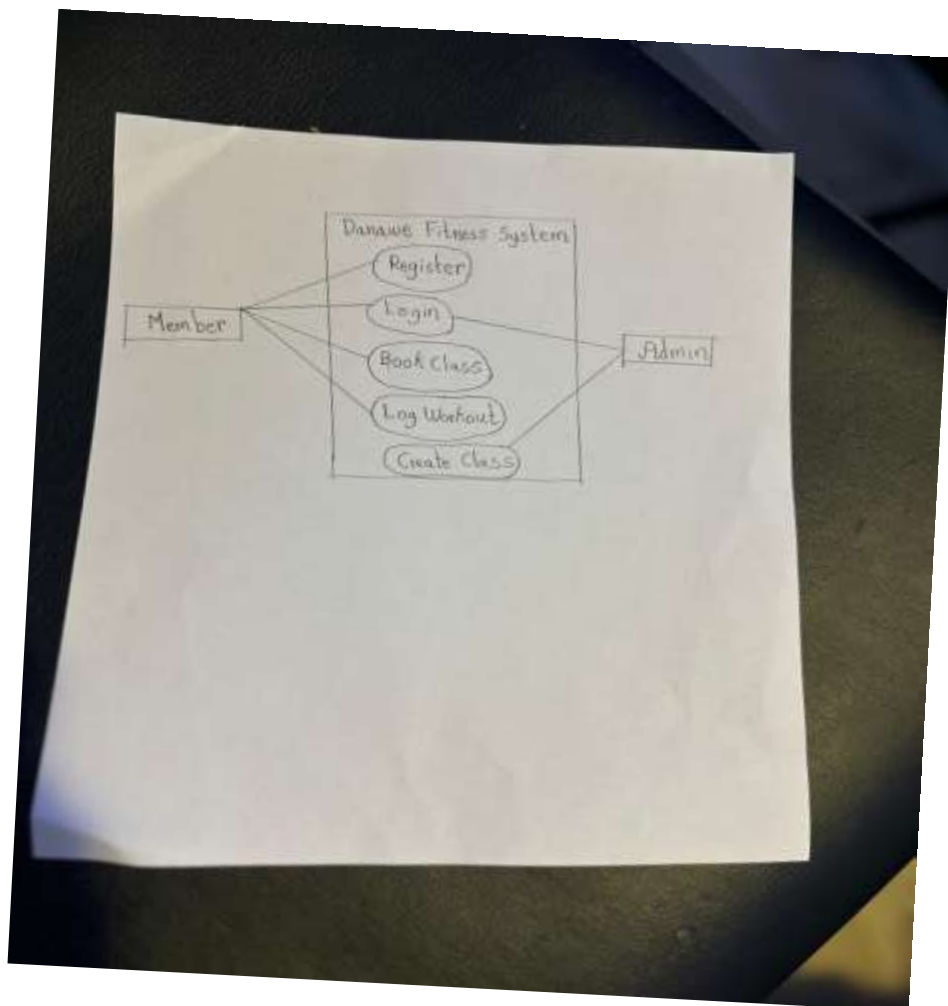
In Project Components 1 and 2, we proposed a larger feature set for the Danawe Fitness system, including membership payments, trainer tools, notifications and reminders, and admin analytics reports.

During implementation, we decided to reduce the scope to a smaller “Danawe Fitness Management System” version that we could realistically build and test within the course timeline. The final implemented system focuses on:

- User registration and login (members + admin).
- Admin class management (create and delete classes with date, time, capacity).
- Member class booking (view classes, book a class, cancel a booking, prevent overbooking).
- Member workout logging (log workouts and view workout history).
- Data persistence using browser local Storage.

Features such as third party payment integration, trainer tools, notifications/reminders, and analytics reports are considered out of scope for this implementation and are left as possible future work.

### Question 1



## Question 2

### **1. Define “units” vs. “non-units”**

Our project defines a unit as a single JavaScript function or small feature, considering non-units as the larger features which require multiple functions to operate together.

