

**IT 309 SOFTWARE ENGINEERING**

PROJECT DOCUMENTATION

PROJECT NAME

Prepared by:

**Hamza Bakaran**

**Ademir Hebibović**

Proposed to:

**Nermina Durmić, Assist. Prof. Dr.**

**Aldin Kovačević, Teaching Assistant**

21.06.2023

Table of Contents

[1. Introduction 3](#__RefHeading___Toc14999_2824873067)

[1.1. About the Project 3](#__RefHeading___Toc17416_1970057414)

[1.2. Project Functionalities and Screenshot 3](#__RefHeading___Toc15001_2824873067)

[2. Project Structure 8](#__RefHeading___Toc15009_2824873067)

[2.1. Technologies 8](#__RefHeading___Toc13717_546472535)

[2.2. Database Entities 8](#__RefHeading___Toc15011_2824873067)

[2.3. Design Patterns 9](#__RefHeading___Toc15013_2824873067)

[2.4. Tests 9](#__RefHeading___Toc15015_2824873067)

[3. Conclusion 10](#__RefHeading___Toc15017_2824873067)

# 1. Introduction

Welcome to the documentation for Gym Management System This document serves as a guide to understand and use the functionalities of the project. It provides an overview of the project, its purpose, features.

## 1.1. About the Project

The gym management system is a software application designed to streamline and enhance the management of gym operations. It provides a comprehensive solution for gym owners/administrators to efficiently manage various aspects of their gym, while also offering convenient features for gym members.

The primary goal of the gym management system is to simplify the process of managing gym memberships, tracking membership validity, and maintaining records of previous membership plans. With this system, gym members can easily log in to their accounts and access information related to their memberships, including expiry dates and a history of past membership plans.

For gym owners/administrators, the system offers extensive functionality to manage users, memberships, and employees. They can create and update user accounts, assign and modify membership plans, and track various metrics through a convenient dashboard. The dashboard provides essential insights and metrics that enable gym owners to make informed decisions regarding their business operations.

To access the deployed gym management system, please visit <https://gymhotelbravo.xyz/>.

**Admin: test credidentials**:

email: githubtestadmin@gmail. Com

password: 1234

**User: test credidentials**:

email: githubtestuser@gmail. Com

password: 1234

## 1.2. Project Functionalities and Screenshot

The gym management system offers a range of functionalities to both gym administrators (admins) and gym members (users). Here are the main features of the application:

Admin functionality:

Admin dashboard: the dashboard provides an overview of essential gym metrics, such as the total number of users, the number of active users, total earnings for the last month, the number of employees, and recently sold memberships. This consolidated view helps administrators make informed decisions and monitor the overall performance of the gym.

Member registration: the system allows new members to register for an account by providing necessary details such as name, contact information, and address. The registration process is designed to be user-friendly, secure, and streamlined.

