SOFTWARE DESIGN DOCUMENT

FOR

MATHEMATICS CHALLENGE PROJECT

Githubt link

Date: / 06/2024.

Supervisor: Dr. Mary Nsabagwa

|  |  |  |
| --- | --- | --- |
| **NAME** | **REGESTRATION NUMBER** | **STUDENT NUMBER** |
| KAWEESI AGAPTUS | 23/U/09427/PS | 2300709427 |
| ARMAND SHEMATSI | 23/X/25074/PS | 2300725074 |
| NASSIWA HAFSWA | 23/U/23750 | 2300723750 |
| MALE MUSA | 23/U/10997/PS | 2300710997 |
| MUKABERA ALICE | 23/U/23678 | 2300723678 |

**Group G-11**

**TABLE OF CONTENTS**

[1. INTRODUCTION 1](#_Toc169286623)

[1.1 Purpose 1](#_Toc169286624)

[1.2 Scope 1](#_Toc169286625)

[1.3 Overview of the Document 2](#_Toc169286626)

[1.4 Reference Material 3](#_Toc169286627)

[1.5 Definitions and Acronyms 3](#_Toc169286628)

[2. SYSTEM OVERVIEW 5](#_Toc169286629)

[3. SYSTEM ARCHITECTURE 8](#_Toc169286630)

[3.1 Architectural Design 8](#_Toc169286631)

[3.2 Decomposition Description 9](#_Toc169286632)

[3.3 Design Rationale 15](#_Toc169286633)

[4. DATA DESIGN 16](#_Toc169286634)

[4.1 Data Description 16](#_Toc169286635)

[4.2 Data Dictionary 16](#_Toc169286636)

[5. COMPONENT DESIGN 22](#_Toc169286637)

[5.1. Summary of Algorithms 23](#_Toc169286638)

[6. HUMAN INTERFACE DESIGN 27](#_Toc169286639)

[6.1 Overview of user interface 27](#_Toc169286640)

[6.2 Screen images 28](#_Toc169286641)

[6.3 Screen Objects and Actions 32](#_Toc169286642)

[7. REQUIREMENTS MATRIX 33](#_Toc169286643)

**LIST OF FIGURES**

Figure 2.1 Use case diagram of the system ……………………………………………………....6

Figure 3.1 Architectural Design..............................................................................................…....7

Figure 3.2.1 Mathematics challenge system top level decomposition…………………………...10

Figure 3.2.2 School Management Decomposition………………………………………………..10

Figure 3.2.3 Participant Management Decomposition…………………………………………..11

Figure 3.2.4 Representative Management Decomposition……………………………………….11

Figure 3.2.5 Challenge Management Decomposition…………………………………………….12

Figure 3.2.6 Report Generation Decomposition…………………………………………………..12

Figure 3.2.7 context data flow diagram of the system…………………………………………….13

Figure 3.2.8 Level 1 data flow diagram of the system…………………………………………….14

Figure 4.2.1 Entity relationship model…………………………………………………………….19

**LIST OF TABLES**

Table 1 School…………………………………………………………………………………..15

Table 2 Representative………………………………………………………………………….16

Table 3 Participant……………………………………………………………………................16

Table 4 Challenge……………......................................................................................................16

Table 5 Attempt……………………….………………………………………………………...17

Table 6 Question……………...…………………………………………………………………17

Table 7 Report…………………………………………………………………………..18

Table 8 Rejectedparticipant……………………………………………………………..18

Table 9 Mark…………………………………………………………………………….18

# INTRODUCTION

## Purpose

This software design document describes the architecture and system design of the Mathematics Challenge System, a web-based system for a mathematics competition tailored for primary school children. It is intended for developers who will implement the system, administrators who will manage the system, and stakeholders who will review the system design.

## Scope

The Mathematics Challenge System will be designed to facilitate a national mathematics competition for primary school children. The software will manage the registration of schools and participants, the creation and execution of challenges, and the generation of reports and analytics.

**The goals of the system are to:**

1. Provide a seamless registration process for the challenge for schools and participants

2. Ensure the integrity and fairness of the competition by randomly selecting questions for each challenge attempt.

3. Automate the management of challenges, including timing and scoring.

4. Generate detailed performance reports for participants and schools.

5. Provide analytics to help understand performance trends and identify areas for improvement.

**The objectives of the system include**:

1. Developing a command line interface (CLI) for participants and school representatives to register and participate in challenges.

2. Implementing a web interface for administrators to manage challenges and view reports.

**The benefits of the system are**:

1. Streamlined competition management, reducing the administrative burden on organizers.

2. Enhanced participant experience with clear instructions and automated feedback.

3. Provision of accurate and timely performance reporting, helping schools and participants track their progress.

4. Provision of …Valuable insights through analytics, allowing for continuous improvement of the competition.

## Overview of the Document

This document has been organized into chapters as described below;

Chapter 1: This chapter describes the purpose of the document, the scope, the reference materials used in the document and the definitions and acronyms used with their meanings.

Chapter 2: This chapter describes the system overview i.e. the general description of the functionality, context and design of the system.

Chapter 3: This chapter describes the system architecture which comprises of the system architecture design, the decomposition of the system and the design rationale.

Chapter 4: This chapter describes the data design of the system which shows how the information domain of the system is transformed into data structures

Chapter 5: This chapter describes the component design of the system. It gives the functional description of each component in a systematic way.

Chapter 6: This chapter describes the human interface design of the system, the way users interact with the system and provides screen images for each of the components

## Reference Material

<https://drive.google.com/open?id=1qxp8v3pNxliFzvvUQhsmMk1AqP5463wA&usp=drive_fs>

## Definitions and Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Description** |
| SDD | Software Design Document |
| CLI | Command Line Interface |
| CRUD | Create Read Update Delete |
| ERD | Entity Relationship Diagram |
| PHP | Hypertext Preprocessor (Personal Home Page) |

**Definitions**

* Software Design Document

Is a document that describes the design of software created to facilitate analysis, planning implementation and decision making during the software development process.

* Use case diagram

Is a diagram which summarizes the details of the system and the actors (users) within the system.

* Entity Relationship Diagram (ERD)

It is a diagram which provides a visual starting point for the database design.

* Data flow diagram.

Is a diagram that represents the flow of data within the system.

* Hypertext Preprocessor (php)

Is an open source server side scripting language that is embedded into html and used for web development.

# SYSTEM OVERVIEW

The Mathematics Challenge System will facilitate a mathematics competition for primary school students. The system includes a client application for participants and school representatives, a server application for data processing and business logic, and a web interface for administrators. Participants register and attempt challenges through the CLI, while administrators manage challenges and view reports through the web interface.

**The system provides the following functionalities:**

All pupils in registered primary schools are eligible to take part in the competition, the schools are uploaded into the system by an administrator indicating the name, district, school registration number, email of representative and name of representative, these representatives are normally validated before being registered into the system.

A registered administrator uploads questions and answers into the web system and If a challenge is to consist of 10 questions, 100 questions should are uploaded such that for each attempt, the 10 questions are selected randomly from the 100 questions, the set questions are picked from an excel document (questions.xlxs) and they are accompanied by answers with accompanying marks from another excel document (answers.xlxs) for the corresponding questions and the two excel documents are uploaded into the web system by the administrator.

In order to open the competition, the administrator sets parameters including the date when the challenge will be opened and closed, the duration of the challenge will take, how many questions will be presented, as long as the dates are valid, the participants can log in and attempt the assignment.

At the close of the challenge, a report of answers for all attempted questions are sent to all participants and the first two winners are recognized on the website. Each participant is automatically sent an email providing them a PDF report of the report of right answers.

