# ClubConnect - University Clubs Management System

This project implements the core functionality for the ClubConnect system.

## Section A Implementation Details

The following core requirements from Section A have been implemented:

1. **User Authentication and Roles (10 marks):**
    - **Login Screen:** Implemented in `LoginFrame.java` with secure password hashing using `PasswordHasher.java` .
    - **Registration:** Implemented in `RegistrationFrame.java` with input validation using `Validator.java` and checks for unique Student ID and Email using `UserDAO.java` .
    - **Role-specific Dashboards:** The `LoginFrame` directs users to the appropriate dashboard ( `AdminDashboard` , `LeaderDashboard` , `MemberDashboard` ) based on their `UserRole` .
    - **Guest Mode:** Implemented in `LoginFrame.java` and `GuestDashboard.java` to allow unauthenticated viewing.
    - **Session Management:** Handled by `SessionManager.java` to maintain the logged-in user state.
2. **Look and Feel (Implicit in GUI):**
    - The GUI components ( `LoginFrame` , `RegistrationFrame` , and all Dashboard placeholders) use **Java Swing** with a consistent, professional, and intuitive layout (BorderLayout, GridBagLayout, FlowLayout) as required.
    - Basic color schemes and fonts are applied for visual appeal.
3. **Data Persistence (Implicit in DAO/DB):**
    - **MySQL Database:** The system is designed to use MySQL via JDBC.
    - **Database Initialization:** `DatabaseManager.java` automatically creates the database ( `clubconnect_db` ) and all required tables ( `users` , `clubs` , `events` , etc.) on application startup if they do not exist.
    - **User Data Access:** `UserDAO.java` provides all necessary CRUD and authentication methods for user data.

## Project Structure

The project follows a standard Java package structure:

``` . ├── src │ └── com │ └── clubconnect │ ├── MainApp.java <- Main entry point │ ├── dao <- Data Access Objects (DAO) │ ├── database <- Database connection and management │ ├── models <- Data Models (POJOs) │ ├── ui <- User Interface (GUI Frames/Panels) │ └── utils <- Utility classes (Hashing, Validation, Session) ├── run.sh <- Script to compile and run the application ├── C7-JAV-11-END-QP.pdf <- Assignment Question Paper ├── CompleteProjectGuide-ClubConnect.pdf ├── IMPLEMENTATION_GUIDE.md.pdf ├── PROJECT_STRUCTURE.md.pdf └── QUICK_REFERENCE_CARD.md.pdf ```

## Important Note on Database Connection

**Due to environmental constraints, the MySQL server was not installed on the local machine where this code was finalized. Therefore, the database connection and initialization logic were not tested end-to-end.**

The code is structured to meet the **Data Persistence** requirement:

- The application uses JDBC and is configured to connect to a MySQL server on `localhost:3306`.
- `DatabaseManager.java` contains the logic to automatically create the `clubconnect_db` database and all necessary tables on first run.
- The default credentials in `db_manager.java` are set to a common testing configuration (`DB_USER = "root"`, `DB_PASSWORD = ""`). **If your local MySQL server requires a password, you MUST update these constants before running.**

---

## How to Run the Application

**Prerequisites:**

1. Java Development Kit (JDK) 8 or higher.
2. A running MySQL server instance.
3. The MySQL JDBC Connector JAR file (e.g., `mysql-connector-j-8.0.33.jar`) must be placed in the project root directory.

**Steps:**

1. **Download the MySQL JDBC Connector:** Ensure you have the JDBC driver.
2. **Update DatabaseManager:** If your MySQL server requires a password, you must edit `src/com/clubconnect/database/db_manager.java` and update the `DB_USER` and `DB_PASSWORD` constants.
3. **Execute the Run Script:** ```bash ./run.sh ```

The script will compile all Java files and execute the `MainApp`. The application will automatically initialize the database and open the `LoginFrame`.

**Note on Dashboards:** The dashboard classes (`AdminDashboard`, `LeaderDashboard`, `MemberDashboard`, `GuestDashboard`) contain placeholder content for the functionality required in Section B. The focus of this submission is the successful implementation of the Section A requirements (Authentication, Registration, Session Management, and Database Initialization).