# SOFTWARE REQUIREMENT SPECIFICATION

## ASSESSMENT PLATFORM

| | |
|---|---|
| **Name** | MOUNISH R |
| **Roll no** | 7376221CS232 |
| **Seat no** | **184** |
| **Project id** | **1** |
| **Problem Statement** | An interface for assessment that should contain mcq, short answer and match type of question and compiler. |

## TECHNICAL COMPONENT:

| **Component** | **Tech Stack** |
|---|---|
| Frontend | Angular(Js Framework) |
| Backend | Express.js(Web framework for Node.js)<br>Node.js(Javascript runtime environment) |
| Database | MongoDB(NOSQL Database**)** |
| API | RESTFul API / GraphQL APIs |

**IMPLEMENTATION TIMELINE:**

| Phase | Deadline | Status | Notes |
|---|---|---|---|
| Stage 1 | **20/07/2024** | Completed | Planning and requirement |
| Stage 2 | **24/08/2024** | Completed | Design and Prototyping |
| Stage 3 | **26/09/2024** | Completed | DB Designing |
| Stage 4 | **24/10/2024** | Completed | Backend Implementation |
| Stage 5 | | Not Started | Testing & Implementation |
| Stage 6 | | Not Started | Deployment |

## 1. INTRODUCTION :

### 1.1 Purpose :

The purpose of developing an assessment platform with multiple question types (such as MCQs, short answers, and match types) and a compiler component is to provide a versatile tool for educational institutions, companies, and individuals. It aims to streamline the process of creating and administering assessments, facilitating educational, professional, and self-assessment needs.

## 1.2. Scope of Project :

- The project aims to develop an assessment platform featuring various question types, compiler integration, security measures, feedback/reporting systems, scalability, and documentation/support.

- The scope of the project involves creating an assessment platform with features like multiple question types (MCQs, short answers, match types), question bank management, an assessment engine for delivery and scoring, compiler integration for programming questions, security measures, feedback/reporting systems, testing/iteration for refinement,deployment/maintenance,and documentation/support.

## 2. SYSTEM OVERVIEW :

### 2.1 User Side :

- Authentication for secure login.
- Assessment taking with various question types.
- Real-time feedback on performance.
- Compiler integration for code assessments (if applicable).
- Intuitive and accessible user interface.

### 2.2 Admin Side :

- Secure authentication for administrators.
- Question bank management for creating, editing, and organizing questions.
- Assessment creation with customization options.
- Participant management including account creation and activity tracking.
- Reporting and analytics for evaluating participant performance and assessment effectiveness.
- Platform configuration for settings and permissions management.
- Access to support and documentation resources.

**2.3. Features :**

1. User Authentication: Secure login system for users and administrators.
2. Assessment Creation: Ability for administrators to create assessments with customizable settings.
3. Question Bank Management: Management of a repository of questions, including creation and editing.
4. Real-Time Assessment: Immediate feedback on performance for users.
5. Reporting and Analytics: Generation of comprehensive reports on participant performance and assessment statistics.

**3.FUNCTIONAL REQUIREMENTS :**

1.User Authentication:

- Users and administrators must be able to register new accounts.
- Registered users must be able to log in securely.
- Administrators should have additional privileges for managing assessments and users.

2.Assessment Creation:

- Administrators should be able to create new assessments.
- They should be able to specify details such as title, duration, question types, and scoring criteria.

3.Question Bank Management:

- Questions can be selected from the question bank or created a new.
- Administrators should have the ability to manage a repository of questions.
- They should be able to create, edit, categorize, and tag questions.
- Questions should support various formats like multiple-choice, short answer, match type, and programming questions.

**4.NON-FUNCTIONAL REQUIREMENTS :**

1. Performance:

- The website should respond quickly to user interactions, with minimal latency.

- It should support a large number of concurrent users without significant performance degradation.

2.Reliability:

- The website should be available and operational at all times, with minimal downtime for maintenance or updates.
- It should have mechanisms in place to handle and recover from system failures gracefully.

3.Usability:

- The user interface should be intuitive and easy to navigate, requiring minimal training for users.
- It should adhere to accessibility standards to accommodate users with disabilities.