For a pupil to participate in the competition, they should register using a command line interface and before one can register, they are presented with a menu of instructions including

Register username, first name, last name, emailAddress, date\_of\_birth, school\_registration\_number and an image\_file.png

ViewChallenges displays challenges, which are still valid. If there are more than one valid challenges, a prospective participant will choose which competition they wish to participate in and one can participate in more than one challenge if they wish.

If the school registration number does not match registered school numbers, the pupil is informed. If there is a match, the record is added to a file and an email notification is sent to the

respective school representative reminding them to confirm the applicant.

School representatives log into the system via the command line interface to confirm the newly registered prospective participants via a menu item viewApplicants. The menu displays all applicants with their registration numbers. To reject or activate an applicant, the representative enters a menu item confirm yes/no username. If one is rejected, they are deleted from the file and moved into a database table rejected while those that are accepted are deleted from the file and inserted into the participant table in the database.

An email notification is sent for all cases. If one tries to register again under the same school after being rejected, they are informed on the command line interface and denied from registering under the school. Once accepted, the prospective participant can log in and see details of the challenges using viewChallenges command.

If the challenge is open, the participant can issue a command attemptChallenge challengeNumber to start the challenge. Questions are picked randomly from those that were issued. A participant has a maximum of three chances to attempt a challenge and each attempt presents random questions. The questions will be presented one by one and each time a question is presented, the number of remaining questions and time are indicated above the question.

If a wrong answer is given by a participant, 3 marks are deducted, while if a participant is not sure, they enter a negative or – and are awarded 0 for that question. Else, the participant gets the marks that were assigned to that particular question.

When the time for attempting the question expires, the participant challenge is closed and the participant is given their score and report. The report shows their scores and time taken, for each attempted question and the total time they have taken to complete the challenge.

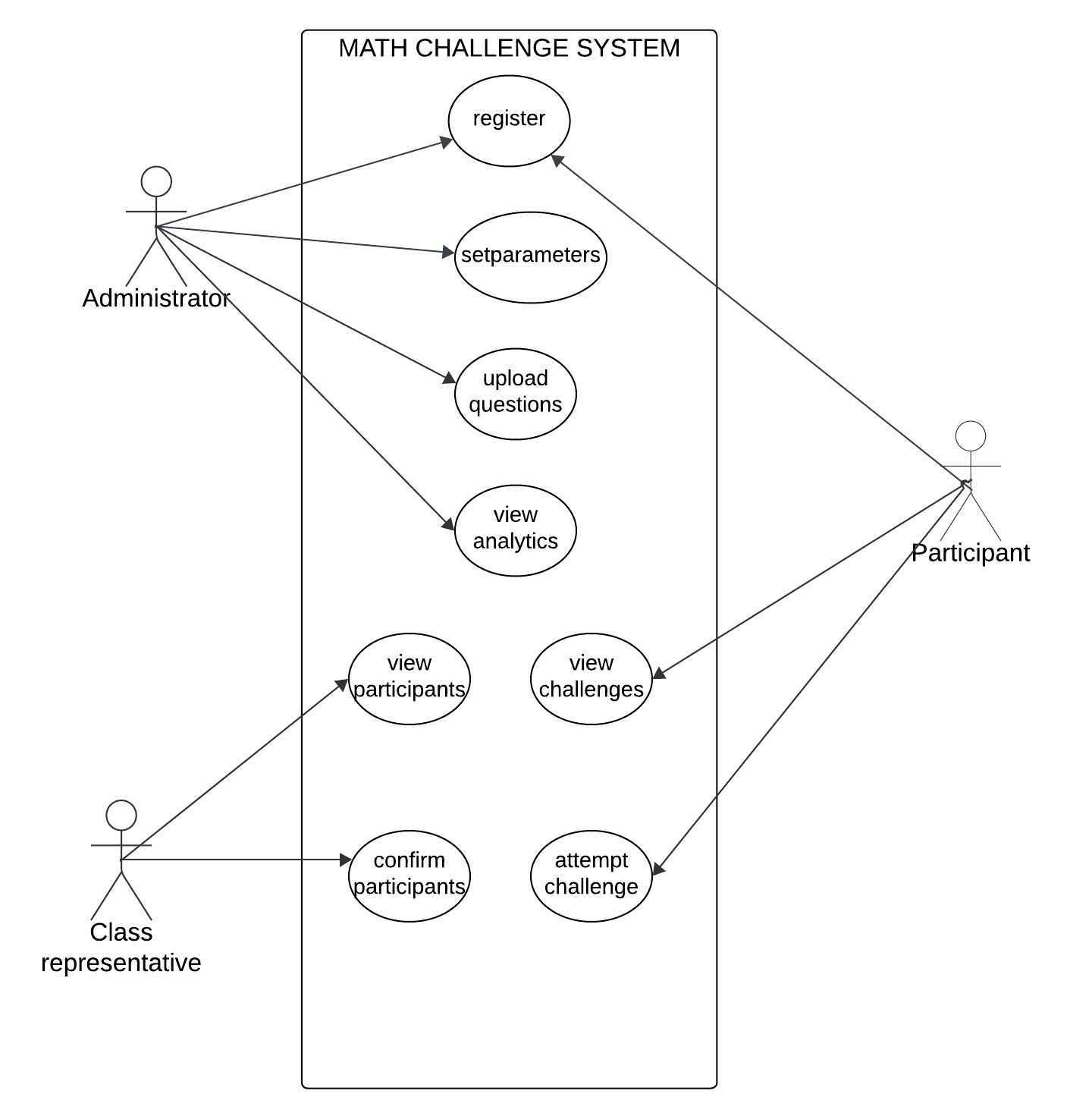


Figure 2. Use case diagram

It shows how the different actors interact with the system

# SYSTEM ARCHITECTURE

## Architectural Design

Developing a modular program structure is crucial for achieving the complete functionality of the system. This involves partitioning the system into high-level subsystems, each with specific responsibilities, and ensuring these subsystems collaborate effectively. The system will be divided into four main components, with their respective high-level subsystems:

The system architecture consists of three main components:

**1. Client Application (Java):** Provides menu options for participant and representative actions.

**2. Server Application (Java):** Handles file processing and database operations.

**3. Web Interface (PHP, Laravel):** Manages administrative tasks and displays reports.

**4. Database:** Stores all the data of the system.

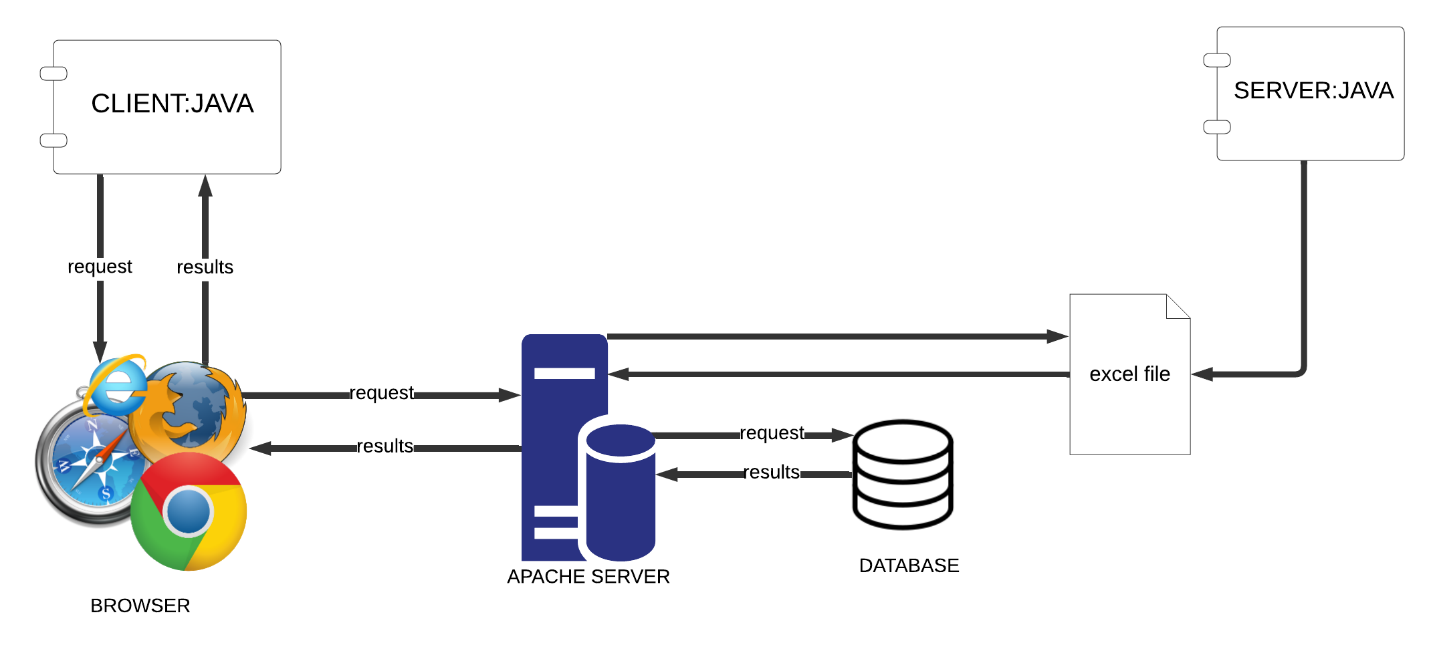


Figure 3. Architecture design

It represents the different modules or components of the system

## Decomposition Description

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an object-oriented description. For a functional description, put top-level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

**1. Client (Java):**

- User Interface Module: Handles interactions with participants and school representatives via the command line interface.

- Command Handler Module: Processes user commands and communicates with the server.

**2. Server (Java):**

- Database Manager Module: Manages interactions with the database for CRUD operations.

- File Manager Module: Handles Excel file uploads and data extraction.

- Challenge Manager Module: Manages challenge creation, random question selection, and scoring.

- Email Service Module: Sends notifications for various events.

**3. Web Interface (PHP, Laravel):**

- Admin Dashboard Module: Allows administrators to manage challenges and view reports.

- Report Generator Module: Generates participant score reports in PDF format.

- Analytics Module: Provides insights into question performance, school rankings, and more.

**4. Database:** The database stores all of the data for the system. Houses all necessary data tables, including schools, representatives, participants, challenges, questions, and attempts.

The relationships between the subsystems are as follows:

**User Interface Module and Command Handler Module:** The User Interface Module captures user inputs and forwards them to the Command Handler Module, which processes these inputs and communicates with the Server for necessary actions.

**Server and its Modules:** The Server acts as the core logic handler, coordinating actions among the Database Manager Module, File Manager Module, Challenge Manager Module, and Email Service Module to manage data, handle challenges, and send notifications.

**Admin Dashboard Module and Server:** The Admin Dashboard Module provides a web interface for administrators, allowing them to interact with the system via the Server.

**Report Generator Module and Analytics Module:** These modules interact with the Database Manager Module to generate reports and perform analytics, providing valuable insights and data-driven decisions.

**Database:** The Database serves as the central repository for all data, including school, representative, participant, challenge, question, and attempt information. It is accessed by the Database Manager Module for all CRUD operations.

This modular approach ensures each subsystem has a well-defined responsibility, promoting scalability, maintainability, and efficient collaboration between different parts of the system.

