Online Quiz System

📌 Project Overview

Develop a **basic online quiz system** with user authentication and quiz management.

Candidates can choose:

- **Backend**: Java (Spring Boot) / Golang / Python (Django or FastAPI)
- **Frontend**: React.js / Next.js / Flutter / Angular
- **Database**: MySQL / PostgreSQL

🛠️ Prerequisites

✅ **Knowledge of REST APIs** (CRUD operations)
✅ **Authentication (JWT/session-based login)**
✅ **SQL skills (relations, indexing, queries)**
✅ **Frontend development (component-based UI, API integration)**
✅ **GitHub/GitLab for submission**
✅ **Rate Limiting (100 requests/sec per user)**

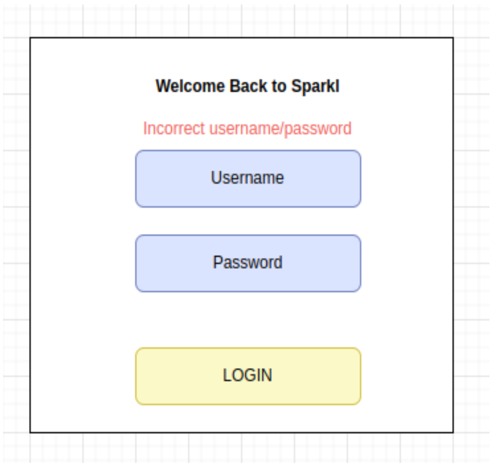
📋 Project Requirements

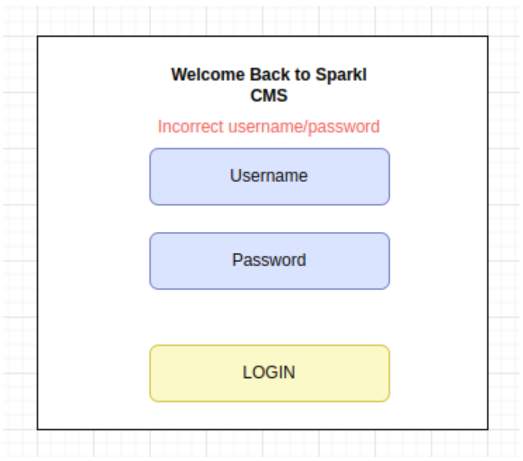
🔐 1. Security & API Access Control

- **JWT-based Authentication** for all APIs except login
- **Rate Limiting**: Each user can make **100 requests per second**
- **Session Expiry**: Token expires after **60 minutes**

🔹 2. User Authentication & Dashboard

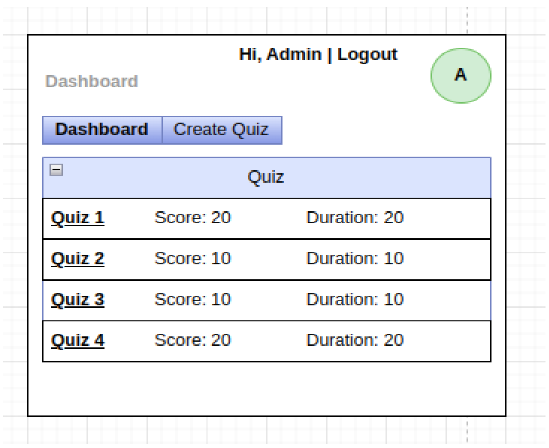
- Login pages for **Admin** and **User**
- **After login**, dashboard should display:
  ✅ **Username on top right** with **Logout** option
- **User & Admin Screen/Webpage**



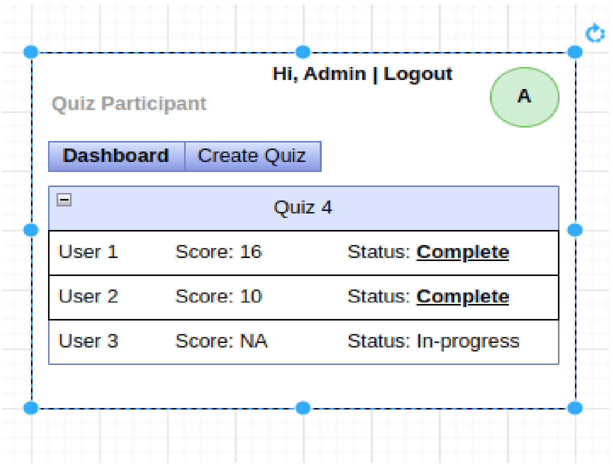


🔹 3. Quiz Management (Admin Panel)