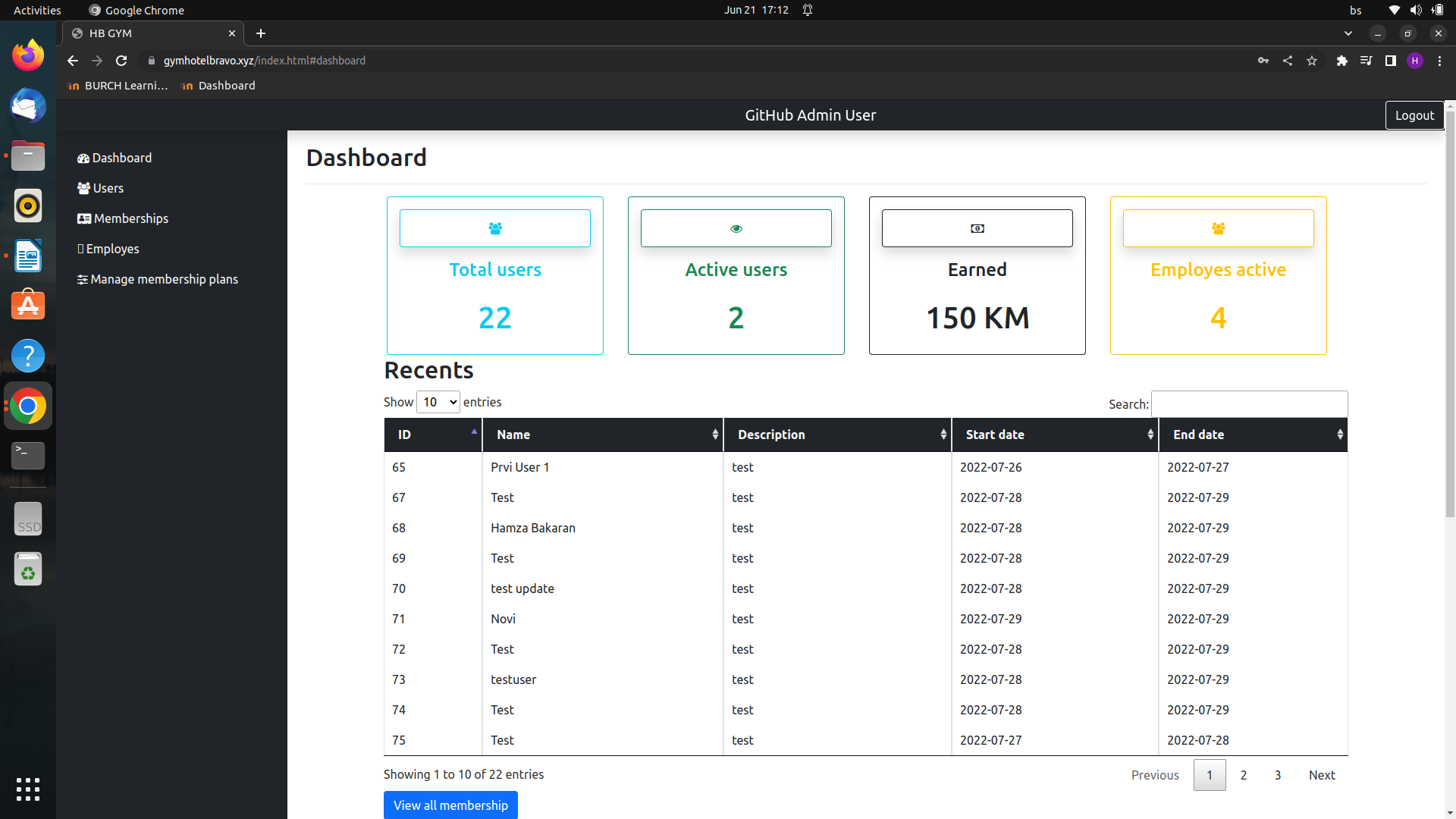
Membership management: administrators have the ability to manage existing memberships efficiently. This includes activating or deactivating memberships, upgrading or downgrading membership plans, extending or canceling memberships, and handling other related tasks. The system automates these processes, reducing administrative workload and ensuring smooth membership management.