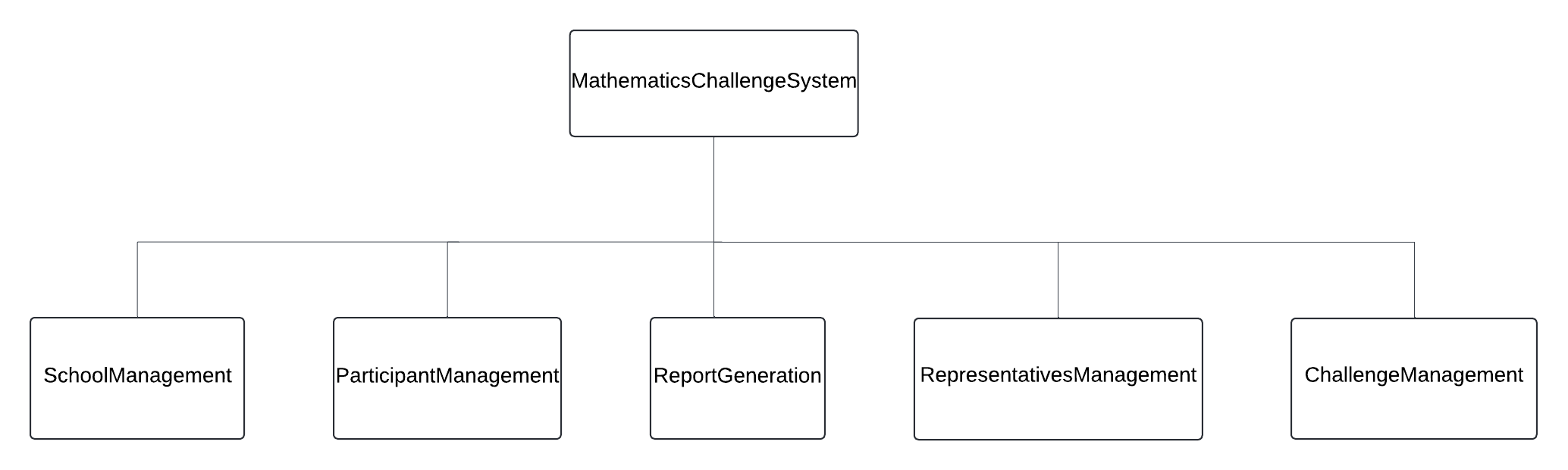


Figure 3.2. Mathematics challenge system top level decomposition

Shows the major processes that the mathematics challenge system will perform

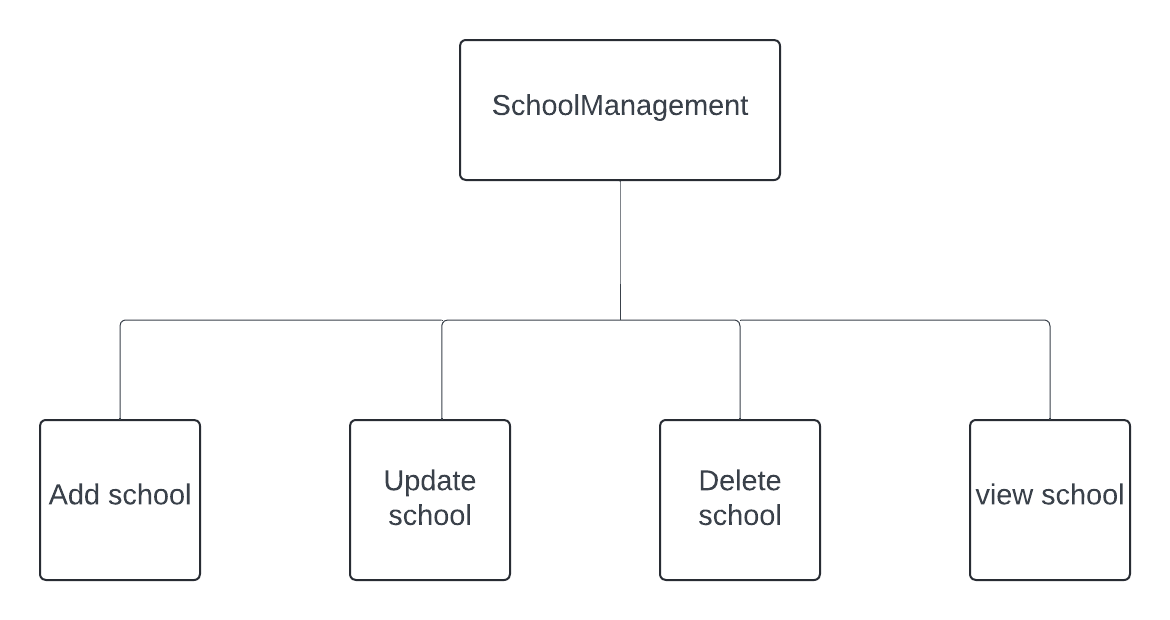


Figure 3.2. 2 School Management Decomposition

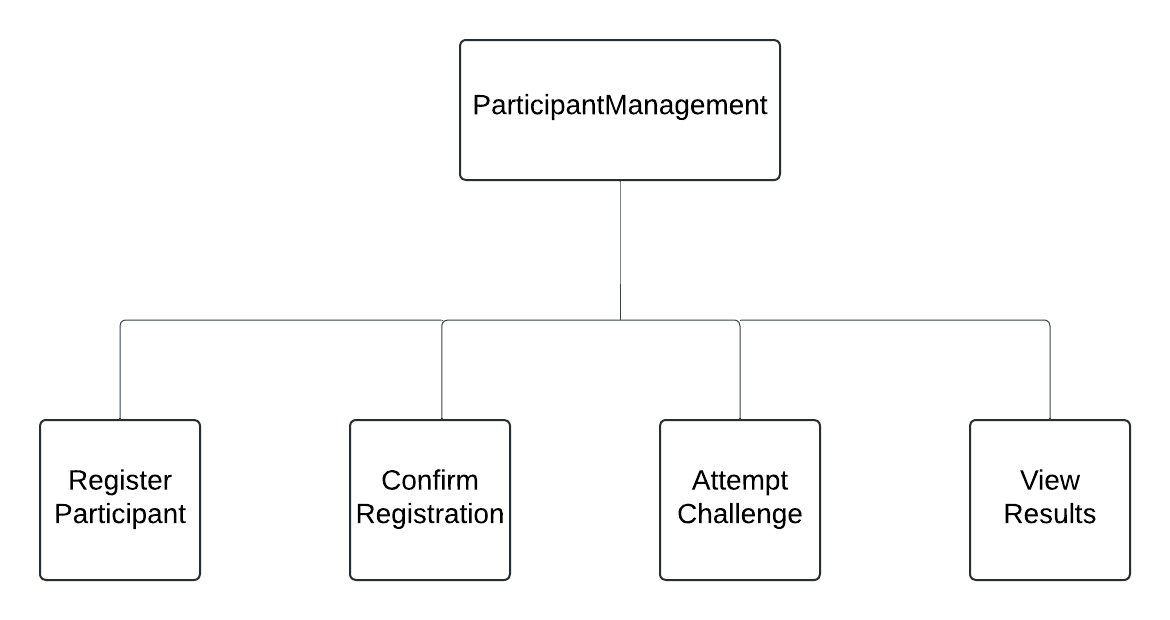


Figure 3.2. 3 Participant Management Decomposition

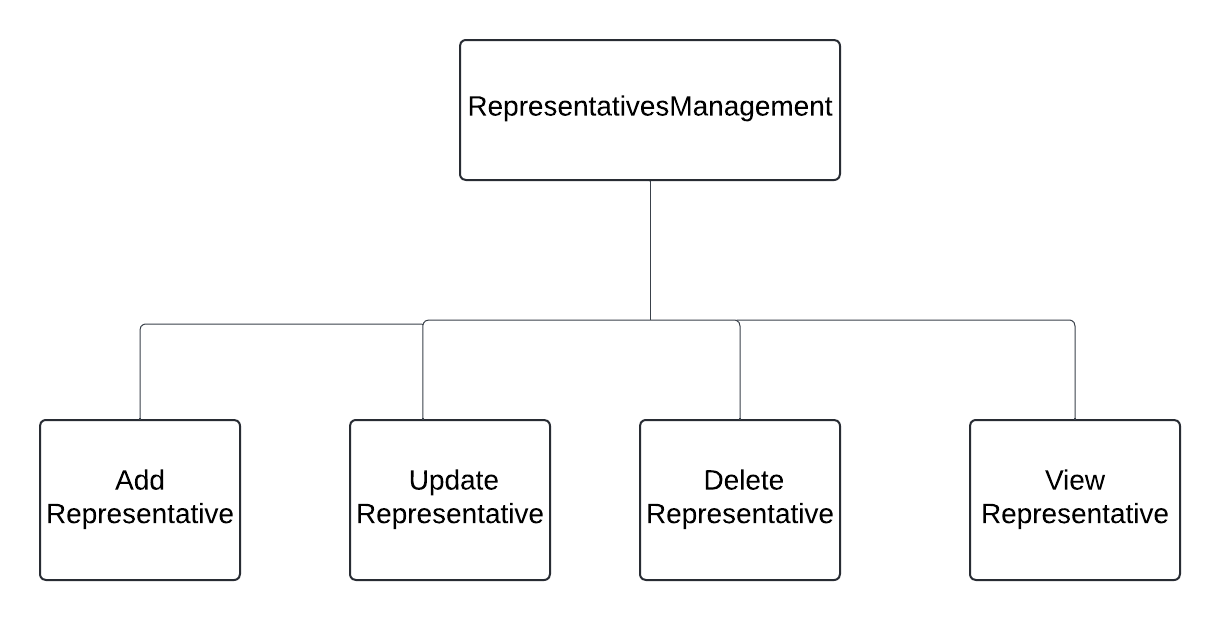


Figure 3.2. 4 Representative Management Decomposition

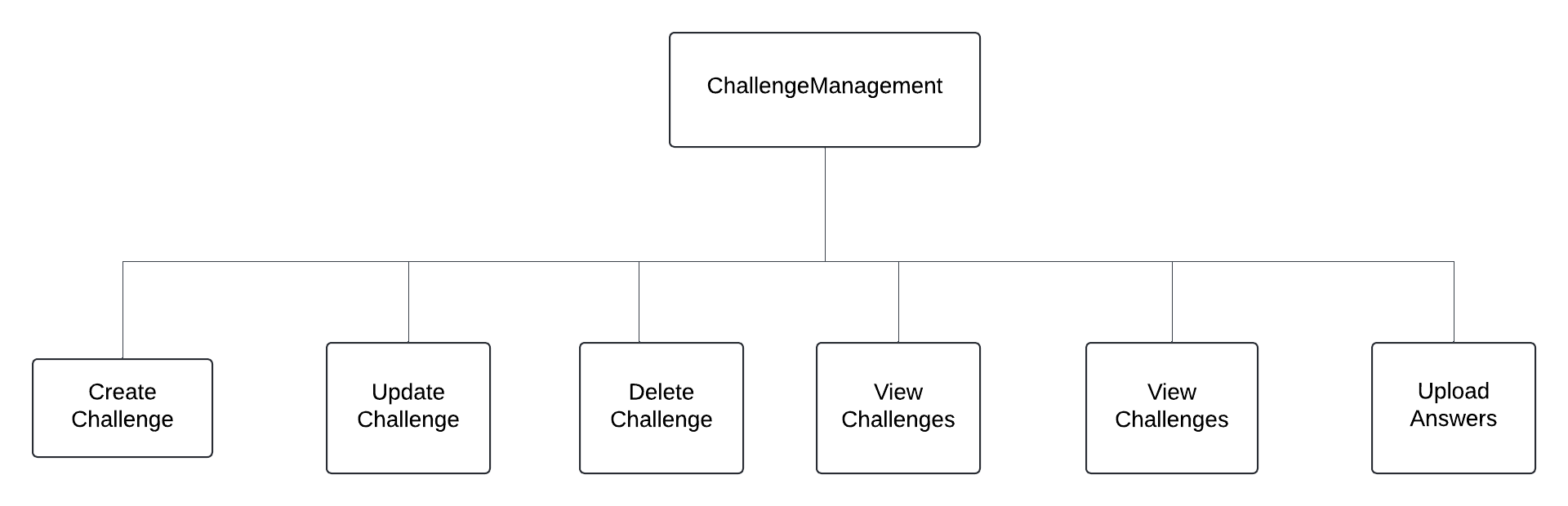


Figure 3.2. 5 Challenge Management Decomposition

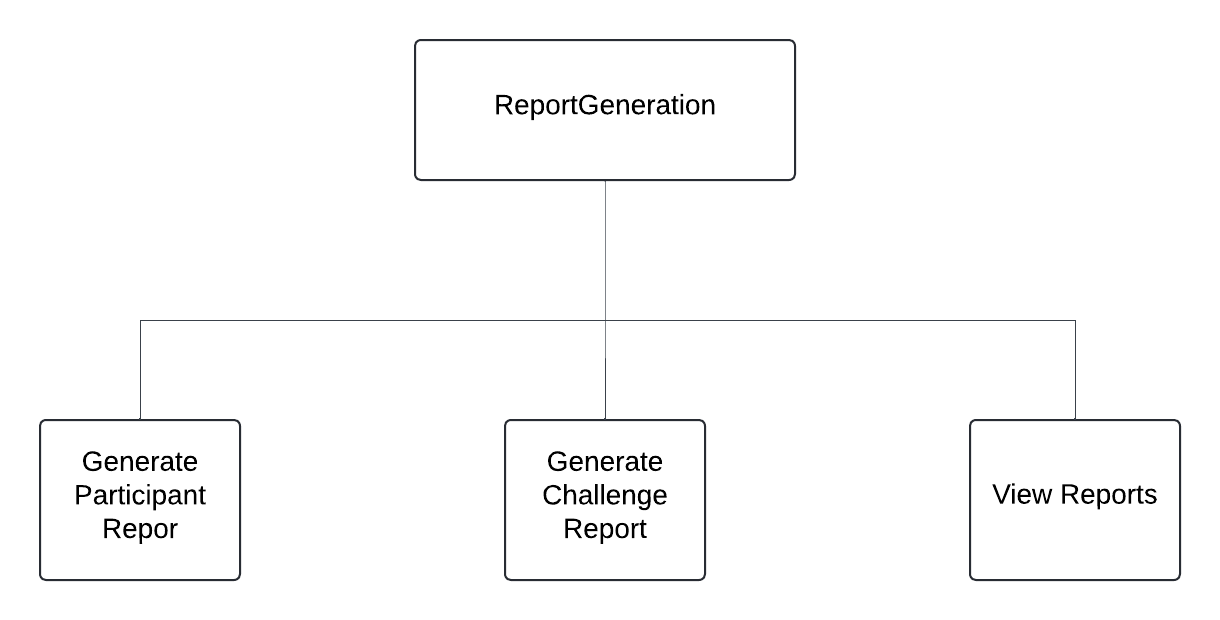


Figure 3.2. 6 Report Generation Decomposition

Shows the major processes that the mathematics challenge system will perform.



Figure 3.2. context Data flow Diagram

It provides a broad view of the system and external entities.



Figure 3.2. level 1 data flow diagram

It describes the flow of data within the system

## Design Rationale

The architecture was chosen to provide a scalable, maintainable, and secure system. The decomposition into subsystems allows for easier development, testing, and maintenance.

The client-server approach also enables concurrent access to the system and promotes a clear separation of concerns between the client interface and server-side processing.

# DATA DESIGN

## Data Description

Data will be stored in a relational database with the following main tables:

1. School: Stores school information.
2. Representative: Stores representative details and their association with schools.
3. Participant: Stores participant registration details and their status.
4. Challenge: Stores details about each challenge.
5. Question: Stores questions and answers.
6. Attempt: Stores records of participant attempts and their scores.
7. Rejected applicant: Stores details of participants who were rejected during the registration process.
8. Mark: Stores marks awarded for each question

## Data Dictionary

Table 1 school

Describes the data of the school

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| school\_id | INT(6) | Primary key |
| district | VARCHAR | District of the school |
| name | VARCHAR | Name of the school |
| registration\_number | VARCHAR | Unique registration number |
| Representative-id | INT(3) | Foreign key referencing the representative table |

Table 2 Representative

Describes the data of the representative of the school

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| Representative-id | INT(3) | Primary key |
| email | VARCHAR | Email of the representative |
| name | VARCHAR | Name of the representative |

Table 3 Participant

