MIDDLE EAST TECHNICAL UNIVERSITY
NORTHERN CYPRUS CAMPUS

Computer Engineering Program

CNG 495 - Cloud Computing
2024-2025 Fall

Term Project Proposal
“SoccerMatch Scheduler”

Team Members

Elif Ilgın Savaş - 2453512
Alper Kutay Ören - 2526556
Yiğit Berk Atcı - 2526101

Table of Contents

SoccerMatch Scheduler

The purpose of SoccerMatch Scheduler is for teams that want to play a football match but cannot find an opponent to easily find an opponent and create a match schedule. A person who registers to the system can create their team and add it to the system. Then, they can send a request to the team they want to play. If the two teams agree, a suitable match time is set. The win rate of each team is recorded in the system. As a result, teams can easily find each other and play a fun and competitive game. Our goal is to provide a user-friendly platform where students can easily manage match schedules and participate in university sports way faster.

Cloud Delivery Models

IaaS (Infrastructure as a Service):

For the SoccerMatch Web Application, Amazon Elastic Compute Cloud (Amazon EC2) will virtually serve as the infrastructure backbone that will provide the virtual services for managing scheduling and requests between the users (teams). EC2 can scale up and down considering user demand which will help to handle huge demand without affecting the performance. With this we will have a flexible server management system, and the cost will be efficient since it uses the ‘pay what you use’ method.

PaaS (Platform as a Service):

To be able to manage the database seamlessly, we have chosen Amazon Relational Database Service (Amazon RDS) which will help us focus more on the functionality rather than the management of the infrastructure. RDS will store important data such as user (team) information, match schedules, and rankings. It will provide secure data handling and automatic backups itself. It will scale itself as the

user base grows without affecting the performance. Amazon RDS supports the usage of MySQL; therefore, it will be used in our project.

For notifications of our application we are considering Amazon SNS and we are considering using an email system to notify the users since it is the main system used in our school. SNS allows us to send messages to multiple users instantly, notifying them when a team is looking for an opponent. By using SNS we eliminate the need to manage complex notification infrastructure by automatic mailing with high reliability.

SaaS (Software as a Service):

The SoccerMatch Web Application will work as a SaaS that will allow the students of METU to schedule their matches easier without having to look for teams on the campus in real life. It will provide users with an interface to view or book time slots, send, or accept match requests. It will have automated updates so that the user can easily login and access the application without installation required. This setup will be a powerful tool for students to coordinate soccer matches efficiently.

For SoccerMatch Scheduler, we will be implementing the backend with Python-Flask and will be using pure HTML, CSS JavaScript for the frontend. As mentioned before the database will be implemented using MySQL.

Cloud Model	AWS Service	Role
IaaS	EC2	Will provide the server for hosting the backend and handling the processes.
PaaS	RDS	Will handle the database without the infrastructure level of the model
PaaS	SNS	Will handle notifications via email without complex structure in the application.
Saas	SoccerMatch	Will manage the end-user part where they make schedules through the web app.

Figure 1: Cloud Delivery Models Table

Diagrams & Figures

Client-Server Interactions

The following sequence diagram illustrates some of the actions that can be taken in the application. These actions are the login, main page time slot table display, selecting a schedule for a match, and how a user can check their notifications.