#### **Units examples**

- register logic (saving a new user)
- login logic (checking username/password)
- createClass logic
- bookClass logic
- saveWorkout logic

#### **Non-units examples**

- Member Dashboard (view classes, bookings, and workouts)
- Admin Dashboard (create and manage classes)
- Data persistence (reloading data from local Storage)

### **2. Types of testing performed**

Integration testing process verifies that different parts work together, e.g. login, dashboard, create class and member view.

System testing includes testing the full app through their web browser interface. We don't have a dedicated QA team.

### **3. Extent of testing**

We prioritized the most important scenarios that is:

- Users can successfully register and log in.
- Admin can create and delete classes.
- Members can book and cancel classes correctly, and overbooking is prevented.
- Members can log workouts and see their history.
- All this data remains after refreshing or reopening the page.

Our fundamental test cases include:

- The system allows members to register and log in.
- The system allows admins to create and delete classes.
- The system allows members to schedule classes and make cancellations and prevents it from overbooking
- . The system allows members to access their log workouts and see their history.
- The system maintains all stored data after refreshing or reopening the page.

#### **4. Documentation**

Our testing will be documented as follows:

- A test case table listing included in the document shows each test ID, the feature under test, test steps, and expected result.
- Notes on if all tests passed or failed when we actually run them.
- No special testing tools process will occur without any specialized tools because we will perform manual testing in a web browser to document results in our report or spreadsheet.

#### **4. Documentation**

Our testing will be documented as follows:

- A test case table listing included in the document shows each test ID, the feature under test, test steps, and expected result.
- Notes on whether each test passed or failed when we actually run them.
- No special testing tools will be used. we will test manually in a web browser and record results in our report or a spreadsheet.

## 5. Input sources for tests

We designed our tests based on:

- The use cases and use case diagram we created for Component 3 that includes Register, Login, Create Class, Book Class and Log Workout.
- The functional requirements we selected for the simplified “Danawe Fitness Lite” version of the system.
- The actual implemented behaviour in our HTML/JavaScript prototype.

## 6. Who will test

All testing will be performed by the members of our project group and we will share the test cases and split them up so that each person runs some of the tests as either member or **Admin**.

There is no separate QA or any other third parties.

## 7. Resources

Resources needed:

- One laptop for every group member.
- A modern web browser.
- Our current HTML file for the Danawe Fitness Management System.

## 8. Metrics to be collected

We will track only a few simple metrics that is:

- Number of test cases passed or failed.
- Number of bugs found and fixed during testing.
- Which main features had problems for example login, booking, workout logging

### TEST CASE TABLE

Test ID	What are we testing?	How we test it	The Expected Result
T1	Register new member	After opening the app we proceeded to Register(Member) and entered a new member,s user name and password and then click Register	Message says registration successful. New user is saved and can later log in.
T2	Registration duplicate	After doing T1 we could try registering the member again with the same credentials.	The system will display an error "Username already exists." And no second user with that name is created.
T3	Member login	The member logs in with where credentials from T2	The login/register section will hide and we will get to the members dashboard.
T4	Admin login	We login as an administrator with admin as the user name and admin123 as the password.	The Admin Dashboard will be displayed.
T5	Admin creates class	After logging in as an Admin, go to Create Class and enter a valid name, date, time and positive capacity	You will get to see the created class in the All classes in admin's Dashboard and Available classes when a members logs in.
T6	Member books a class	To test this ensure that at least one class exists from T5, afterwards a member can decide to book a that class.	Confirmation message shown. Class appears under "My Bookings" and the capacity display increases by 1.
T7	Member cancels booking	After T6 in my Bookings, click cancel Booking for that class	Booking disappears from "My Bookings", and capacity in the Available Classes list decrements by 1.
T8	Log workout and history	Log in as member then Fill Log Workout form with date, type, positive duration, and notes then Click <b>Save Workout</b>	Workout entry is added under "Workout History" and the form fields are cleared.
T9	Data persistence	After creating at least one class, one booking, and one workout, Refresh the browser page and then Log in again as same user/admin.	The previously created classes, bookings, and workouts are still visible meaning that data was correctly loaded from the local Storage after refresh.



