Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (Spring, Year: 2024), B.Sc. in CSE (Day)

UML Class Diagram

Lab Report 8

Course Title: Integrated Design Project I
Course Code: CSE-324
Section:213- D1

Students Details

Name	**ID**
Arman Hossain	221002624
Jannatul Ferdous	221902002
Afnan Khan Shopnil	221002570

Submission Date: 22 December 2024
Course Teacher's Name: Rusmita Halim Chaity

[For teachers use only: Don't write anything inside this box]

Lab Project Status

Marks:	**Signature:**
Comments:	**Date:**

1 Objective

- **Understand Class Relationships:** Analyze the interactions and connections between entities in the quiz management system.
- **Define Class Responsibilities:** Identify key attributes and methods for each class to fulfill system requirements.
- **Implement Object Interactions:** Build and test code reflecting relationships like inheritance and composition.
- **Apply OOP Principles:** Use encapsulation, polymorphism, and inheritance to achieve efficient functionality.
- **Validate System Design:** Ensure the implemented system aligns with the class diagram through testing and debugging.

2 Introduction

The system is designed for managing online quizzes. It consists of multiple classes, each responsible for specific aspects of the functionality. The key components include authentication, user management, quiz operations, leaderboard maintenance, notifications, and withdrawal functionality.

3 Description of Classes and Their Roles

3.1 Authentication

- **Attributes:**
  - authId: String
  - otp: String
- **Methods:**
  - verifyOTP(otp: String): Boolean
  - validateCredentials(username: String, password: String): Boolean
- **Purpose:** Handles user authentication, including OTP verification and credential validation.

3.2 User

- **Attributes:**
  - userId: String
  - userName: String
  - email: String
  - phoneNumber: String
- **Methods:**
  - register(): Registers a new user.

  - login(): Logs in the user.
  - logout(): Logs out the user.
- **Purpose:** Base class for user management, supporting registration and session management.

3.2 Offline User (Inheritance from User)

- **Methods:**
  - viewScoreboard(): Allows offline users to view quiz results.

3.3 Online User (Inheritance from User)

- **Attributes:**
  - accountBalance: Float
- **Methods:**
  - participateQuiz(): Enables users to participate in a quiz.
  - viewLeaderboard(): Displays the leaderboard.
  - changePassword(): Allows password updates.
  - withdrawCash(amount: Float): Facilitates withdrawal of account balance.

3.4 Guest User (Inheritance from User)

- **Attributes:**
  - sessionId: String
- **Methods:**
  - playOneTimeQuiz(): Permits guest users to attempt a quiz without registration.

3.5 Quiz Management

- **Attributes:**
  - adminId: String
  - adminName: String
- **Methods:**
  - addQuiz(quiz: Quiz): Adds a new quiz.
  - deleteQuiz(quizId: String): Removes a quiz.
  - updateQuiz(quiz: Quiz): Updates an existing quiz.
  - manageDataBackup(): Manages backup of quiz data.

3.6 Quiz

- **Attributes:**
  - quizId: String
  - title: String

  - description: String
  - startTime: DateTime
  - endTime: DateTime
  - reward: Float
- **Methods:**
  - startQuiz(): Initiates the quiz.
  - submitAnswer(questionId: String, answer: String): Submits an answer.
  - endQuiz(): Ends the quiz.

3.7 Leaderboard

- **Attributes:**
  - rank: Int
  - userId: String
  - score: Float
- **Methods:**
  - updateLeaderboard(userId: String, score: Float): Updates the leaderboard with user scores.
  - getTopRanks(limit: Int): Fetches top-performing users.

3.8 Notification

- **Attributes:**
  - notificationId: String
  - message: String
  - dateTime: DateTime
- **Methods:**
  - sendNotification(userId: String): Sends notifications to users.

3.9 Withdraw

- **Attributes:**
  - withdrawId: String
  - userId: String
  - amount: Float
  - dateTime: DateTime
  - status: String
  - isInitiated: Boolean
- **Methods:**
  - confirmWithdraw(withdrawId: String): Boolean: Confirms a withdrawal.
  - cancelWithdraw(withdrawId: String): Boolean: Cancels a withdrawal.