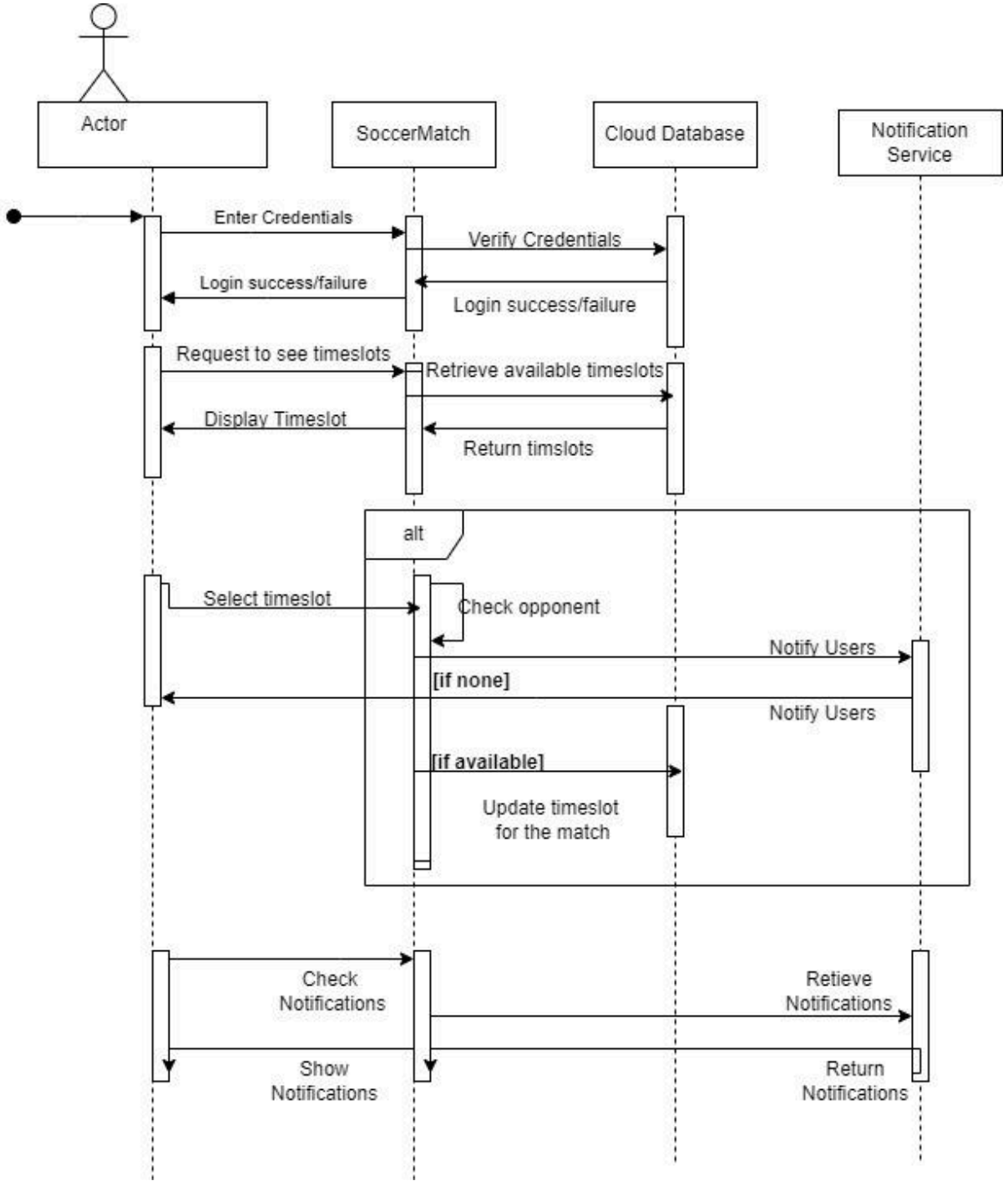

Figure 2: Sequence Diagram for Client-Server Interactions

Data Types

In our system text data will be held mostly, some binary data such as team logos can be added as well. As a mixed data we will hold notifications with both text and timestamp.