User functionality:

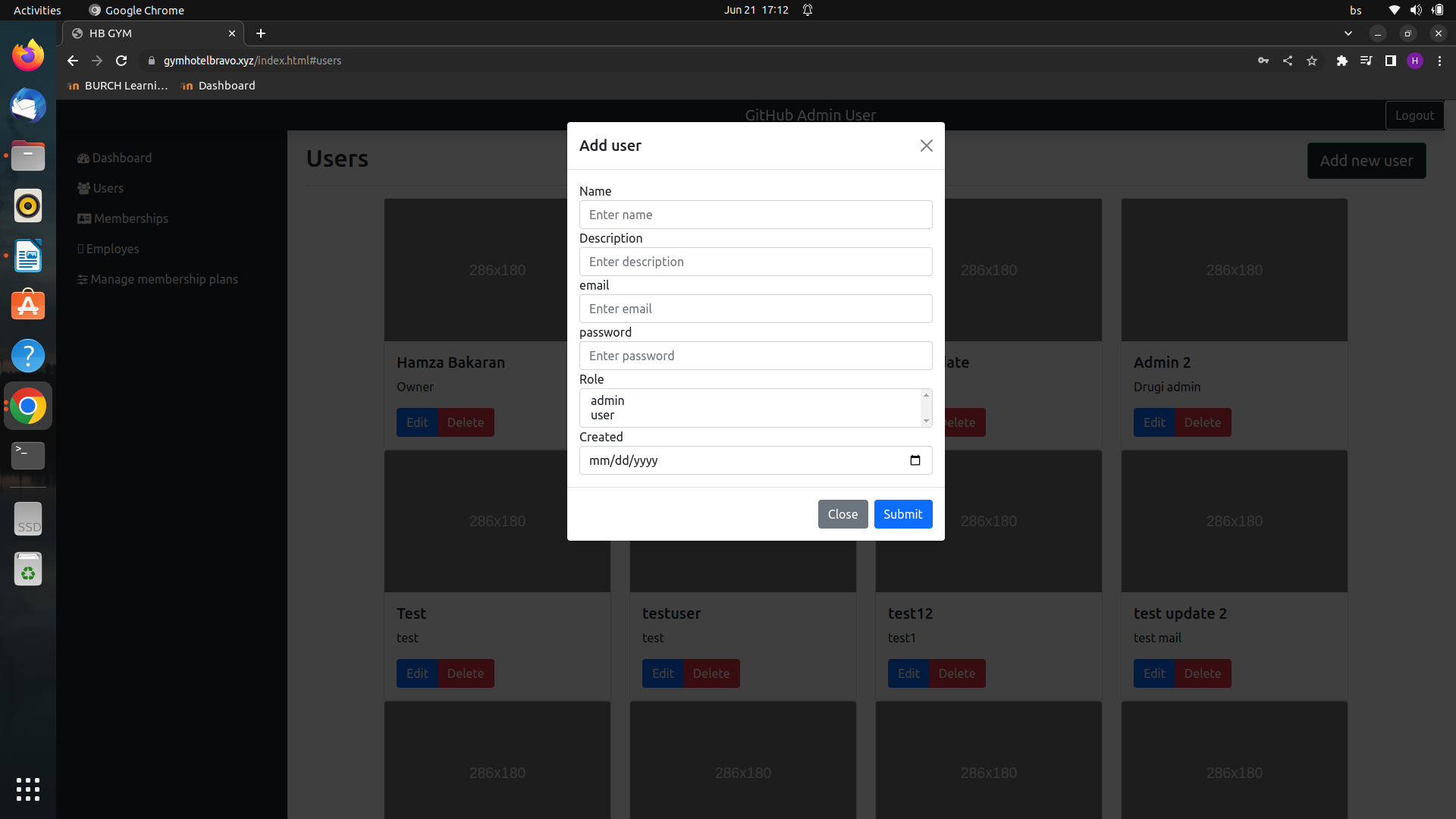
Membership validity: gym members can easily view the validity of their membership. This feature provides information about the membership plan, its validity period, and a log of previous memberships. Members can track their progress, plan their workouts accordingly, and ensure timely membership renewals.

