Student: Rogoz Bogdan Dorin
Group: 30433

Table of Contents

1. Requirements Analysis

1.1 Assignment Specification

Design and implement an application for a ping-pong association that organizes tournaments on a regular basis.

1.2 Functional Requirements

The application should have two types of users: a regular user represented by the player and an administrator user. Both kinds of uses have to provide an email and a password in order to access the application. The regular user can perform the following operations:

- View Tournaments
- View Matches
- Update the score of their current game. (They may update the score only if they are one of the two players in the game. The system detects when games and matches are won)

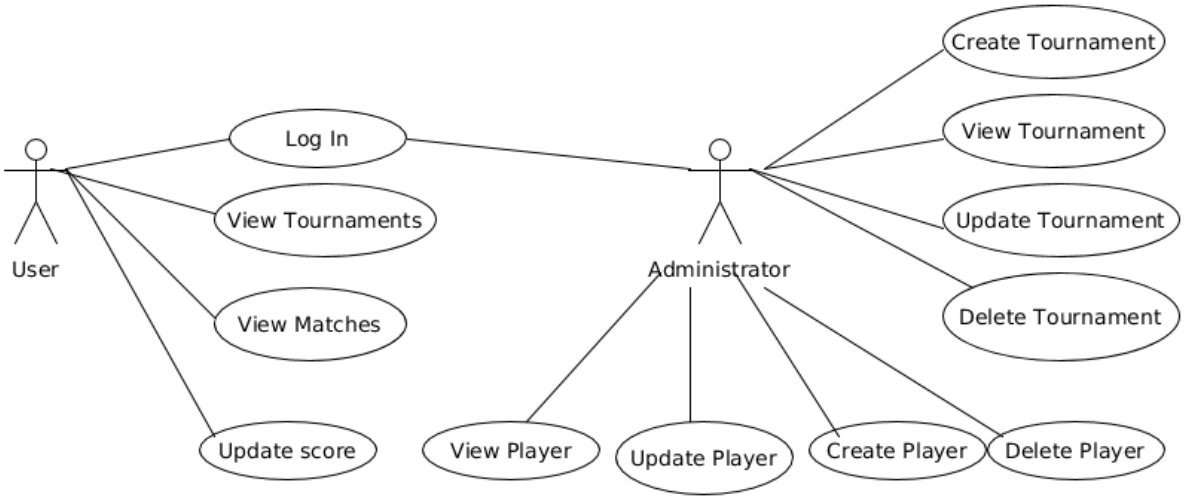
The administrator user can perform the following operations:

- CRUD on player accounts
- CRUD on tournaments: He creates the tournament and enrolls the players manually.

1.3 Non-functional Requirements

- The data will be stored in a database. Use the Layers architectural pattern to organize your application. Use a domain logic pattern (transaction script or domain model) / a data source hybrid pattern (table module, active record) and a data source pure pattern (table data gateway, row data gateway, data mapper) most suitable for the application.
- All the inputs of the application will be validated against invalid data before submitting the data and saving it in the database.

2. Use-Case Model



Use case: User Log In

Level: User-goal level

Primary actor: User

Main success scenario: The user provides his username and password, the credentials are found in the database and the log in attempt is validated.

Extensions: The username – password combination is not found, so the user is prompted to try again.