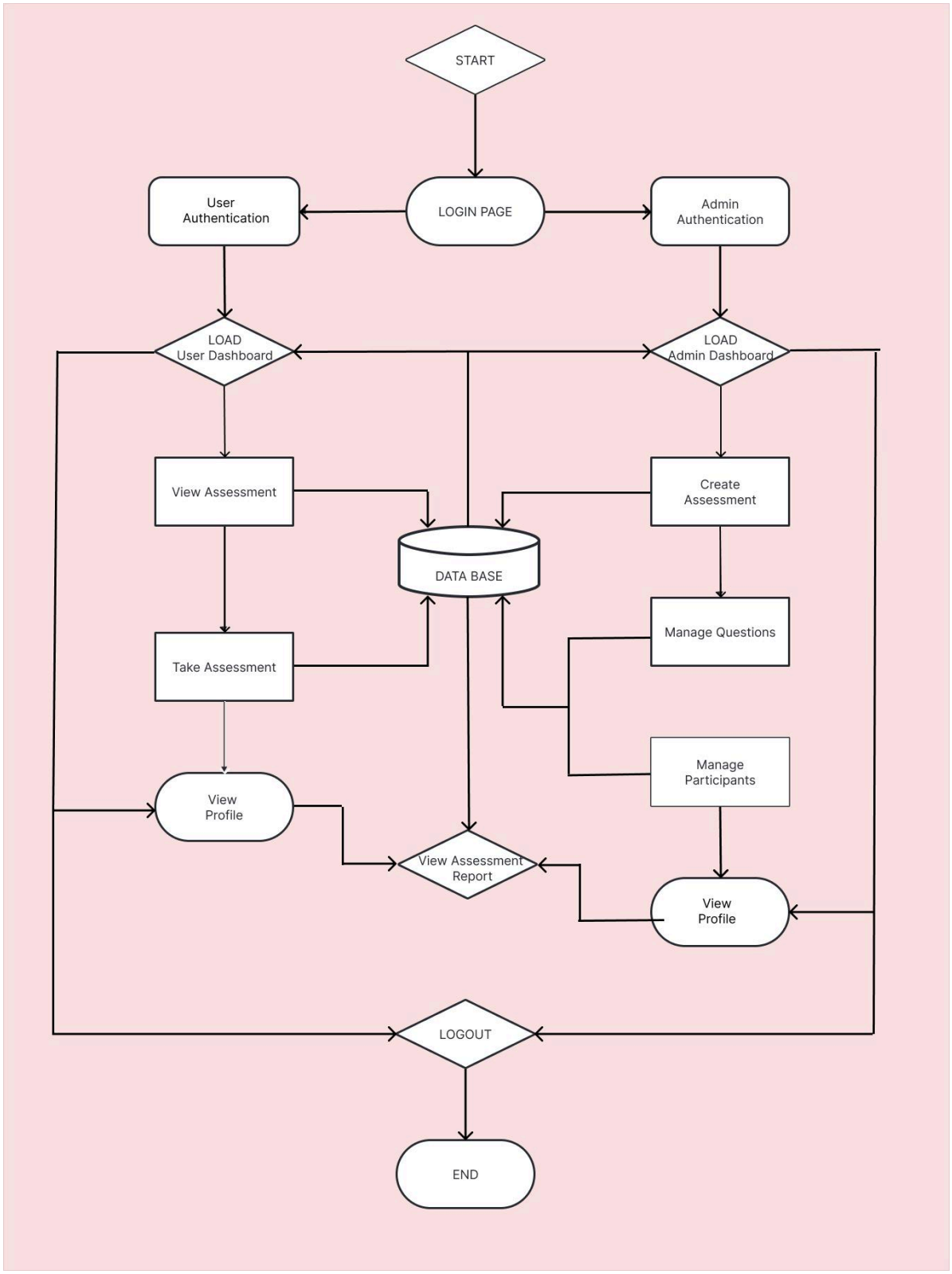
4.Security:

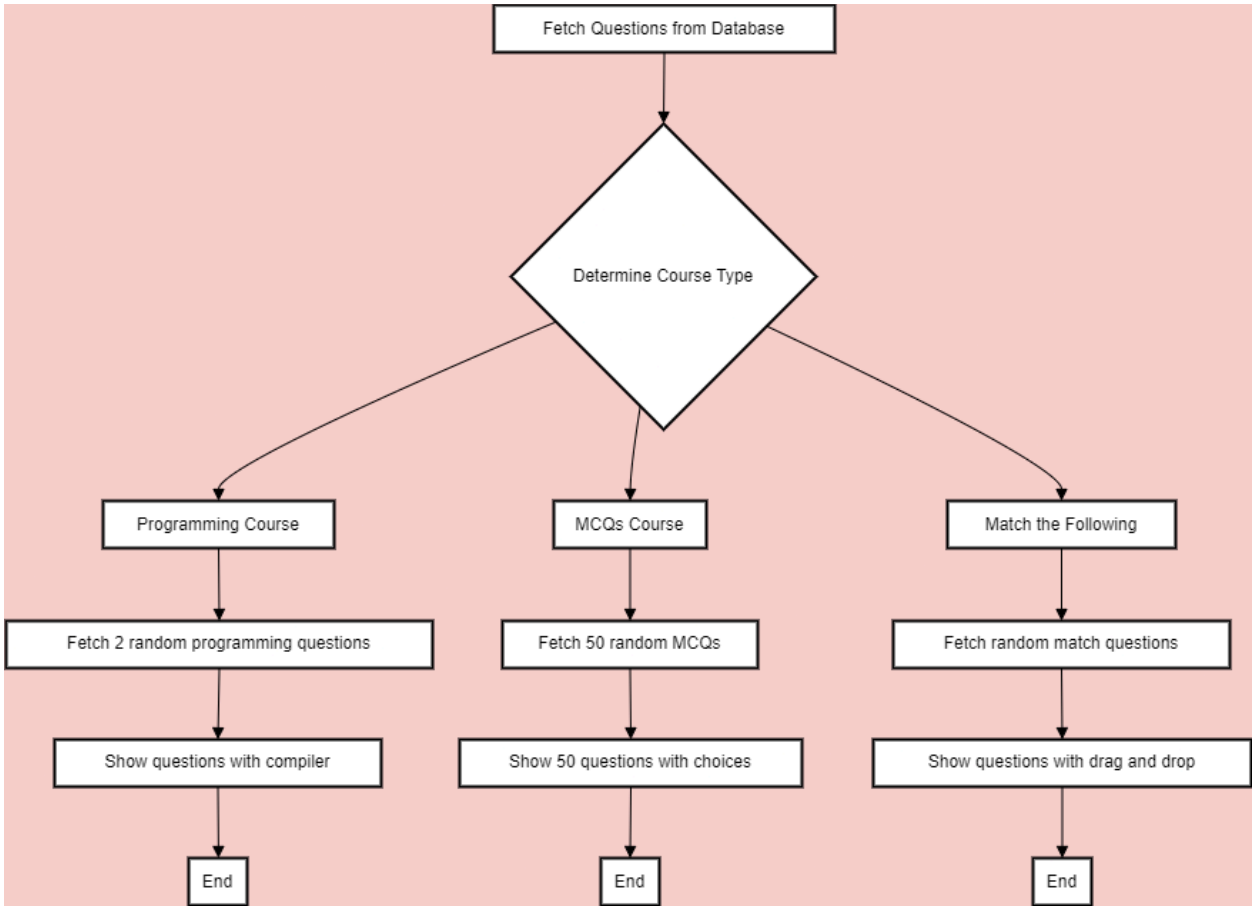
- User data should be securely stored and transmitted using encryption techniques.
- The website should implement strong authentication mechanisms to prevent unauthorized access.