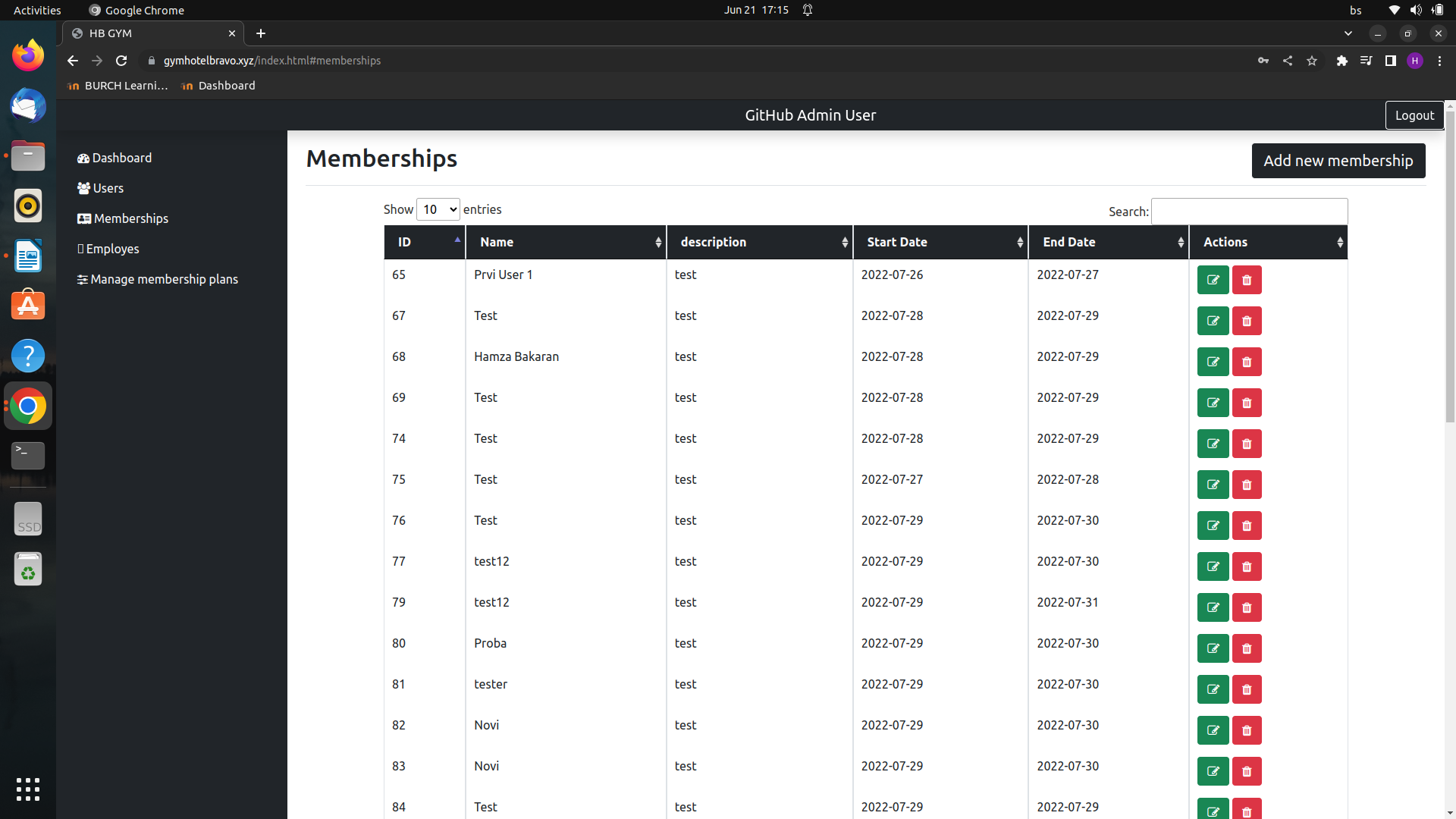
Below are a few screenshots showcasing the user interface and functionalities of the gym management system:

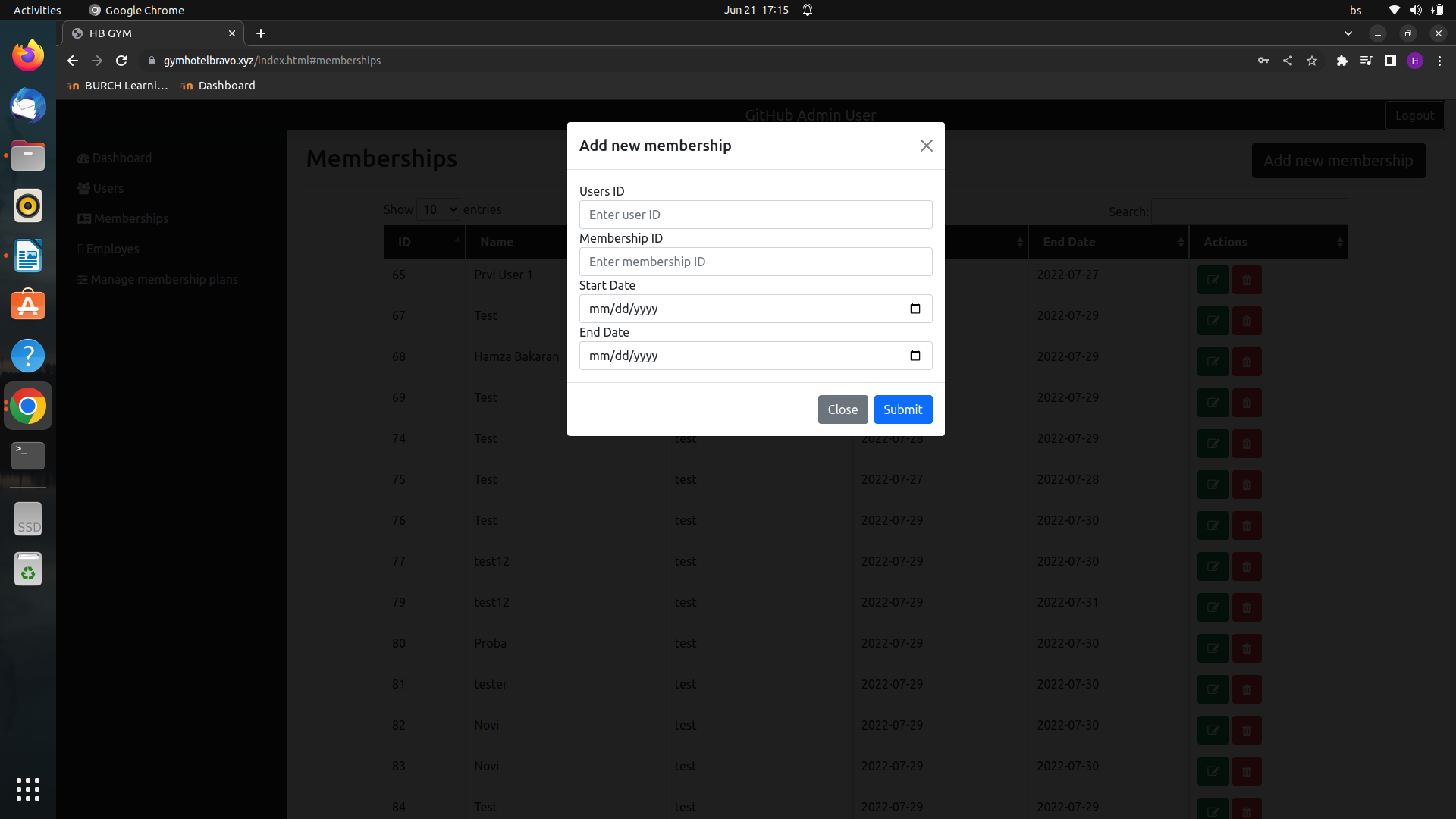
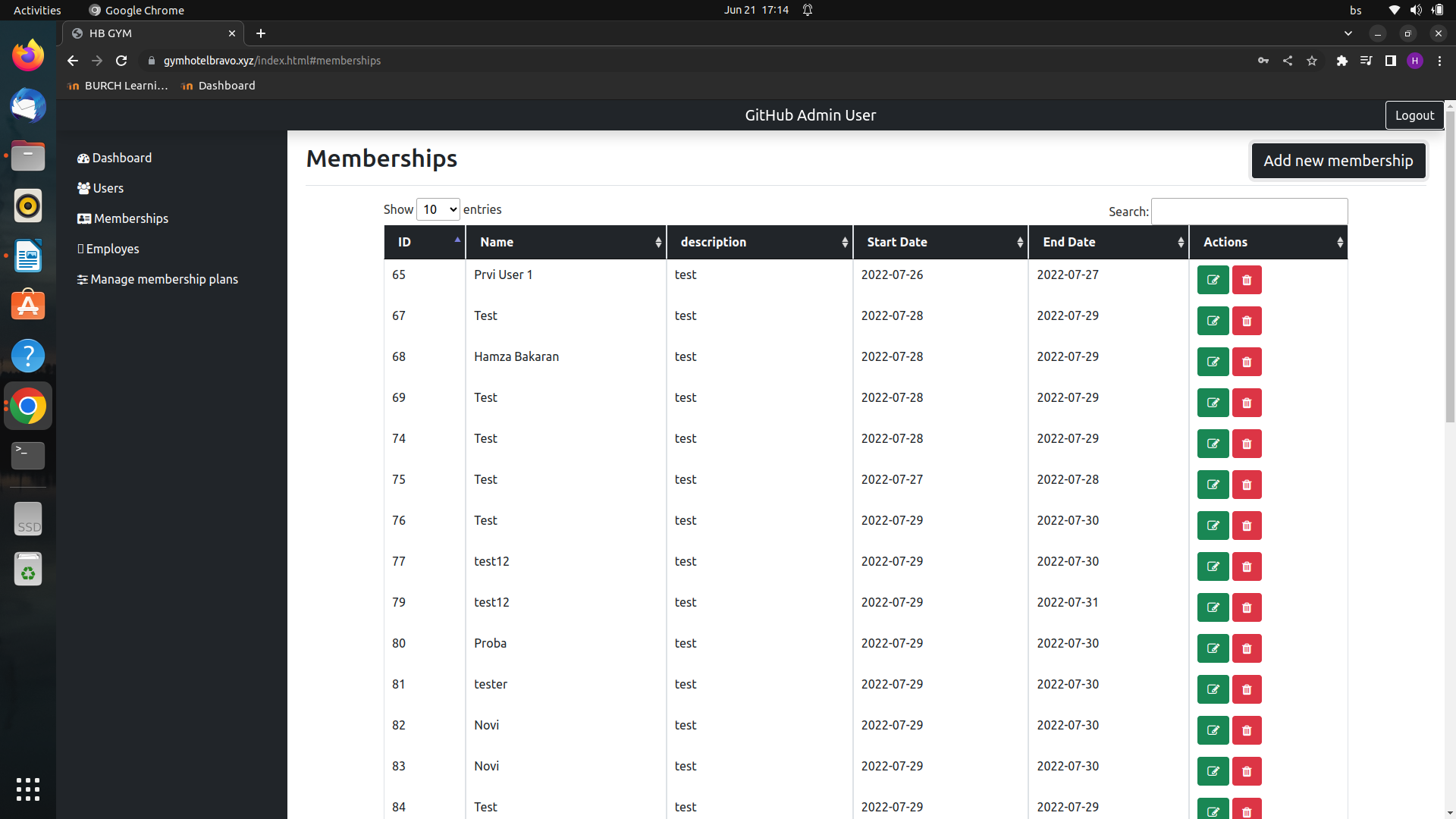
1.**Admin dashboard**