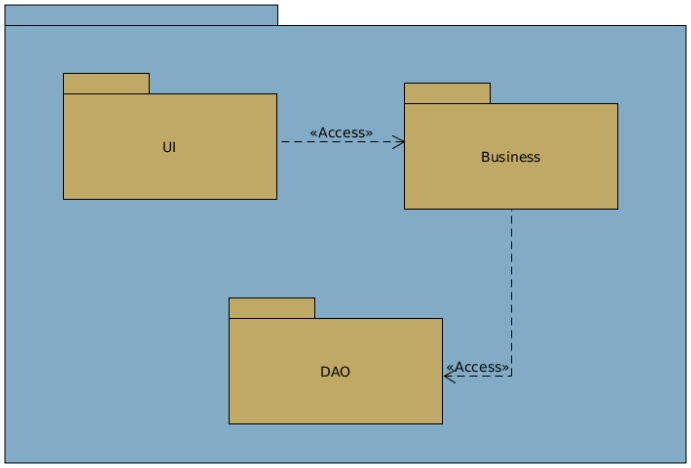
3. System Architectural Design

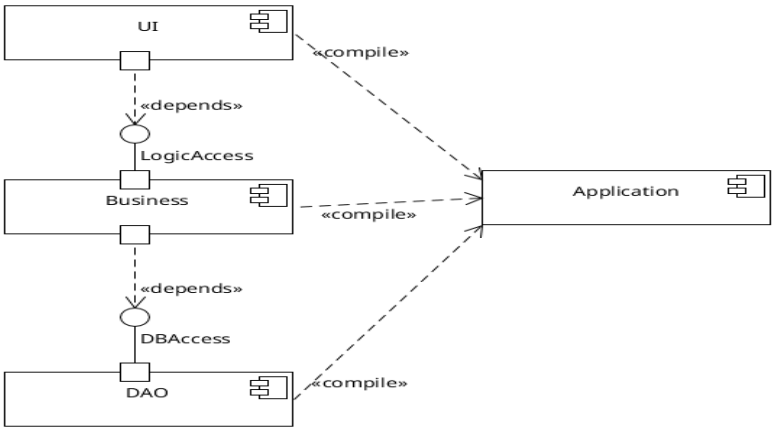
3.1 Architectural Pattern Description

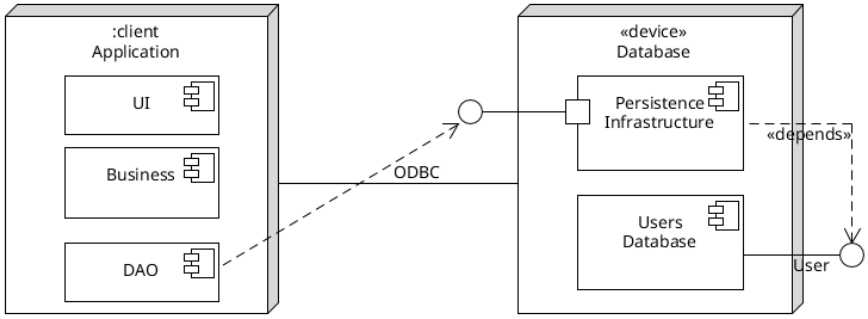
The architectural pattern used is the Layers pattern, in order to decompose the code into 3 distinct, non-overlaping layers:

- DAO, which accesses the external database
- Business, which processes data provided by the user and uses methods visible inside the DAO
- UI, the Graphical User Interface