Describes the participant’s data

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT(3) | Primary key |
| date\_of\_birth | DATE | Date of birth of the participant |
| email | VARCHAR | Email of the participant |
| first\_name | VARCHAR | First name of the participant |
| last\_name | VARCHAR | Last name of the participant |
| school\_id | INT(6) | Foreign key referencing Schools table |
| status | VARCHAR | Status of the participant (Pending/Confirmed) |
| username | VARCHAR | Username of the participant |
| password | VARCHAR(8) | Password of the participant |

Table 4 challenge

Describes the data of the challenge

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT | Primary key |
| duration | TIME | Duration of the challenge |
| start\_date | DATE | Start date of the challenge |
| end\_date | DATE | End date of the challenge |
| name | VARCHAR | Name of the challenge |
| question | TEXT | Questions attempted in the challenge |

Table 5 Attempt

Describes the data of the attempt

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT(3) | Primary key |
| challenge\_id | INT(2) | Foreign key referencing Challenges table |
| participant\_id | INT | Foreign key referencing Participants table |
| score | INT | Score of the participant |
| time\_taken | TIME | Time taken to complete the challenge |
| Date | DATE | The date of attempting the challenge |

*Table 6 question*

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT | Primary key |
| answer | VARCHAR | Correct answer |
| challenge\_id | INT | Foreign key referencing Challenges table |
| marks | INT | Marks for the question |
| question\_text | TEXT | Text of the question |

Table 7 Report

Describe the data of the report

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Type** | **Description** |
| **Report-id** |  | Primary key |
| **scores** | INT | Scores of a participant |
| **Challenge-time** | DATE | Total time taken to complete the challenge |
| **School ranking** | INT | Ranks according to the performance of a school |
| **Attempting-time** | DATE | Time taken to attempt a question |

Table 8 RejectedParticipant

Describes the data of the rejected participants

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT | Primary key |
| participant\_id | INT | Foreign key referencing Participant table |

*Table 9 mark*

|  |  |  |
| --- | --- | --- |
| Attribute | Type | Description |
| id | INT(3) | Primary key |
| attempt\_id | INT (3) | Foreign key referencing Attempt table |
| question\_id | INT | Foreign key referencing Question table |
| marks\_awarded | INT | Marks awarded for the question |

