Database Systems

CIS2101, IS273

Project:

Sporting Club Management System

Objective:

You are tasked with developing a system for managing a sporting club. The system must handle various aspects of the club's operations, including **members**, **subscriptions**, **teams**, and **expenses**. The system should be implemented either as a **web application** (using Python) or as a **desktop application** (using Java).

You are required to design the database, implement the application logic, and provide an intuitive user interface. Finally, you will submit the **Entity-Relationship Diagram (ERD)**, **source code**, and either the **URL of the website** or the **executable file** of the desktop application.

Functional Requirements:

The system should have the following functionalities:

1. **Member Management**:

- Add, edit, and delete members.
- View the list of all members.
- Track the subscription status for each member (active, inactive, overdue).
- Record member contact information (name, phone number, email).

2. **Subscription Management**:

- Add subscription plans (e.g., monthly, quarterly, yearly).

- Assign subscriptions to members.
- Track payments made by members.
- Generate reports on overdue subscriptions.

3. **Team Management**:
   - Create and manage teams within the club (e.g., Football Team, Basketball Team, etc.).
   - Add members to teams.
   - View the roster of each team.
   - Assign team leaders or coaches.

4. **Expense Management**:
   - Record and track expenses for the teams (e.g., equipment, travel expenses, training costs).
   - Allocate expenses to specific teams.
   - Generate reports on team expenses.

5. **Reporting**:
   - Generate reports on total income from subscriptions.
   - Generate reports on team expenses.
   - Track financial health of the club.
   - Track membership statistics (active, inactive, overdue).

6. **User Authentication**:
   - Secure login system for administrators to manage the club.
   - Admin access to all features, while members may have limited access.

Database Design:

You need to design an **Entity-Relationship Diagram (ERD)** that illustrates the key entities and relationships in the system. The following are the main entities you should consider:

Members

- MemberID
- Name
- Email
- PhoneNumber
- SubscriptionStatus (active, inactive, overdue)

Subscriptions

- SubscriptionID
- PlanType (e.g., Monthly, Quarterly, Yearly)
- StartDate
- EndDate
- Amount

Teams

- TeamID
- TeamName
- TeamLeaderID

TeamMembers

- TeamMemberID
- MemberID

  - TeamID

Expenses

  - ExpenseID

  - ExpenseType (e.g., equipment, travel, etc.)

  - Amount

  - Date

  - TeamID

Technologies:

- **Backend**:

  - Python with Flask/Django for the web application, or Java with JavaFX/Swing for the desktop application.

  - Database: MySQL, PostgreSQL, or SQLite.

- **Frontend** (if implementing as a web app):

  - HTML/CSS, JavaScript (for web interfaces).

  - Use any frontend framework (e.g., React, Bootstrap, or plain HTML).

- **Libraries/Frameworks**:

  - Python: Flask or Django for the web app.

  - Java: JavaFX or Swing for the desktop app.

Deliverables:

1. **Entity-Relationship Diagram (ERD)**:

   - Submit an ERD showing all entities and their relationships.

- Include attributes for each entity.
- Represent all primary keys, foreign keys, and any relevant constraints.

2. **Source Code**:
   - Submit the full source code of the application (either the Python web app or Java desktop app).
   - Include comments and documentation within the code.
   - Organize your code with clear folder structure (e.g., separate files for models, views, controllers).

3. **Executable or Web URL**:
   - **For a web app**: Provide a URL to the deployed web application.
   - **For a desktop app**: Provide a compiled executable file (.exe for Windows, .jar for Java) and any necessary instructions for running the application.

Evaluation Criteria:

Your project will be evaluated based on the following criteria:

1. **Functionality**:
   - Does the application implement all the required features (member management, subscription management, etc.)?
   - Does the application perform all actions as specified in the functional requirements?
   - Is the database design appropriate and normalized?

2. **Database Design**:
   - Is the ERD well-designed, with clear relationships between entities?
   - Are all necessary entities and relationships included?
   - Is the database normalized (at least to 3NF)?

3. **Code Quality**:

   - Is the code well-structured and easy to understand?

   - Are appropriate comments and documentation included?

   - Is the application free from major bugs or errors?

4. **User Interface**:

   - Does the application have an intuitive and user-friendly interface (UI)?

   - If it's a web app, does the frontend have a clean and responsive design?

   - If it's a desktop app, is the UI consistent and easy to navigate?

5. **Deployment**:

   - If it's a web app, is it deployed and accessible online?

   - If it's a desktop app, is it executable and easy to run?

Optional Advanced Features (Bonus Points):

- **Email Notifications**: Notify members about subscription renewals or overdue payments via email.

- **Dashboard**: A summary dashboard for administrators to view key statistics (total income, active members, team expenses, etc.).

- **Mobile Responsiveness (Web App)**: Ensure the web application is responsive on mobile devices.

- **Search and Filter**: Add search functionality for members, teams, or expenses.

- **Graphical Reports**: Visualize data (e.g., subscription revenue, team expenses) in charts or graphs.

Project Submission:

Submit the following through the course's submission portal:

1. Your **ERD** diagram as a PDF or image file.
2. Your **source code** in a zip file.
3. **Executable file** (.exe/.jar) for desktop apps or a **link to your deployed website** for web apps.
4. A **short user guide** explaining how to run and use the application.

Deadline:

- **Submission Deadline**: 26-12-2024

Good Luck