**Software Design Specification (SDS)**

**Project Name**:Ticket Arena  
**Prepared By**:Ahmed Ebaid - Ahmed Yasser - Abdulkhaliq Sarwat - Mohamed Saleh - Kareem Ezzaldin   
**Date**:9/11/2024

**1. Introduction Software Design Specification (SDS)**

**Project Name**:Ticket Arena  
**Prepared By**:Ahmed Ebaid - Ahmed Yasser - Abdulkhaliq Sarwat - Mohamed Saleh - Kareem Ezzaldin   
**Date**:9/11/2024

**1. Introduction**

**1.1 Purpose**

The purpose of this document is to describe the design, architecture, and technical specifications of the **Ticket arena platform**. It outlines the functionality, system components, and design decisions to be followed during the development process.

**1.2 Scope**

Ticket Arena system will offer a convenient platform to browse, select, and book tickets for events. The system will provide functionality for:

* browsing available football matches.
* filter matches by team, stadium, and ticket category.
* Password recovery and security measures
* view real-time ticket availability for each match
* view the store of the favorite team.
* selecting seats based on availability.
* making secure payments using Fawry or Aman.
* sending email confirmation after booking by system
* viewing highlights of the favorite team matches.
* allowing users to register and log in.
* viewing booking history.

**2. System Overview**

Ticket Arena is a standalone system that interacts with users and external payment services. The system architecture includes:

* **Frontend**: using React,HTML,JavaScript for user interface
* **Backend**: Django will handle the back-end
* **Database**: The storage for all persistent data.

**3. System Architecture**

**3.1 Architectural Design**

This project follows a **client-server architecture**, where:

* **Frontend** communicates with the backend using REST APIs
* **Backend** handles requests from the frontend, performs business logic, and interacts with the MySQL database to manage and retrieve data.

**3.2 Data Flow**

1. **User Interaction**: The user interacts with the UI to perform an action like browse upcoming matches, book tickets, view highlights, and access other features
2. **Request Processing**: The frontend sends an API request to the backend server.
3. **Data Handling**: The backend processes the request, interacts with the database, and performs necessary actions like filtering matches, update available tickets, and processing payment
4. **Response**: The backend sends the response back to the frontend, updating the UI.

**4. Database Design**

The system will store data in [insert database type, e.g., relational (MySQL), NoSQL (MongoDB)] with the following entities and relationships:

**Table 1**:Users

* + **Column 1**: user\_id: Primary Key
  + **Column 2**: username
  + **Column 3**: password
  + **Column 4** : Email
  + **Column 5**: Booked\_Matches

**Table 2**: Matches

* + **Column 1**: Match\_id (Primary key)
  + **Column 2**: team1\_id
  + **Column 3:** team 2\_id
  + **Column 4**: stadium\_id
  + **column 5** : Date

Matches can have multiple Tickets (One-to-Many).

Users can have multiple Bookings (One-to-Many).

Stadium can host multiple matches (One-to-Many)

**5. Technology Stack**

* **Frontend**: HTML, JavaScript, and React for an interactive, user-friendly interface.
* **Backend**: Django for a robust, secure backend server with business logic
* **Database**: MySQL for reliable relational data storage.
* **Hosting**: AWS, or Microsoft Asura

**6. Testing Plan**

**6.1 Unit Testing**

Each component (frontend and backend) will undergo unit testing to verify that individual functions and modules are performing correctly.

**6.2 Integration Testing**

Integration tests will validate that the frontend, backend, and database interact seamlessly, especially during operations like booking tickets, filtering matches, and viewing highlights.

**6.3 User Acceptance Testing (UAT)**

End users will be involved in testing the system to verify that it meets their requirements and expectations.

**6.4 Performance Testing**

Stress and load testing will be conducted to ensure the system can handle the required number of users and operations without degradation in performance.

**7. Conclusion**

The Ticket Arena is designed to fulfill the specified functional and non-functional requirements as described in this SDS. The design outlined here will ensure that the system is robust, scalable, and user-friendly, providing the intended value to its users.