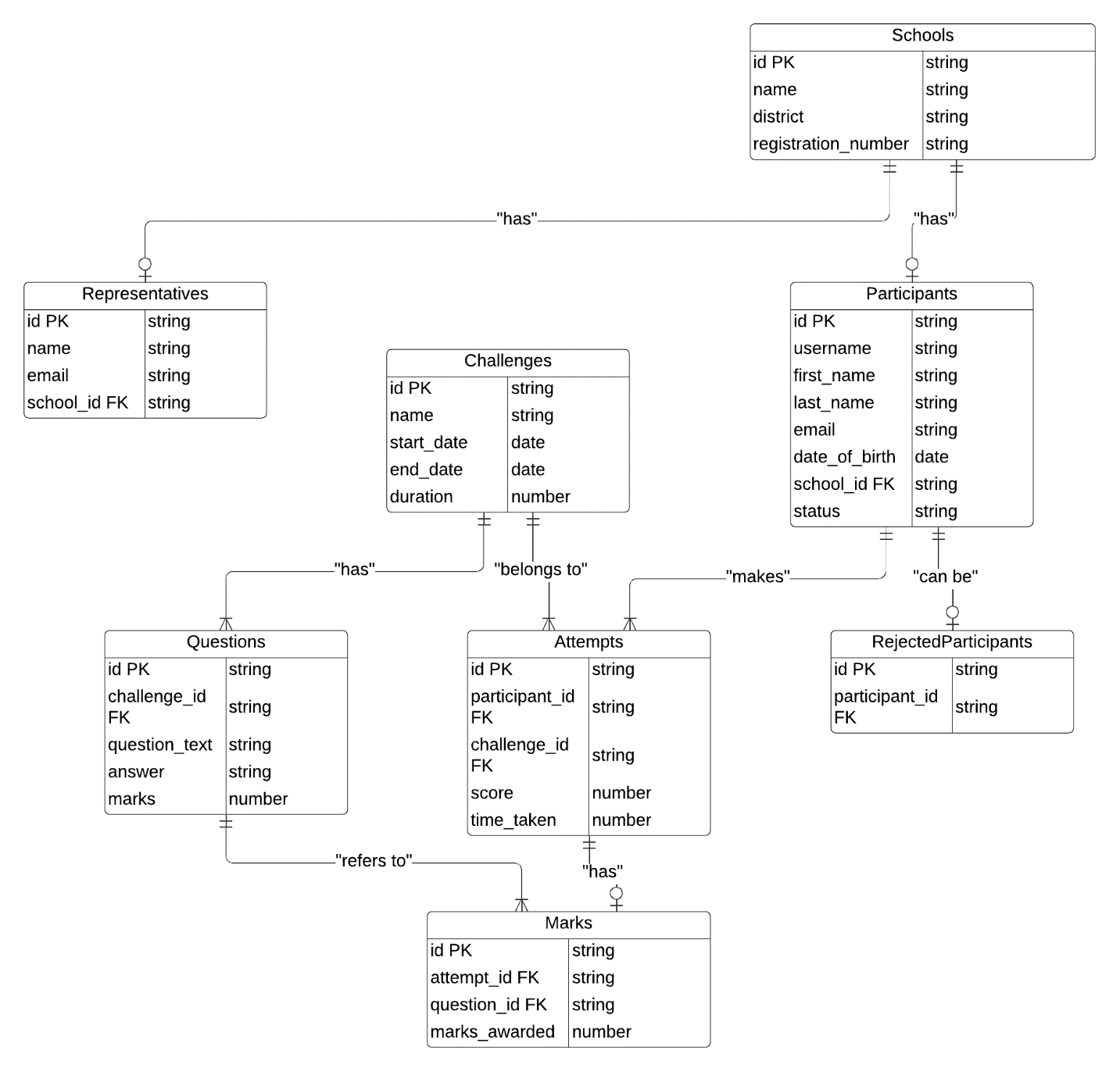


Figure 4.2.1. Entity relationship model

It provides a visual starting point for the database which is to be used to determine the information system of the organization

Functions and Function Parameters

* RegisterParticipant (username, first\_name, last\_name, email, date\_of\_birth, school\_registration\_number, image\_file): Registers a new participant in the system.
* viewChallenges (): Displays the list of currently active challenges.
* attemptChallenge (challenge\_number): Starts an attempt for the specified challenge.
* confirmApplicant(username): Confirms the registration of a participant by a school representative.
* uploadQuestions(file\_path): Uploads questions from an Excel file.
* uploadAnswers(file\_path): Uploads answers from an Excel file.
* createChallenge(name, start\_date, end\_date, duration): Creates a new challenge with the specified parameters.
* sendEmail(recipient, subject, message): Sends an email to the specified recipient.
* generateReport(participant\_id): Generates a report for the specified participant.
* viewReports(): Allows administrators to view all generated reports.
* generateAnalytics (): Generates various analytics reports for administrators.

# COMPONENT DESIGN

This section provides a detailed overview of the algorithms used by each component in the Mathematics challenge system.

## 5.1. Summary of Algorithms

**1.registerParticipant(username,first\_name,last\_name,email,date\_of\_birth,**

**school\_registration\_number, image\_file)**

**pseudocode**

Function registerParticipant(username, first\_name, last\_name, email, date\_of\_birth, school\_registration\_number, image\_file)

If school\_registration\_number not in Schools table

Print "Invalid school registration number"

Return

End If

participant\_id = Generate unique ID

Insert into Participants (participant\_id, username, first\_name, last\_name, email, date\_of\_birth, school\_registration\_number, status)

Save image\_file with participant\_id

Send email to school representative for confirmation

Print "Registration successful, pending confirmation"

End Function

**2.viewChallenges()**

**Pseudocode**

Function viewChallenges()

current\_date = GetCurrentDate()

challenges = Select \* from Challenges where start\_date <= current\_date and end\_date >= current\_date

Print challenges

End Function

**3.attemptChallenge(challenge\_number)**

**pseudocode**

Function attemptChallenge(challenge\_number)

If challenge\_number not in Challenges table

Print "Invalid challenge number"

Return

End If

questions = Select \* from Questions where challenge\_id = challenge\_number

random\_questions = Randomly select 10 questions from questions

For each question in random\_questions

Print question

answer = GetUserInput()

If answer is correct

score += question.marks

Insert into Marks (attempt\_id, question\_id, marks\_awarded) values (current\_attempt\_id, question.id, question.marks)

Else If answer is "-"

score += 0

Insert into Marks (attempt\_id, question\_id, marks\_awarded) values (current\_attempt\_id, question.id, 0)

Else

score -= 3

Insert into Marks (attempt\_id, question\_id, marks\_awarded) values (current\_attempt\_id, question.id, -3)

End If

End For

time\_taken = Calculate time taken for the challenge

Insert into Attempts (participant\_id, challenge\_number, score, time\_taken)

Print "Challenge completed. Your score: " + score

End Function

**4.confirmApplicant(username)**

**pseudocode**

Function confirmApplicant(username)

participant = Select \* from Participants where username = username and status = "Pending"

If participant is not found

Print "No pending applicant with the given username"

Return

End If

Print participant details

confirmation = GetUserInput("Confirm applicant? (yes/no)")

If confirmation == "yes"

Update Participants set status = "Confirmed" where username = username

Print "Applicant confirmed"

Else

Insert into Rejected\_Participants (participant\_id, details)

Delete from Participants where username = username

Print "Applicant rejected"

End If

End Function

**5. uploadQuestions(file\_path)**

**pseudocode**

Function uploadQuestions(file\_path)

questions = ReadExcelFile(file\_path)

For each question in questions

Insert into Questions (challenge\_id, question\_text, answer, marks)

End For

Print "Questions uploaded successfully"

End Function

**6.uploadAnswers(file\_path)**

**pseudocode**

Function uploadAnswers(file\_path)

answers = ReadExcelFile(file\_path)

For each answer in answers

Update Questions set answer = answer where question\_id = answer.question\_id

End For

Print "Answers uploaded successfully"

End Function

**7.createChallenge(name, start\_date, end\_date, duration)**

**pseudocode**

Function createChallenge(name, start\_date, end\_date, duration)

challenge\_id = Generate unique ID

Insert into Challenges (challenge\_id, name, start\_date, end\_date, duration)

Print "Challenge created successfully"

End Function

**8.sendEmail(recipient, subject, message)**

**pseudocode**

Function sendEmail(recipient, subject, message)

email = CreateEmail(recipient, subject, message)

Send email

Print "Email sent to " + recipient

End Function

**9. generateReport(participant\_id)**

**pseudocode**

Function generateReport(participant\_id)

participant = Select \* from Participants where id = participant\_id

attempts = Select \* from Attempts where participant\_id = participant\_id

report = GeneratePDFReport(participant, attempts)

Send email with report to participant.email

Print "Report generated and sent to " + participant.email

End Function

**10. viewReports()**

**pseudocode**

Function viewReports()

reports = GetAllReports()

Display reports

End Function

**11.generateAnalytics()**

**pseudocode**

Function generateAnalytics()

analytics = Generate various analytics reports

Display analytics

End Function

# HUMAN INTERFACE DESIGN

## Overview of user interface

Describe the functionality of the system from the user’s perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

The Mathematics Challenge System provides different interfaces for administrators, school representatives, and participants.