Text Data Types:

- User Credentials
- Match Records
- Rankings

Binary Data Types:

- Team Logos

Mixed Data Types:

- Notifications

Data-Flow Diagram

Context Data-Flow Diagram:

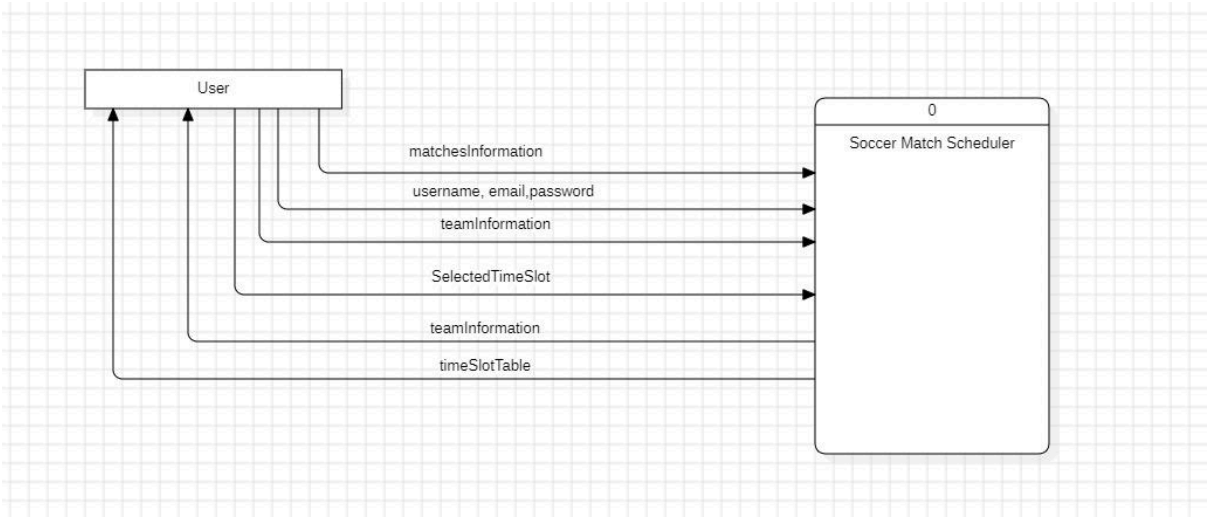

Figure 3: Context Data-Flow Diagram

In our context data-flow diagram the following information about our application that illustrated data exchange between the user and the application is shown.

The user has to send a username and password to login to the system. Same for registration. Each team has to create a user as a team in the system. When logged into the system a time slot table will be shown from the database. .If the user wants to reserve a match with another team to play then the opponent match information should be given to the system with a selected time slot. After the match ends, the scores should be provided as well for the rating table. If a team wants to modify the details, there is also an edit page.

Level-0 Data-Flow Diagram:

Our Level-0 Data-Flow diagram illustrates the details of SoccerMatch Scheduler data exchange. Showing where exactly the data will be transferred to in the application.

