# System Design

**Group:** Under Development
**Project:** SportCred
**Instructor:** Prof. Illir Dema
**TA:** Jerry Meng

# Table of Contents

# CRC Cards

---

Class Name: **Account**

Responsibilities:
*Handle basic information of the user (username, password, bio, email, phone number, people on radar, A.C.S.)

Parent Class: None

Subclasses: None

Collaborators:
*DBLogin
*DBProfile

---

INTERFACE: **Game**

Responsibilities:
*Every class that implements **Game** must calculate changes in ACS

Classes that implement this interface:
*DebateHandler
*TriviaHandler
*PicksPredictions

---

Class Name: **SignUpHandler**

Responsibilities:
*Allows users to create a new account.
*Asks users for their personal data and stores it

Parent Class: None

Subclasses: None

Collaborators:
*DBProfile
*DBLogin
*ProfileHandler

---

Class Name: **LoginHandler**

Responsibilities:
*Verifies username and password when user logs in

Collaborators:
*DBLogin

---

Class Name: **ProfileHandler**

Responsibilities:
*Stores the information of a user
*Allows user to update their information

Parent Class: None

Subclasses: None

Collaborators:
*DBProfile
*DBLogin

---

Class Name: **DebateHandler (Implements Game)**

Responsibilities:
*Sends question to user based on their tier
*Notifies user to answer question
*Allows users to agree/disagree to the questions
*Calculates users' ACS based on their responses and participation

Parent Class: None

Subclasses: None

Collaborators:
* DBDebate

---

Class Name: **TriviaHandler (Implements Game)**

Responsibilities:
*Allow users to play alone or with another user
*Generates questions and options
*Allows user to pick an option within a time limit
*Checks what the user answered, and sees if it is correct (and returns that info)
*Display who the winner is in a two player game
*Calculates users' ACS based on their answers

Parent Class: None

Subclasses: None

Collaborators:
*DBTrivia

---

Class Name: **ZoneHandler**
Responsibilities:
*Displays the posts for the page
*Displays an updated response when user makes a new post, comments on a post or agrees/disagrees to a post
*Allows users to search for posts related to a particular topic

Parent Class: None

Subclasses: None

Collaborators:
*DBZone

---

Class Name: **PicksPredictionsHandler (Implements Game)**

Responsibilities:
*Allow users to select a game whose outcome they want to predict
*Allow users to predict which team wins in a game.
*Display the game's outcome to users once a game is over
*Increase and decrease the ACS points if a user picks correctly/incorrectly.

Parent Class: None

Subclasses: None

Collaborators:
*DBPicksPredictions
---
Class Name: **DBLogin**

Responsibilities:
*Create and fetch login information from the database

Parent Class: None

Subclasses: None

Collaborators:
*Connect
---
Class Name: **DBUserInfo**

Responsibilities:
*Create user node in DB with all the data given by user when signing up

Parent Class: None

Subclasses: None

Collaborators:
*Connect
---
Class Name: **DBProfile**

Responsibilities:
*Update and fetch user profile information from the database

Parent Class: None

Subclasses: None

Collaborators:

*ProfileHandler
*Connect

---

Class Name: **DBDebate**

Responsibilities:
*Retrieve the topic for the day

Parent Class: None

Subclasses: None

Collaborators:
*Connect

---

Class Name: **DBTrivia**

Responsibilities:
*Store/fetch trivia questions
*Store/fetch the correct answer for each question
*Store/fetch all the answers/options for each question

Parent Class: None

Subclasses: None

Collaborators:
*Connect

---

Class Name: **DBZone (old open_court)**

Responsibilities:
*Stores new posts/stories

Parent Class:

Subclasses: None

Collaborators:
*Connect

---

Class Name: **DBPicksPredictions**

Responsibilities:
*Update/fetch the predictions for sport results

Parent Class:

Subclasses: None

Collaborators:
*Connect

---

Class Name: **Connect**

Responsibilities:
*Handles the connection to our Neo4j Database

Parent Class: None

Subclasses: None

Collaborators: None

# System Architecture - Description

**Questions**

**System Dependencies**

**Q: The description of system interaction with the environment should indicate any dependencies or assumptions made about the operating environment of the system. E.g. OS, programming language compilers and virtual machine, DB's, network configuration, etc.**

Our system is designed to be cross-platformed, across Android, iOS, and web apps. However, it is assumed that the user will have the most up-to-date Android or iOS version or a Web Browser with the latest HTML5 standard. The frontend does not have direct access to the db as its main job is to handle the view to allow the user to interact with our app. To interact with the db, the front end connects with our backend controller through REST APIs which are programmed in Java to handle requests and responses with a Neo4J database. While the server will require a Java compiler and a Neo4j database, the users will only be interacting directly with our frontend and do not need Java or Neo4J on their device.

However for server and network setup, the server should have Java 1.8 installed, and the respective Neo4J version 4.1.1 Enterprise drivers imported to the maven project to get running.

Further neo4j setup can be found here
https://neo4j.com/docs/operations-manual/current/installation/windows/

The java REST API is defaulted to localhost:8080
The neo4j database with configurations

name: neo4j
password: 1234
host: localhost
port: 7687

For frontend dev, flutter installation guide https://flutter.dev/docs/get-started/install

**Q: The system decomposition should relate the system architecture to the detailed design, to identify the role of each component in the higher-level architectural view. Description of strategy for dealing with errors and exceptional cases (e.g. invalid user input, network or external system failure) that might arise**

**in the use of the software. For anticipated errors and exceptions, a summary of how the software will respond in these situations.**

Our frontend has a few basic verification checks (e.g. check that both passwords are the same when signing up). Once basic verification is done, user input will be sent to the control, where our control classes will perform more in depth checks (e.g. check that username and emails aren't taken). In the event that an error occurs, a relevant error message and http status will be returned (e.g. error 409 CONFLICT when a duplicate username is found during sign up).

For any unexpected errors, or in the event that the database is down, an error code of 500, SERVERERROR is returned. These errors are sent to and interpreted by our view/frontend, allowing us to provide relevant feedback to the user.

**Q: Describe the architecture of the system, that is the most abstract view of how your system is divided into components and how those components are interconnected. The architecture should be described with a diagram showing components and how they are related (or equivalent in words). Beware of designs based on a large number of components, they may signal a design that is overly complex.**

## System Architecture - Diagram