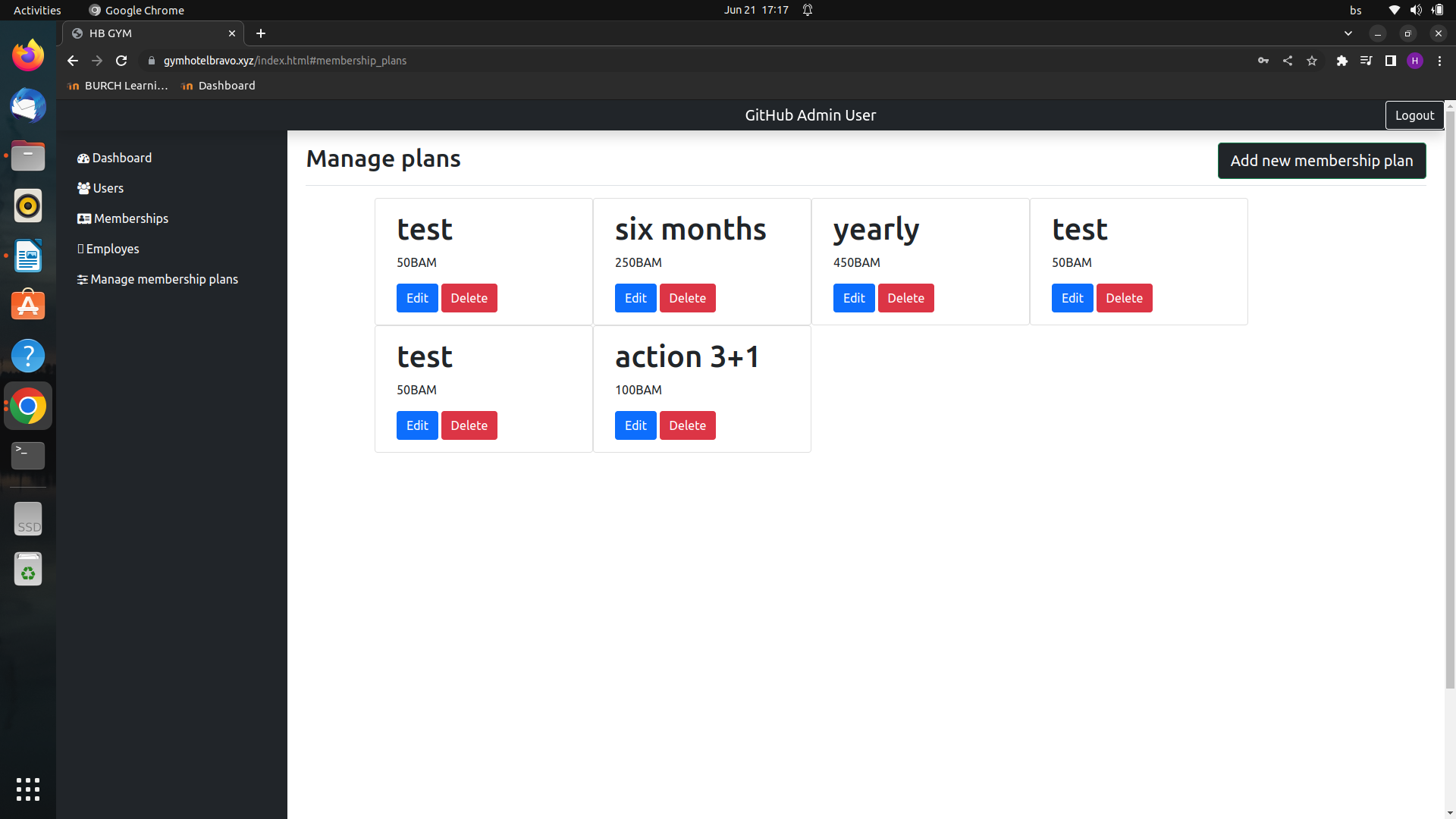
2.**Member registration**

3.**Membership management**

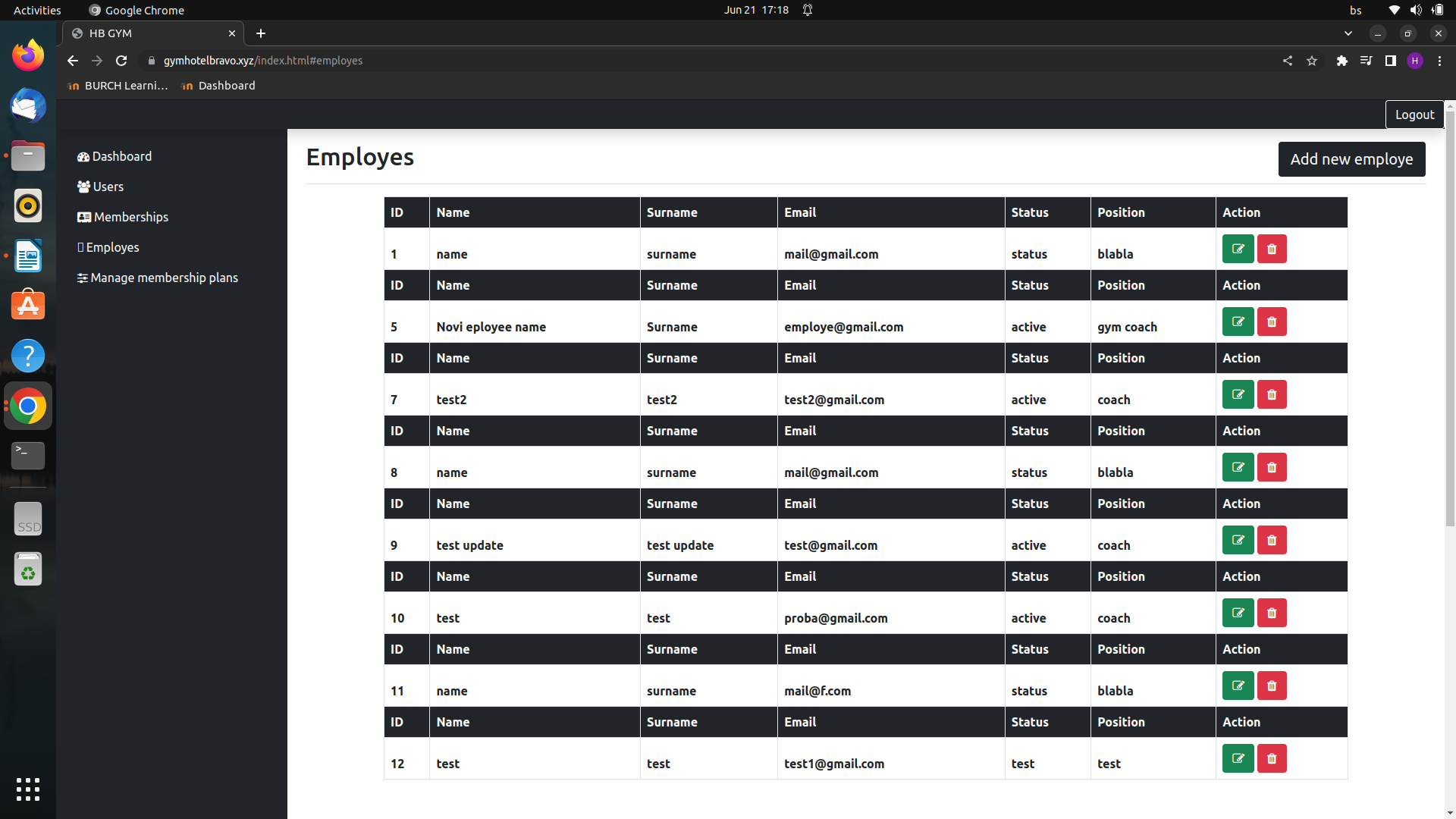




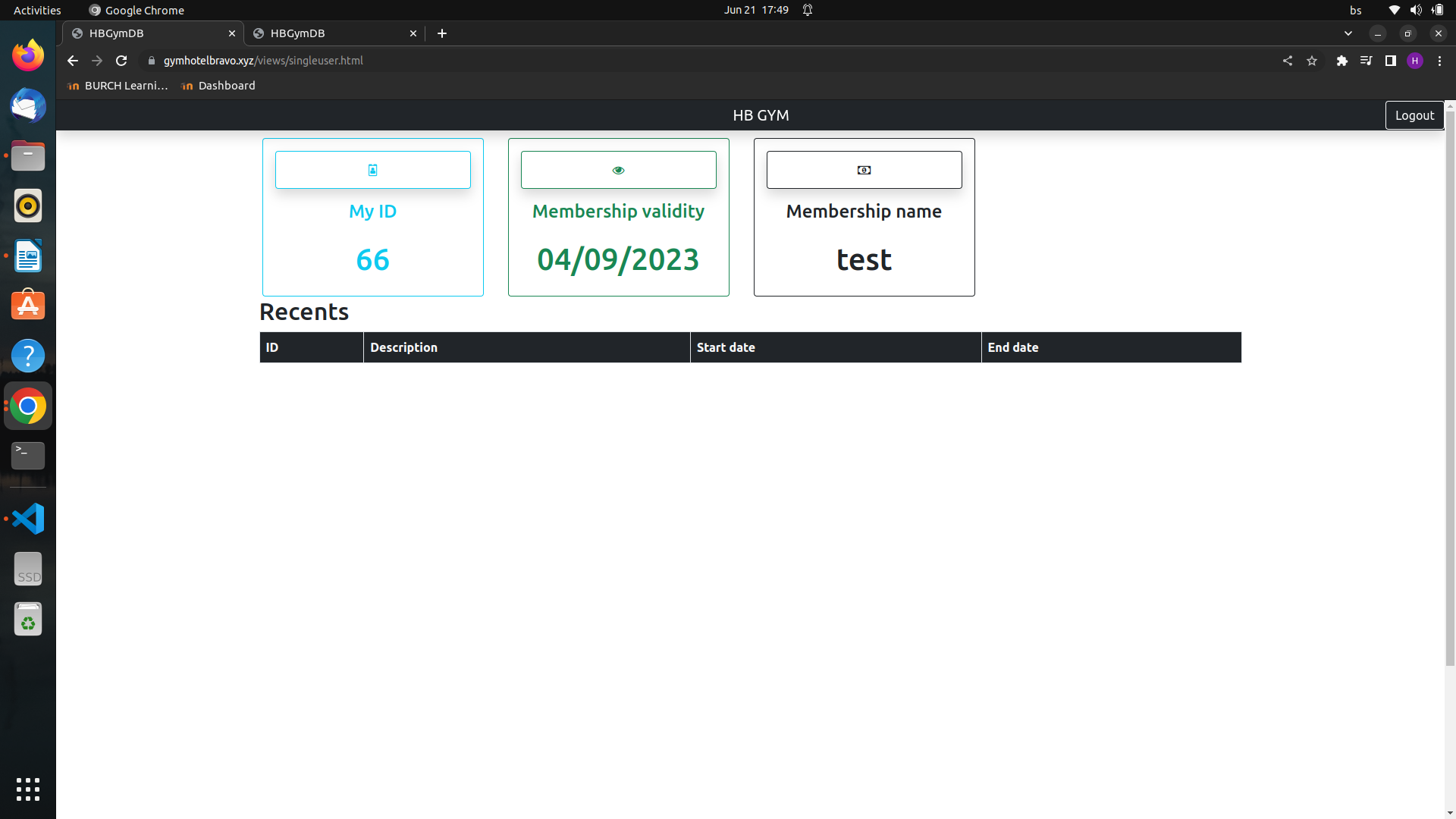
4.**Plans management**



5.**Employe management**



6.**User Dashboard**



# 2. Project Structure

## 2.1. Technologies

In the project, the following technologies were used:

Backend:

PHP (Flight PHP) - a lightweight framework for building restful apis and web applications.

Mysql (Mariadb) - a popular open-source relational database management system.

Frontend:

HTML - the standard markup language for creating web pages.

CSS - used for styling the html elements.

Bootstrap - a css framework that provides pre-designed components and styles for building responsive web interfaces.

Javascript - a programming language used for client-side scripting.

Jquery - a javascript library that simplifies html document traversal, event handling, and ajax interactions.

Middleware:

jwt token - a json web token used for authentication and authorization purposes.

Coding standard:

PSR-12 - a coding standard for php that defines coding style and structure guidelines. It was used for the backend part of the project.

## 2.2. Database Entities

The list of tables/entities in the database schema includes:

Employes: represents employees in the gym with columns such as id, name, surname, email, status, and position.

Membership: stores information about gym membership options with columns like id, description, and price.

Payments: tracks the payments made by users with columns such as id, user\_membership\_id, and status.

Users: contains user information with columns like id, name, description, email, password, role, and created.