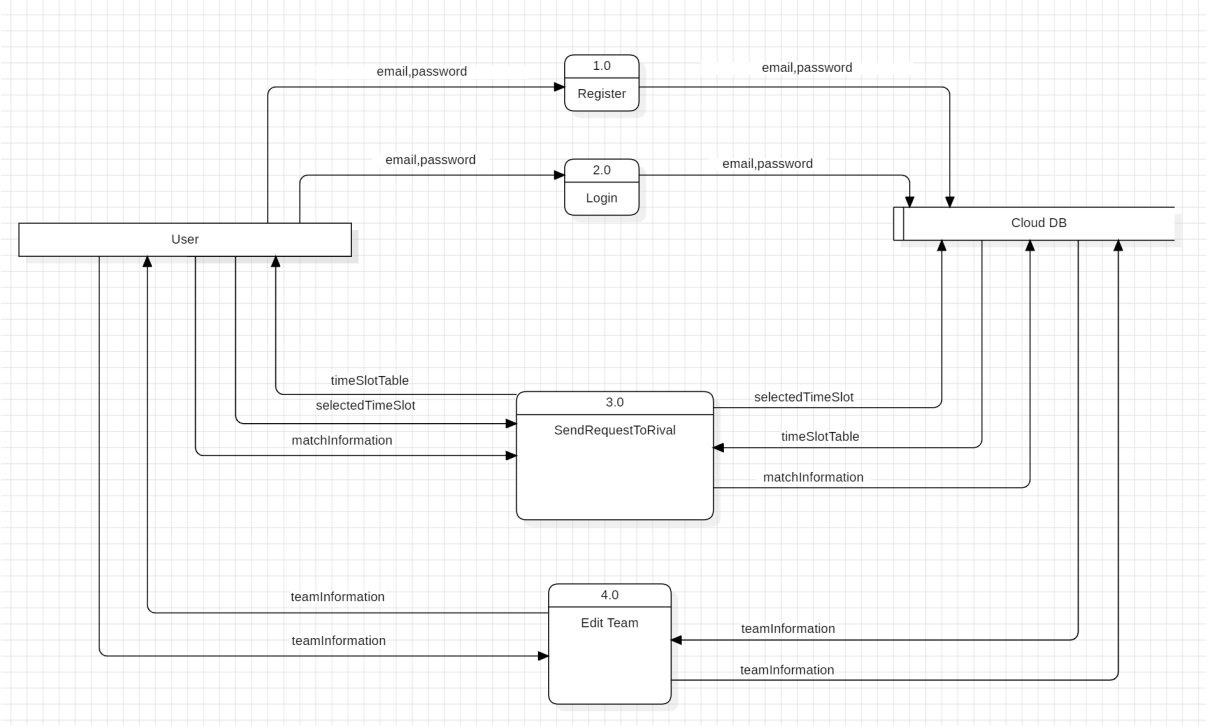

Figure 4: Level-0 Data-Flow Diagram

Computation

For the computation of our application in terms of the number of users, volume of cloud database and the estimated cost, we have considered our population in METU as 3000 students.

Estimated Students that will use the Application

Let’s say 20% of our students in the university will be an active soccer player and if each team consists of 6 people, 100 teams (users) will be the active users of the app. If 20% of these use the application daily, 20 teams will be the daily login value. Then we can consider that 20% of these users will be using the system simultaneously. Which will make 4 users at the same time.

Estimated Volume of Cloud Database

For 100 users in the cloud let consider the max memory size of 1KB for each user. (100 KB)

In our campus 4 time slots in a day are available to be played and in Morphou 6 time slots. Let’s consider 6 time slots to calculate the max. if 256 bytes are used for each timeslot, for a month we will have 6 X 30 X 256 = 46 KB for time slots. Then if we consider that 50% of slots will be used to play a match, meaning 90 matches is a month. If we say 256 bytes are used to store the rankings of teams that means 100 teams X 256 = 23KB for result. Therefore when we sum up, 169 KB.

Estimated Notification Volume

For a day we have considered 3 matches a day (50%) and if we say 2 of them are looking for a team then in a day 2 X 99 teams will be notified about the match and for a month 5940 notifications a month will be sent.

Estimated Cloud Cost

If t3.micro is assumed to be used for EC2 it costs 7$ for a month. For RDS for database 5GB of storage costs 10 cents for a month which is the minimum option. For SNS for notifications, it costs 50 cents for a million notifications meaning 0.003$ and if we prefer an email system as a notification system then it will cost 0.12$ for a month. In total 7.22$ is estimated to be the cost per month for the cloud usage cost of the application.

Expected Contribution

Elif Ilgın Savaş - Backend Development, Cloud Database Design

Will be responsible for implementing the logic of scheduling, database connection, integration with RDS MySQL and integration with the frontend.

Alper Kutay Ören - Frontend Development, Cloud Database Design

Will be responsible for the interface of the SoccerMatch Application, will retrieve user actions from the user and communicate with the backend. Also contribute to the design of the cloud model.

Yiğit Berk Atcı - Cloud Database Design

Will be the head of cloud design, deployment on RDS, management and configuration of the system in terms of cloud design. Also will contribute to parts of the frontend and backend if needed.

References

N/A