Java Final Sprint SD12 - Evan, Nick, & Vanessa

# User Documentation

The **Gym Management System** is a Java-based application designed to help gym administrators manage users (admins, members, trainers), memberships, and workout classes through a command-line interface.

### Application Overview

* Console-based UI with role-based access.
* Users can register, log in, and interact based on their role.
* Admins manage users, memberships, and classes.
* Trainers manage workout classes.
* Members can view and register for classes and purchase memberships.

### Main Classes

* **GymApp.java**Entry point of the application. Initializes database connections and launches the CLI.
* **User.java**Base class for all user types. Includes ID, name, email, and password.
* **Admin.java, Member.java, Trainer.java**Role-specific subclasses of User.java.
* **AdminMenu.java, MemberMenu.java, TrainerMenu.java**Handle the menu system for different user roles.
* **Membership.java**Represents a gym membership with pricing, duration, and type.
* **WorkoutClass.java**Represents a scheduled class with trainer and participant info.
* **Service Classes**Handle business logic for users, memberships, and classes:
  + MembershipService.java
  + WorkoutClassService.java
  + UserService.java
* **DAO Classes**Handle database operations:
  + MembershipDAO.java
  + WorkoutClassDAO.java
  + UserDao.java

### Class Diagram

| classDiagram  User <|-- Admin  User <|-- Trainer  User <|-- Member  Admin --> Membership  Member --> Membership  Trainer --> WorkoutClass  Member --> WorkoutClass  class User {  - int id  - String name  - String email  - String password  - String role  }  class Membership {  - int id  - String type  - double cost  - int userId  }  class WorkoutClass {  - int id  - String classType  - String description  - int trainerId  } |
| --- |

### How to Use the System

**Step 1: Compile the Project**

| mvn compile |
| --- |

This command tells Maven to compile the source code of the project.

**Step 2: Run the Application**

| mvn exec:java -Dexec.mainClass="org.keyin.GymApp" |
| --- |

This command uses Maven's exec plugin to run the program directly from the compiled classes.

From there, you can register or log in using the command line interface. Menus are shown based on your role (Admin, Trainer, Member).

## Development Documentation

### Javadoc Comments

Javadoc comments are included in UserService.java, MembershipService.java, and other logic classes. Example:

| /\*\*  \* Registers a new user.  \* @param user User object  \* @return true if registration was successful  \*/  public boolean registerUser(User user) {...} |
| --- |

### Directory Structure

| src/main/java/org/keyin/  ├── GymApp.java  ├── cli/  │ ├── AdminMenu.java  │ ├── MemberMenu.java  │ ├── TrainerMenu.java  ├── database/  │ └── DatabaseConnection.java  ├── memberships/  │ ├── Membership.java  │ ├── MembershipDAO.java  │ └── MembershipService.java  ├── user/  │ ├── User.java  │ ├── UserDao.java  │ ├── UserService.java  │ └── childclasses/  │ ├── Admin.java  │ ├── Member.java  │ └── Trainer.java  └── workoutclasses/  ├── WorkoutClass.java  ├── WorkoutClassDAO.java  └── WorkoutClassService.java |
| --- |

### Build Process

#### Prerequisites

* Java 17+
* Maven

#### Dependencies

* BCrypt (for password hashing)
* PostgreSQL
* JDBC Driver

### Database Setup

| CREATE TABLE users (  id SERIAL PRIMARY KEY,  name VARCHAR(100),  email VARCHAR(100),  password VARCHAR(255),  phone VARCHAR(20),  address VARCHAR(255),  role VARCHAR(20)  );  CREATE TABLE memberships (  id SERIAL PRIMARY KEY,  type VARCHAR(50),  description TEXT,  cost DECIMAL(10,2),  user\_id INT REFERENCES users(id)  );  CREATE TABLE workout\_classes (  id SERIAL PRIMARY KEY,  class\_type VARCHAR(50),  description TEXT,  trainer\_id INT REFERENCES users(id)  ); |
| --- |

### Cloning and Running from GitHub

| git clone https://github.com/EvKavK/Java\_Sprint\_Final\_ek\_nk\_vm  cd gym-management  mvn compile  mvn exec:java -Dexec.mainClass="org.keyin.GymApp" |
| --- |

## Individual Reports

### Evan

I mainly worked on the database side of things - creating the database itself, and the CRUD functionality for UserDao.java, MembershipDAO.java, and WorkoutClassDAO.java. I also worked alongside Nick to help with implementing said DAOs into the rest of the code, as well as doing some bug fixing once everything was said and done.

Some challenges faced were revolving around the bug fixing itself, mainly in sorting out those issues and making sure the code was tidy and functional. I also needed some extra help with sorting out revenue tracking in particular.

### Nick

My primary contribution for the group was to tackle a bulk of the service classes. I also worked with Vanessa in theorizing the construction of our enumerative menu systems, and worked alongside Evan in creating the DAOs and integrating the database to the rest of the code.

The primary challenge faced was the surprising amount of interplay between classes. It created a little bit of a web at points where things were breaking other things, and required some dedicated debugging time.

### Vanessa

I made changes to fix the CLI menu system to make it more modular, and I also added the Jansi library to enhance the CLI. Then I used that added colour and style to make it more user-friendly and easier on the eyes. Finally I worked on both this documentation and the video for the project. The main challenge I faced during development was installing Maven - I feel like this should have been gone over during regular classes.