COMP 3005

Fall 2025

Project: Health and Fitness Club Management System

Instructor: Abdelghny Orogat

Andrew Dang 101297865

Dylan Nakamura 101306225

December 1, 2025

Youtube Video

The Health and Fitness Club Management System was developed as a database-driven application designed to manage members, trainers, and administrative operations within a fitness center. This Management System integrates key functions such as Member user registration, scheduling personal training sessions, class registration, trainer availability management, room allocation, and equipment maintenance tracking. For this implementation, Maven was used to handle the automated download, versioning, and linking of all ORM-related dependencies. The pom.xml file includes the required Hibernate libraries, ensuring integration between the relational schema and Java-based entity classes

Assumptions:

The Member entity contains basic attributes like name and email. They also contain some attributes like targetWeight and targetBMI for their goals. HealthMetric is a weak entity which relies on a Member to exist. This will contain a member's current stats as well as a timestamp. Members can hold onto multiple HealthMetrics. Members can also optionally sign up for many different GroupFitnessClasses or PersonalTrainingSessions. These GFCs and PTSs need total participation for Members and Trainers, while Member and Trainer are partial. GFCs will hold onto attributes like its name, as well as the currentMember count and max capacity. PTSs contain a room and time. As mentioned before, a Trainer can be in charge of multiple different PTSs and GFCs. Trainers have a name, email, and multiple times when they are available. A trainer does not need to lead a PTS or GFC, so they are partial participants. A single ClassSchedule will track all of the GroupFitnessClasses. It holds the classTime and roomNum. Both are total participants. Admins, who have a name and email, are the ones who manage and organize the class schedule. ClassSchedule must be updated by an Admin. Multiple Admins can update the Schedule,

however they do not need to. EquipmentManagement holds onto the room, issue, and status of an equipment. Needs to be updated by an Admin. Multiple Admins can update the equipment, but they are not obligated to.

ORM Usage:

The system's ORM layer was implemented using Hibernate, allowing object-oriented Java classes to be mapped directly to relational database tables. This eliminated the need for writing most SQL statements and operations such as registering members, adding health metrics, booking sessions, and managing trainer availability were done through object manipulation. Hibernate annotations were used to define entities, primary keys, foreign keys, and relationships such as one-to-many or many-to-many. Maven managed all Hibernate dependencies, and the framework handled schema creation.

```
Session session = factory.openSession();
session.beginTransaction();
session.persist(member1);
session.getTransaction().commit();
session.close();
```

This code is a snippet of PopulateDatabase.java. It opens a Hibernate session, starts a database transaction, adds a member and then commits it. The transaction is opened and closed properly. Finally, the session is closed to free database resources.

```
@Entity
@Table(name = "admins")
public class Admin {
```

```
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long adminId;

    @Column(nullable = false)
    private String name;

    @Column(nullable = false, unique = true)
    private String email;

    public Admin() {}

    public Admin(String name, String email) {
        this.name = name;
        this.email = email;
    }
```

This class defines an Admin entity that Hibernate maps to the admins table in the database. The adminId field is the primary key and is automatically generated by the database. The name and email fields are required, and the email must be unique.

Many entities are present. Within the database, there exists Admin, ClassSchedule and ClassScheduleDetails, EquipmentManagement and EquipmentManagementDetails, GroupFitnessClass and GroupFitnessClassMembers, HealthMetric and HealthMetricDTO, Member, PersonalTrainingSession and PersonalTrainingSessionDetails, Trainer and TrainerAvailability.

ClassSchedule uses an index, to make querying faster. HealthMetric has a DTO which essentially acts as a View with hibernate. This relies on MemberService, which will fetch the latest HealthMetric. HealthMetric also contains a trigger which updates the timestamp upon entering or updating in the database. PersonalTrainingSession and ClassSchedule also contain a trigger upon entering the database. The time set for the schedule will automatically be taken from the associated trainer.