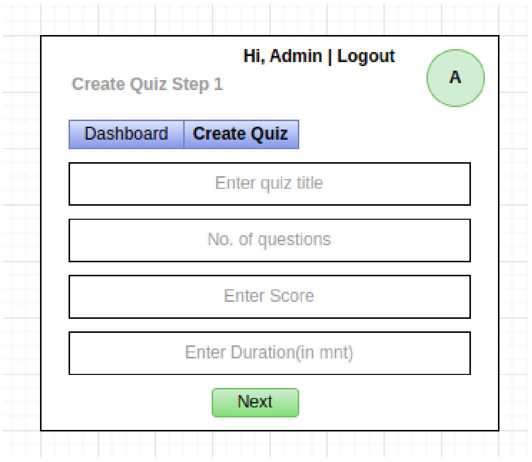
- Create quizzes (**Title, No. of Questions, Total Score, Duration**)
- Map Questions to the quiz & #**QNo.**, **marks** per question
- View Reports with:
  ✅ List of quizzes(**Title, Score, Duration**)
  ✅ List of quiz participants(**User Name, Score, Completed/In-Progress**)
  ✅ View a participant's responses(**User Score, Score, Quiz question response(correct/in-correct, marks)**)
- Screens

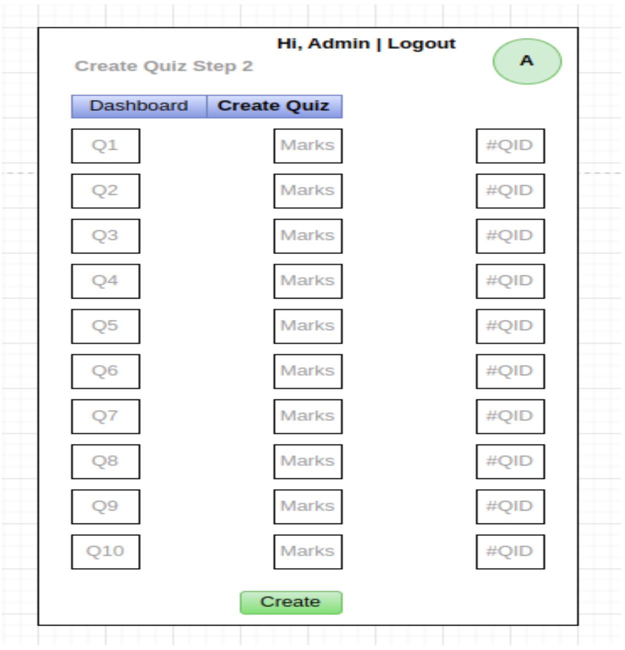


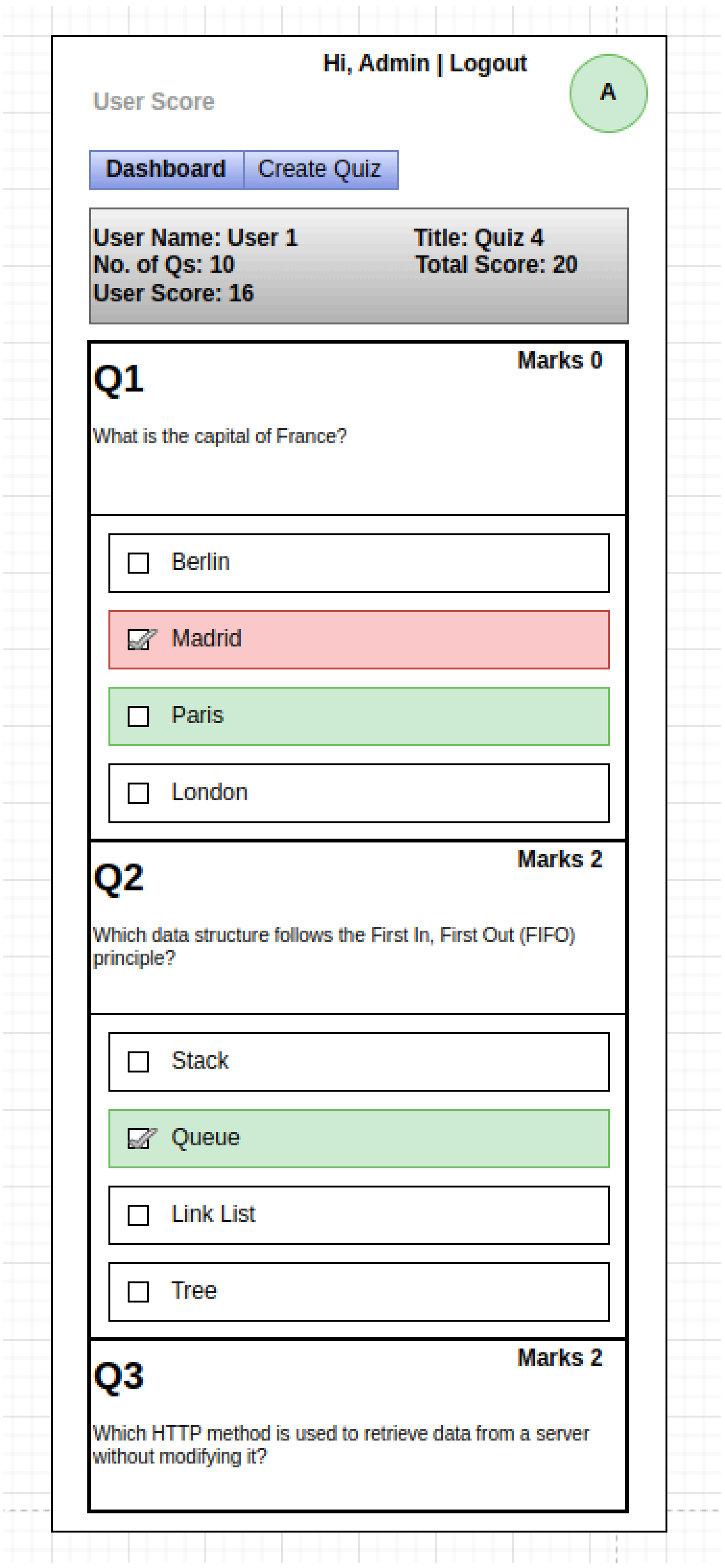
Quiz		
Quiz 1	Score: 20	Duration: 20
Quiz 2	Score: 10	Duration: 10
Quiz 3	Score: 10	Duration: 10
Quiz 4	Score: 20	Duration: 20



Quiz 4		
User 1	Score: 16	Status: **Complete**
User 2	Score: 10	Status: **Complete**
User 3	Score: NA	Status: In-progress


Hi, Admin | Logout
Create Quiz Step 1
A
Dashboard
Create Quiz
Enter quiz title
No. of questions
Enter Score
Enter Duration(in mnt)
Next


Hi, Admin | Logout
Create Quiz Step 2
A
Dashboard
Create Quiz
Q1
Marks
#QID
Q2
Marks
#QID
Q3
Marks
#QID
Q4
Marks
#QID
Q5
Marks
#QID
Q6
Marks
#QID
Q7
Marks
#QID
Q8
Marks
#QID
Q9
Marks
#QID
Q10
Marks
#QID
Create


Hi, Admin | Logout

A

User Score

Dashboard | Create Quiz

User Name: User 1
No. of Qs: 10
User Score: 16
Title: Quiz 4
Total Score: 20

Q1 **Marks 0**

What is the capital of France?

☐ Berlin
☑ Madrid
☐ Paris
☐ London

Q2 **Marks 2**

Which data structure follows the First In, First Out (FIFO) principle?

