Gym Management System

Complete Documentation

the fresca Fellowship

Java Program to manage a Gym

Michael Barney, Joey Thomas, brandon pike

2025

Contents

[Gym Management System Overview 1](#_Toc205802449)

[Complete System Features: 1](#_Toc205802450)

[Explanation of Classes 2](#_Toc205802451)

[UML Class Diagram 3](#_Toc205802452)

[Instructions 5](#_Toc205802453)

[Prerequisites 5](#_Toc205802454)

[Setup 6](#_Toc205802455)

[Project Structure 7](#_Toc205802456)

[Individual Reports 8](#_Toc205802457)

[Joey Thomas Personal Report 8](#_Toc205802458)

[Brandon Pike Personal Report 8](#_Toc205802459)

[Michael Barney Personal Report 8](#_Toc205802460)

# Gym Management System Overview

The Gym Management System is a comprehensive console-based application designed to manage all aspects of a gym's operations. This system handles user registration and authentication, membership management, workout class scheduling, merchandise inventory, and provides role-based access control for different types of users.

## Complete System Features:

1. User Authentication & Registration

* Secure user registration with BCrypt password hashing
* Role-based login system (Admin, Trainer, Member)
* User profile management with contact information

1. Membership Management

* Gym membership purchase and tracking
* Revenue reporting for administrators

1. Workout Class Management

* Class creation and scheduling by trainers
* Class browsing and enrollment for members
* Trainer-specific class management

1. Merchandise Management

* Product inventory management
* Price setting and stock reporting
* Member product browsing

1. Logging & Reporting

* System activity logging to text files
* Revenue and inventory reports
* User activity tracking

# Explanation of Classes

1. User Management
   1. User registration with secure password hashing (BCrypt)
   2. User authentication and login
   3. Role assignment (Admin, Trainer, Member)
   4. User profile management

Key Classes:

* User
* Admin, Trainer, Member (inherit from User)
* UserDAO, UserService

1. Membership Management
   1. Gym membership creation
   2. Member type management

Key Classes:

* Membership
* MembershipDAO, MembershipService

1. Workout Classe Management
   1. Workout class creation by trainers
   2. Class scheduling and management
   3. Member browsing and enrolling for classes
   4. Trainers assignment tracking

Key Classes:

* WorkoutClass
* WorkoutClassDAO, WorkoutClassService

1. Gym Products Management
   1. Product Inventory management
   2. Price setting and updating
   3. Stock level tracking

Key Classes:

* GymProduct
* GymProductDAO, GymProductService

1. Authentication & Security
   1. Password encryption using BCrypt
   2. Role-based access
   3. Security validation
2. System Logging
   1. System activity logging
   2. Error logging
   3. File-based log storage

# UML Class Diagram

A diagram of a company

AI-generated content may be incorrect.

A diagram of a gym management system

AI-generated content may be incorrect.

# Instructions

## Prerequisites

1. Java Development Environment

* Java 11 or higher
* IntelliJ IDEA, Eclipse, or VS Code (see next attachment for installation instructions) <https://www.youtube.com/watch?v=tmEETJUz8uA>

1. Database Setup

* PostgreSQL Server installed and running (see next attachment for installation instructions) <https://www.youtube.com/watch?v=IYHx0ovvxPs>

## Setup

1. GitHub Setup
   1. Download GitHub desktop application <https://desktop.github.com/download/>
   2. Create a GitHub Account and Login
   3. Open the GitHub App
   4. Press CTRL + SHIFT + O to clone a repository a window will open, click the URL tab and it should look like thisA screenshot of a computer program

      AI-generated content may be incorrect.
   5. In the URL box paste this URL: <https://github.com/MikeBarney88/Java_Final_Sprint_MB_JT_BP.git>
   6. Choose the folder on your computer where you would like it to be stored.
   7. Click Clone
   8. Then open your IDE of choice and under the file tab open the folder where you saved the Clone.
2. Database Initialization
   1. Create database: CREATE DATABASE gym\_management;
   2. Run the SQL scripts to create all the tables from the resources folder from the IDE or where you saved the Clone.
3. Build process and dependencies
   1. Build process is IntellJ (Maven)
   2. Dependencies

<dependencies>  
 <!-- https://mvnrepository.com/artifact/org.postgresql/postgresql -->  
 <dependency>  
 <groupId>org.postgresql</groupId>  
 <artifactId>postgresql</artifactId>  
 <version>42.7.5</version>  
 </dependency>  
  
 <!-- https://mvnrepository.com/artifact/org.mindrot/jbcrypt -->  
 <dependency>  
 <groupId>org.mindrot</groupId>  
 <artifactId>jbcrypt</artifactId>  
 <version>0.4</version>  
 </dependency>  
</dependencies>

1. For the Javadoc and Project Directory Structure explanation see the video that is saved in the resources folder it gives an in-depth breakdown of key methods and classes, as well as Instructions on how to start and use the system.

# A screenshot of a computer AI-generated content may be incorrect.Project Structure

# Individual Reports

### Joey Thomas Personal Report

Joey's Contributions: I made 13 total Commits/PRs, more specifically I was in charge of the User parent class and it's subclasses (Admin, Trainer, Member) as well as the GymApp menu console interface and the CustomLogger for logging errors and info to a text file to store as a record.  
  
Joey's Challenges: Implementing RBAC (Role Based Access Control) was a little tricky to get used to because I've only ever done it in JavaScript prior to this assignment.

### Brandon Pike Personal Report

Brandon’s Contributions: I made one big commit/push for these files.

I worked on the Memberships and Workout classes sections, which involved doing stuff like constructors, and some getters and setters and such within the code for them.

Both Memberships and Workout Classes had 3 files within them being: Membership, MembershipDAO, MembershipService, and WorkoutClass, WorkoutClassDAO, and WorkoutClassService.

In all the files for Membership, and WorkoutClass I also wrote in the Javadocs on everything, showing where the constructors are, and showing the Getters and Setters.

Brandon’s Challenges: figuring out how to use the DAO classes

### Michael Barney Personal Report

Michael’s Contributions: I made 10 Commits/PRs, I did the database connection including the scripts, I setup the GymProducts, GymProductsDAO, GymProductsServices. I also completed the UserDocumentation guide with all the data needed to set our project up to be sold as a software application.

Michael’s Challenges: I had a little trouble starting to use the DAO classes, also the documentation was quite extensive and that took a fair amount of time to complete and make sure I had all the pertinent information.