4. Relationships

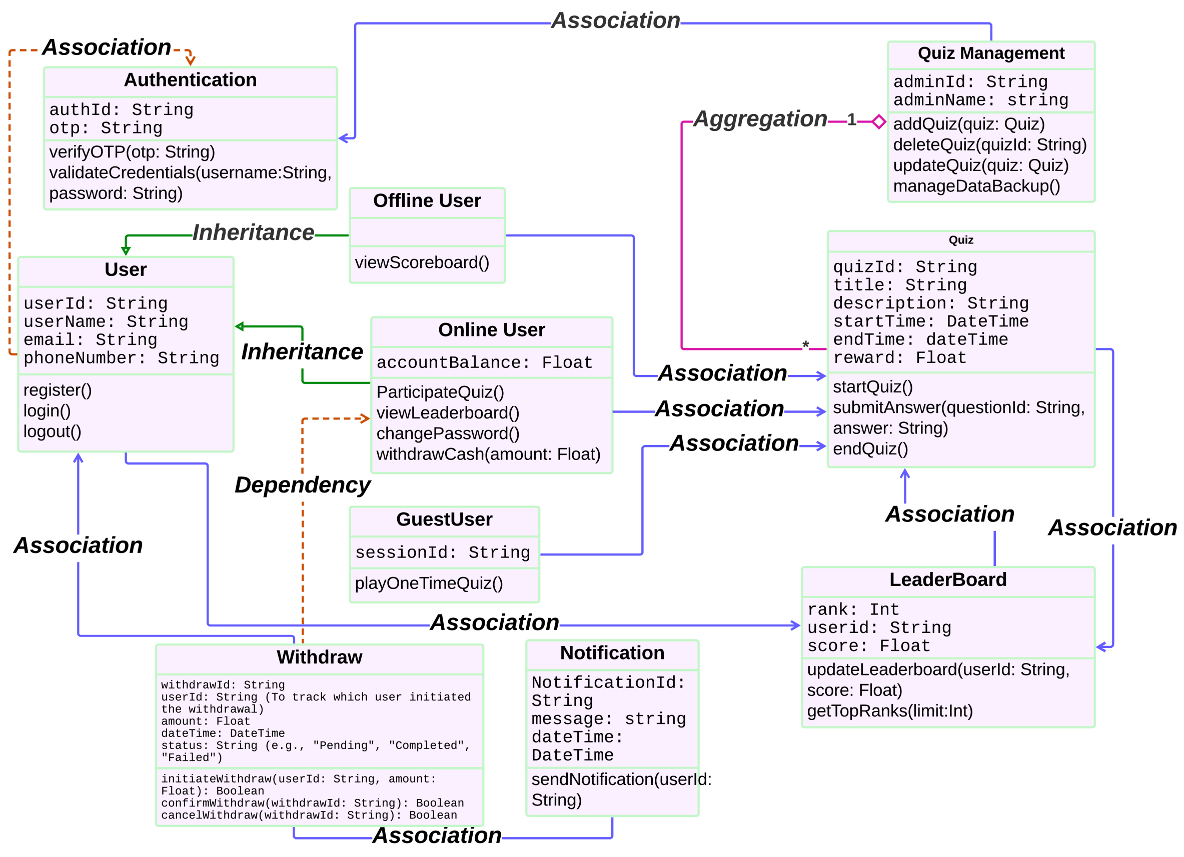
The class diagram highlights the following relationships:

- **Inheritance:** User class serves as a base for OfflineUser, OnlineUser, and GuestUser.
- **Aggregation:** QuizManagement class aggregates Quiz objects.
- **Association:** Relationships between User and various other classes, such as Leaderboard, Notification, and Withdraw, enable interaction.
- **Dependency:** OnlineUser depends on Withdraw for managing cash transactions.

4 Procedure

1. Define the classes and their attributes based on the diagram.
2. Implement the methods as per the functionality described.
3. Establish relationships between classes using inheritance, association, and aggregation.
4. Test the system by simulating various scenarios, such as user registration, quiz participation, leaderboard updates, and cash withdrawal.

5 Implementation



Class Diagram for Quiz and Earn: A Dual-Mode Quiz App

6 Results

After implementing the class diagram, the system should:

- Support user registration, login, and authentication.
- Allow users to participate in quizzes and view leaderboards.
- Manage notifications and cash withdrawals effectively.
- Enable administrators to add, update, and delete quizzes.

7 Conclusion

The class diagram provides a comprehensive blueprint for designing a quiz management system. Its successful implementation demonstrates the utility of object-oriented design principles in building modular and maintainable software systems.

8 References

- UML Class Diagram Notation
- Object-Oriented Design Patterns
- Java/Python Documentation for OOP Concepts