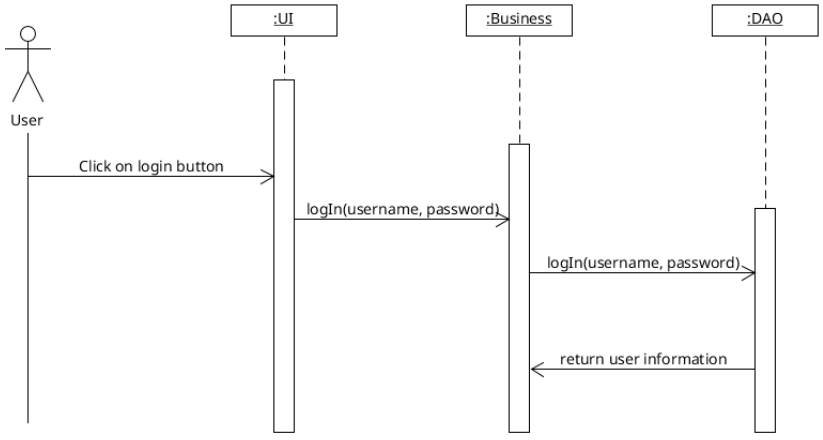
3.2 Diagrams







4. UML Sequence Diagrams

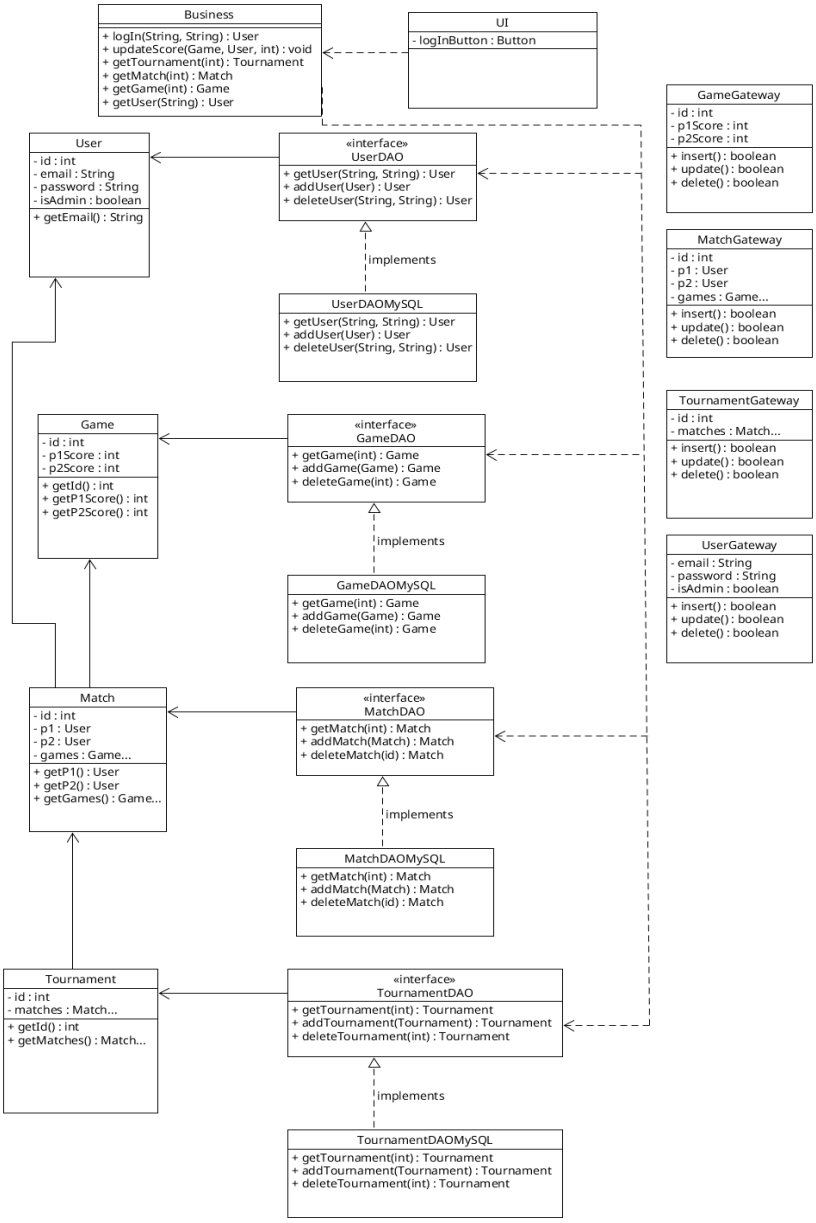


5. Class Design

5.1 Design Patterns Description

Singleton – when a single instance of a given class is wanted across the entire project (in our case the Database Connector)

5.2 UML Class Diagram



6. Data Model

User : represents the ones who use the application; can be either a player (regular user) or admin.
Game : represents a game from a set.
Match : contains information about the games played by 2 players.
Tournament : contains multiple matches.

7. System Testing

Multiple unit tests can be employed, in order to test the system's sanity:

- connect to the database and check status
- check if the "match ended" logic is correct, for a couple of test inputs
- check if invalid input data is correctly processed by the application

8. Bibliography

Martin Fowler – Patterns of Enterprise Application Architecture
https://www.smartdraw.com/component-diagram/