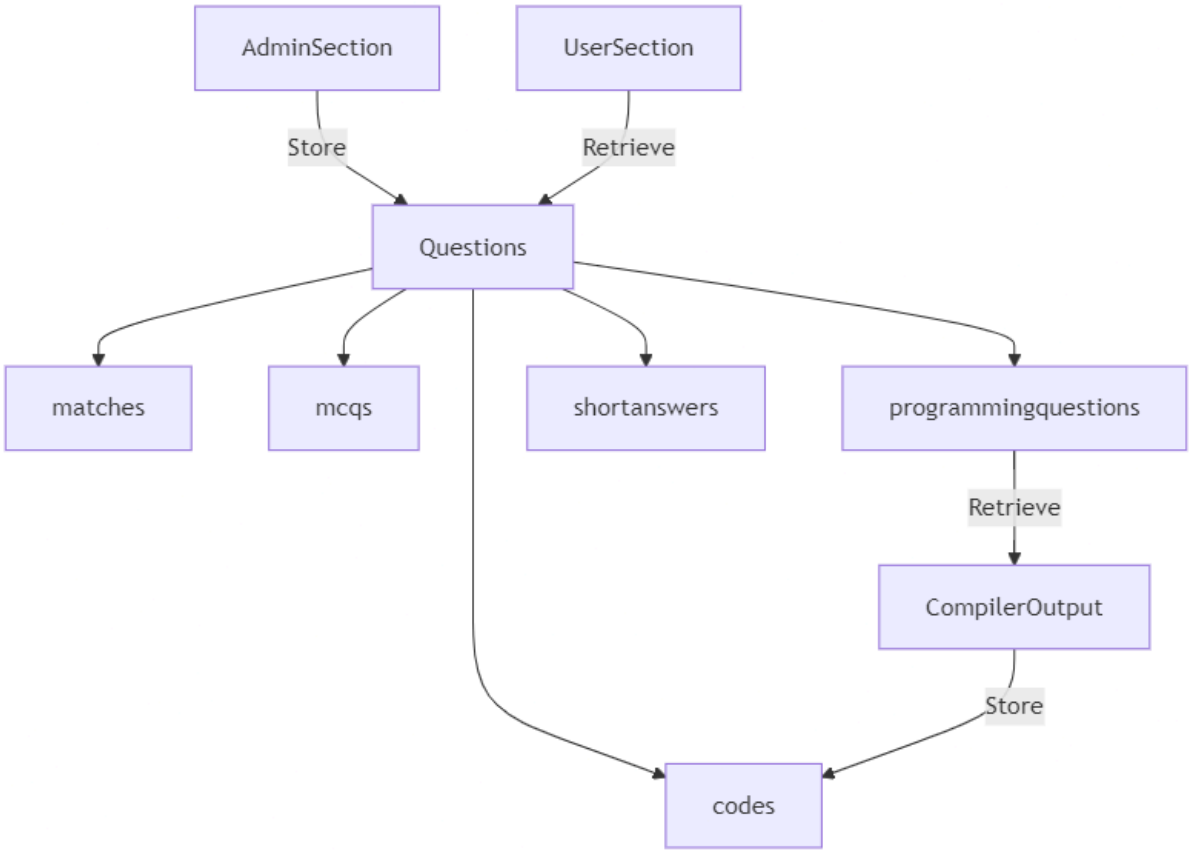
5.Scalability:

- The website should be able to scale horizontally to handle increasing load by adding more servers or resources.
- Database architecture should be designed to handle a large volume of data and concurrent transactions.