Users\_membership: represents the relationship between users and their memberships, including columns like id, user\_id, membership\_id, start\_date, and end\_date.

## 2.3. Design Patterns

**DAO** (data access object) pattern: the Dao pattern is evident in the dao classes (EmployeDao, MembershipDao, UserDao, UsermeMbershipdao). These classes encapsulate the data access logic and provide methods to interact with the corresponding database tables.

**Singleton pattern**: the config class utilizes static methods to provide configuration values. Although it doesn't strictly follow the traditional singleton pattern, it demonstrates a similar concept by providing a single access point for retrieving configuration values throughout the application.

**Template method pattern:** the BaseService class defines a template method pattern by providing a common interface and implementing common crud operations (getall, getbyid, add, update, delete). The subclasses (EmployeService, MembershipService, etc. ) inherit from BaseService and can override specific methods as needed.

## 2.4. Tests

**AddEmployeTest**:

-Adds an employee to the database.

-Verifies the successful addition of the employee by asserting the id.

-Cleans up by deleting the added employee.

**AddMembershipTest**:

-Adds a membership to the database.

-Verifies the successful addition of the membership by comparing the retrieved data.

**AddUserTest:**

-Adds a user to the database.

-Verifies the successful addition of the user by comparing the retrieved data.

**DataBaseConnectionTes**t:

-Tests the database connection by creating an instance of the BaseDao class.

-Verifies that the connection object is an instance of pdo.

**DeleteUserTest**:

-Adds a user to the database for testing.

-Deletes the user and verifies that it no longer exists in the database.

These tests are located in the "tests" folder and utilize the phpunit testing framework to ensure the functionality and correctness of the application's features.

# 3. Conclusion

In conclusion, i am satisfied with the overall implementation of the project. The core functionalities, such as managing employees, memberships, and users, have been successfully implemented. The use of design patterns, such as DAO and service layer, has helped in organizing the codebase and promoting separation of concerns.

However, there are a few areas that can be improved in the future. Firstly, the user interface could be enhanced to provide a more intuitive and user-friendly experience. Improving the ui design and usability would greatly enhance the overall user experience.

Additionally, there is room for adding new important features to further enhance the functionality of the gym management system. For example, incorporating features like class scheduling, attendance tracking, or financial management would provide more comprehensive support for gym operations.

Furthermore, expanding the test suite by adding more unit tests and potentially incorporating other types of tests, such as integration or end-to-end tests, would help ensure the robustness and reliability of the application.

During the implementation process, i encountered some challenges, particularly in handling database operations and designing the interactions between different layers of the application. However, these challenges provided valuable learning opportunities, and i am pleased to have gained new knowledge and skills throughout the project.

Overall, while there are areas for improvement, i am satisfied with the implementation and the learnings gained from this project.