**Administrators:**

* **Login:** Admins log into the system via a web interface.
* **Dashboard:** Access the admin dashboard to manage challenges, upload questions, view reports, and analytics.
* **Manage Challenges:** Create, update, and delete challenges. Upload questions and answers from Excel files.
* **View Reports:** Generate and view detailed reports on participants' performances and overall challenge statistics.
* **Analytics:** Access various analytics to understand performance trends and rankings.

**School Representatives:**

* **CLI Login:** Representatives log into the system via a command-line interface.
* **Validate Participants:** View and validate participant registrations from their respective schools.
* **View Applicants:** Access the list of students who have applied and take actions to confirm or reject their applications.

**Participants:**

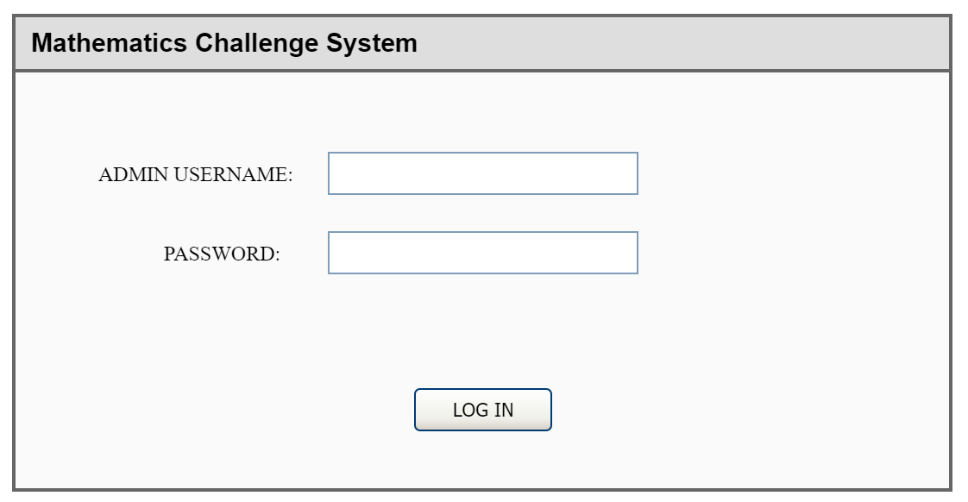
* **CLI Registration:** Participants register using the command-line interface by providing their details.
* **View Challenges:** See the list of available challenges they can participate in.
* **Attempt Challenge:** Start a challenge and answer questions within a time limit.
* **View Results:** After completing a challenge, participants can view their scores and feedback.

The system is designed to be intuitive and user-friendly, with clear instructions and feedback at each step.

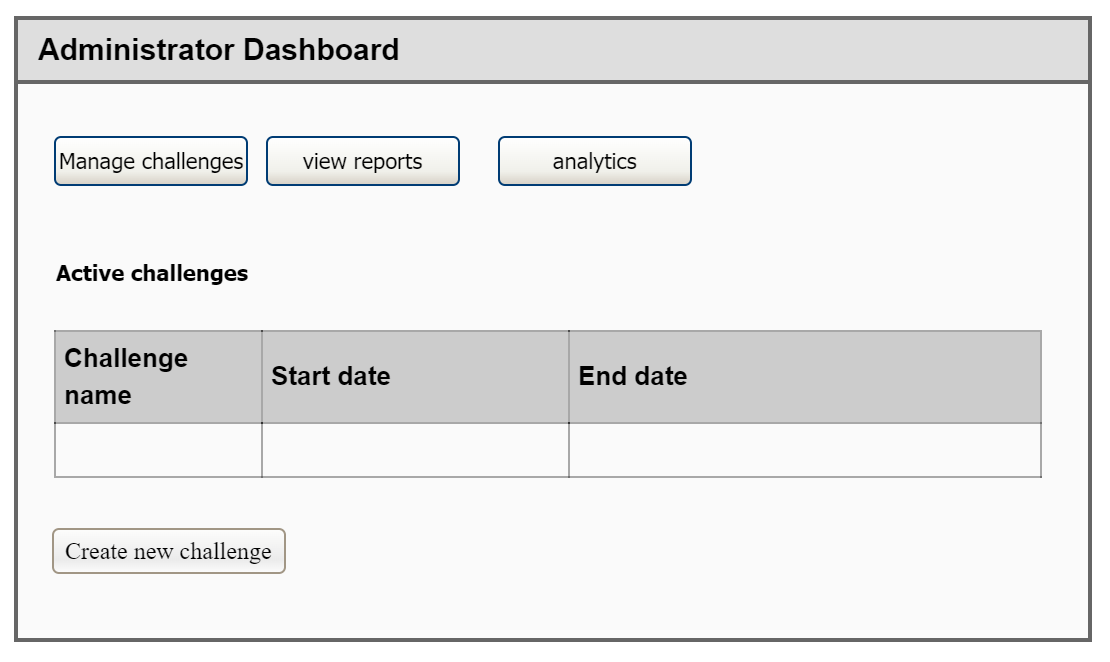
## Screen images

Below are some sample screen images illustrating how different parts of the Mathematics Challenge System might appear to users. These images are provided to give an idea of the user interface and user experience.