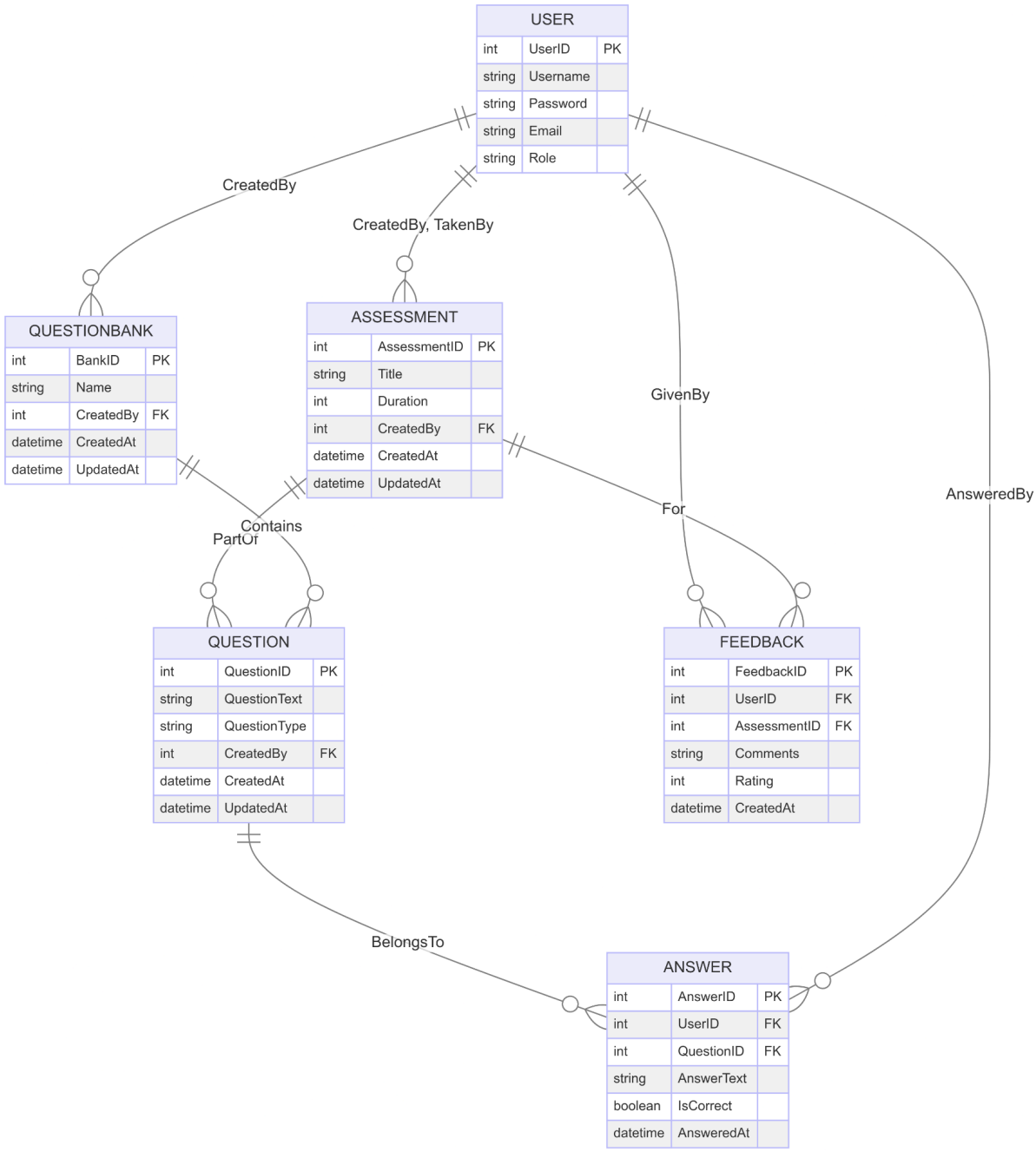
# 5. FLOWCHART :


START
LOGIN PAGE
User Authentication
Admin Authentication
LOAD User Dashboard
LOAD Admin Dashboard
View Assessment
Create Assessment
DATA BASE
Manage Questions
Take Assessment
Manage Participants
View Profile
View Assessment Report
View Profile
LOGOUT
END


Fetch Questions from Database
Determine Course Type
Programming Course
MCQs Course
Match the Following
Fetch 2 random programming questions
Fetch 50 random MCQs
Fetch random match questions
Show questions with compiler
Show 50 questions with choices
Show questions with drag and drop
End
End
End


AdminSection
UserSection
Store
Retrieve
Questions
matches
mcqs
shortanswers
programmingquestions
Retrieve
CompilerOutput
Store
codes

# ER DIAGRAM :


USER
int UserID PK
string Username
string Password
string Email
string Role
CreatedBy
CreatedBy, TakenBy
GivenBy
AnsweredBy
QUESTIONBANK
int BankID PK
string Name
int CreatedBy FK
datetime CreatedAt
datetime UpdatedAt
ASSESSMENT
int AssessmentID PK
string Title
int Duration
int CreatedBy FK
datetime CreatedAt
datetime UpdatedAt
Contains
PartOf
For
QUESTION
int QuestionID PK
string QuestionText
string QuestionType
int CreatedBy FK
datetime CreatedAt
datetime UpdatedAt
FEEDBACK
int FeedbackID PK
int UserID FK
int AssessmentID FK
string Comments
int Rating
datetime CreatedAt
BelongsTo
ANSWER
int AnswerID PK
int UserID FK
int QuestionID FK
string AnswerText
boolean IsCorrect
datetime AnsweredAt