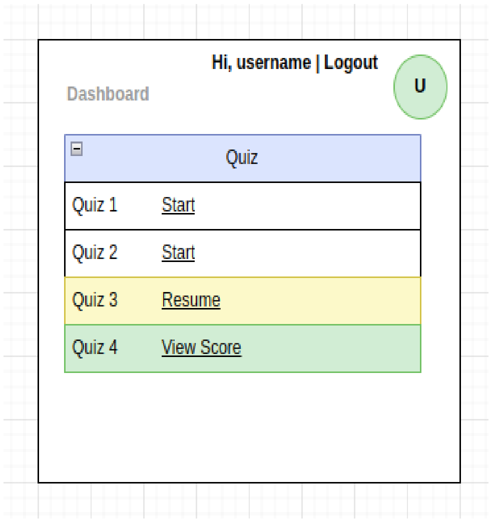
☐ Stack
☑ Queue
☐ Link List
☐ Tree

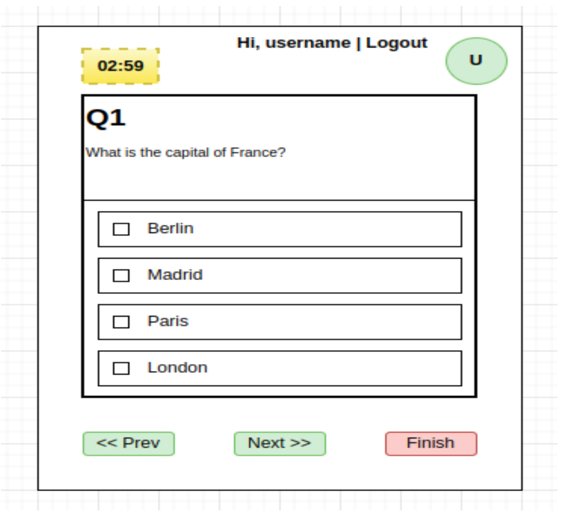
Q3 **Marks 2**

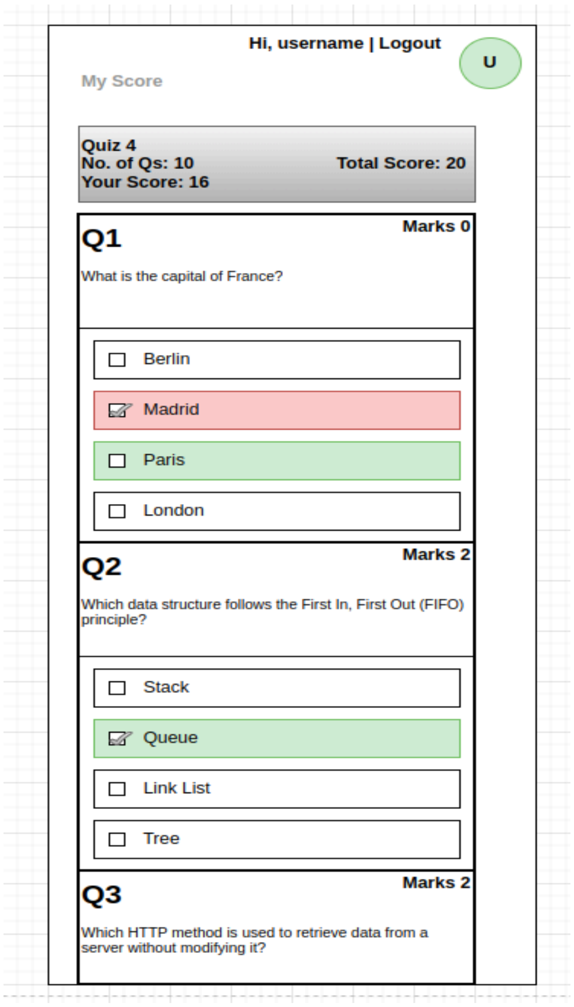
Which HTTP method is used to retrieve data from a server without modifying it?

🔹 4. Quiz Attempt (User Flow)

- Dashboard shows list of quizzes with status:(Quiz title, Start/Resume/View Score)
- View Quiz Score/Response(only completed quizzes)
  ✅ Quiz Title, Total Marks, User Marks
  ✅ All quiz questions with chosen option, correct option, marks,

- Screens/Webpage
- 




Hi, username | Logout

U

My Score

Quiz 4
No. of Qs: 10
Your Score: 16

Total Score: 20

Q1 Marks 0

What is the capital of France?