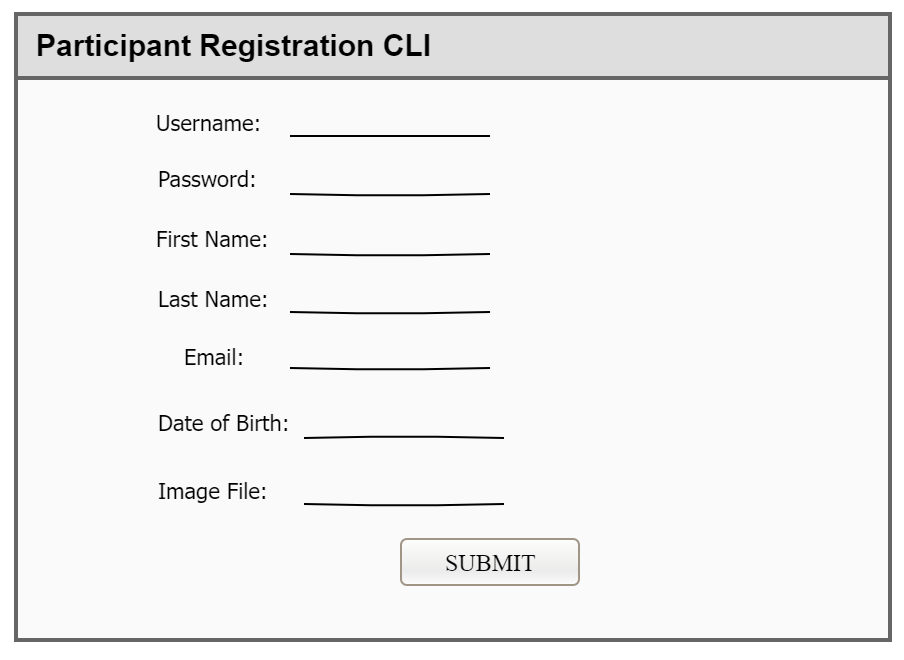
**1. Administrator Login Screen**



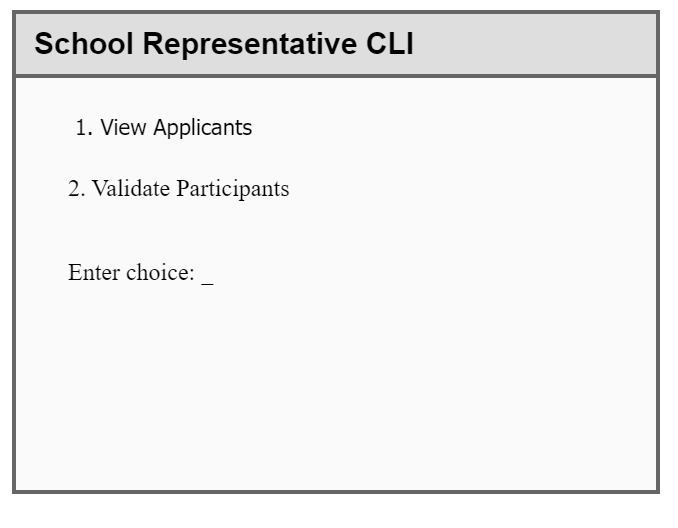
**2.Administrator dashboard**



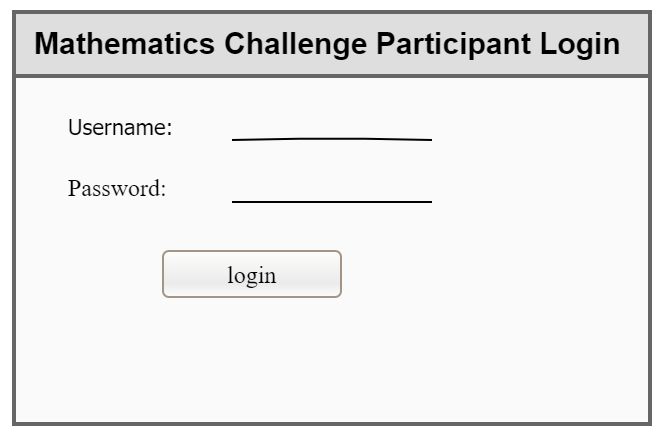
3.Participant registration CLI



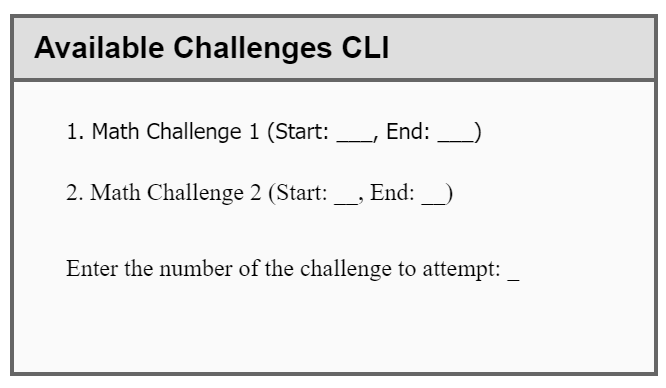
4.School Representative CLI



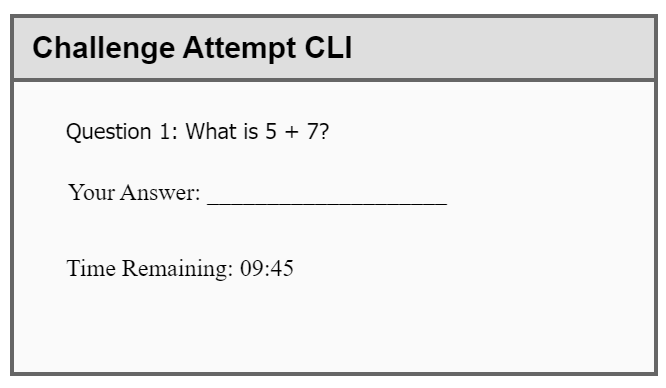
5. Mathematics Challenge Participant Login



6. **Available Challenges CLI**



7.Challenge Attempt CLI



## Screen Objects and Actions

Here we discuss the objects on each screen and the actions associated with those objects:

1. Administrator Login Screen:

* Username field: Text input for the administrator's username.
* Password field: Text input for the administrator's password (masked).
* Login button: Submits the login form for authentication.

1. Administrator Dashboard:

* Navigation menu: Allows navigation to different sections such as Manage Challenges, View Reports, and Analytics.
* Summary of active challenges: Displays an overview of currently active challenges.
* Create New Challenge button: Opens the Create Challenge screen.

1. Participant Registration CLI:

* Username field: Text input for the participant's username.
* First Name field: Text input for the participant's first name.
* Last Name field: Text input for the participant's last name.
* Email field: Text input for the participant's email address.
* Date of Birth field: Text input for the participant's date of birth.
* School Registration Number field: Text input for the participant's school registration number.
* Image File field: Text input for the file path to the participant's image.
* Submit button: Submits the registration form for processing.

1. Challenge Attempt CLI:

* Question text: Displays the current question.
* Answer input: Text input for the participant's answer.
* Time Remaining: Displays the remaining time for the challenge attempt.

1. School Representative CLI:

* Menu options: Allows the representative to choose between viewing applicants and validating participants.

1. View Challenges CLI:

* List of available challenges: Displays the challenges that the participant can attempt.
* Challenge selection input: Allows the participant to choose a challenge to attempt by entering its number.

1. Participant Login Screen:

* Username field: Text input for the participant's username.
* Password field: Text input for the participant's password (masked).
* Login button: Submits the login form for authentication.

This screen is essential for ensuring that only registered participants can access the participant-specific functionalities of the system.

# REQUIREMENTS MATRIX

Below is a cross-reference matrix that maps the components and data structures of the Mathematics Challenge System to the functional requirements specified in the Software Requirements Specification (SRS) document. This table helps ensure that all functional requirements are adequately addressed by the system components.

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement ID | Description | Component | Data Structure |
| FR1 | User authentication for administrators and participants | Authentication Module | Users Table |
| FR2 | Create and manage challenges | Challenge Management | Challenges Table, Questions Table |
| FR3 | Register participants | Participant Management | Participants Table |
| FR4 | Validate participants by school representatives | Representatives Management | Participants Table |
| FR5 | Attempt challenges | Challenge Attempt Module | ChallengeAttempts Table, Questions Table |
| FR6 | Generate reports for participants and challenges | Report Generation | Participants Table, Challenges Table, ChallengeAttempts Table |
| FR7 | View active and past challenges | Challenge Management | Challenges Table |
| FR8 | Manage school representatives | Representatives Management | Representatives Table |
| FR9 | Upload challenge questions and answers | Challenge Management | Questions Table, Answers Table |
| FR10 | Track challenge attempt time | Challenge Attempt Module | ChallengeAttempts Table |

Explanation of Components and Data Structures:

1. Authentication Module:

- Users Table: Stores user credentials and roles for both administrators and participants.

2. Challenge Management:

- \*\*Challenges Table: Stores details about each challenge.

- Questions Table: Stores questions related to each challenge.

- Answers Table: Stores correct answers to the challenge questions.

3. Participant Management:

- Participants Table: Stores participant registration details and status.

4. Representatives Management:

- Participants Table: Used by representatives to validate participants.

- Representatives Table: Stores details about school representatives.

5. Challenge Attempt Module:

- ChallengeAttempts Table: Records details of each participant's challenge attempt.

- Questions Table: Retrieves questions for the challenge attempt.

6. Report Generation:

- Participants Table: Used to generate participant reports.

- Challenges Table: Used to generate challenge reports.

- ChallengeAttempts Table: Used to generate detailed attempt reports.

By mapping each requirement to its corresponding system components and data structures, we ensure that all aspects of the SRS are covered, and we provide a clear path for implementation and verification.