- [ ] Berlin
- [✓] Madrid
- [ ] Paris
- [ ] London

Q2 Marks 2

Which data structure follows the First In, First Out (FIFO) principle?

- [ ] Stack
- [✓] Queue
- [ ] Link List
- [ ] Tree

Q3 Marks 2

Which HTTP method is used to retrieve data from a server without modifying it?

🗂️ Database Schema

1️⃣ Questions Table

```
CREATE TABLE questions (
    id INT UNSIGNED NOT NULL AUTO_INCREMENT,
    question TEXT NOT NULL,
     PRIMARY KEY (`id`)
);
```

Sample Data

```
INSERT INTO Testing.questions (id,question) VALUES
    (1,'Which of the following is a NoSQL database?'),
    (2,'What does CSS stand for?'),
    (3,'In Java, which keyword is used to create a subclass?'),
    (4,'What does REST stand for in REST API?'),
    (5,'Which of the following sorting algorithms has the best average-case time
complexity?'),
    (6,'Which command is used to check active ports on a Linux system?'),
    (7,'In JavaScript, which of the following is used to declare a constant variable?'),
    (8,'Which HTTP status code indicates that the request was successful but no content is
returned?'),
    (9,'In SQL, which clause is used to filter results after an aggregation function is
applied?'),
    (10,'What is the primary advantage of using Docker containers?');
```

2️⃣ Question Options Table

```
CREATE TABLE question_options (
    id INT UNSIGNED NOT NULL AUTO_INCREMENT,
    question_id INT UNSIGNED NOT NULL,
    option TEXT NOT NULL,
    is_correct TINYINT UNSIGNED DEFAULT '0',
     PRIMARY KEY (`id`)
);
```

Sample Data

```
INSERT INTO question_options (id,question_id,`option`,is_correct) VALUES
    (1,1,'MySQL',0),
    (2,1,'PostgreSQL',0),
    (3,1,'MongoDB',1),
    (4,1,'SQLite',0),
    (5,2,'Computer Style Sheets',0),
    (6,2,'Creative Style Sheets',0),
    (7,2,'Cascading Style Sheets',1),
    (8,2,'Colorful Style Sheets',0),
    (9,3,'extends',1),
    (10,3,'implements',0),
    (11,3,'inherits',0),
    (12,3,'super',0),
    (13,4,'Representational State Transfer',1),
    (14,4,'Remote Execution and State Transfer',0),
    (15,4,'Rapid Enterprise System Technology',0),
    (16,4,'Recursive State Transition',0),
    (17,5,'Bubble Sort',0),
    (18,5,'Merge Sort',1),
    (19,5,'Selection Sort',0),
    (20,5,'Insertion Sort',0),
    (21,6,'netstat -tulnp',1),
    (22,6,'ps aux',0),
    (23,6,'top',0),
    (24,6,'htop',0),
    (25,7,'var',0),
    (26,7,'let',0),
    (27,7,'const',1),
    (28,7,'final',0),
    (29,8,'200',0),
    (30,8,'201',0),
    (31,8,'204',1),
    (32,8,'404',0),
    (33,9,'WHERE',0),
    (34,9,'HAVING',1),
    (35,9,'GROUP BY',0),
    (36,9,'ORDER BY',0),
    (37,10,'They allow running multiple OS instances on a single machine',0),
    (38,10,'They improve CPU performance',0),
    (39,10,'They provide consistent environments across different systems',1),
    (40,10,'They increase RAM efficiency',0);
```

2️⃣ Other Required table

Design and create tables required to manage users and admin(cms), manage token & it's expiry(60 minutes), api rate limit(100 per/sec), manage quizzes and question in questions, attempted quizzes with status and each quiz question response.

🚀 Expected API Endpoints

🛠️ User Authentication

- `POST /login` → Authenticate user & return JWT token

Rate Limiting Applied: ✅

🛠️ Quiz Management (Admin)

- `GET /quizzes` → Get quiz list
- `POST /quizzes` → Create a quiz
- `POST /quizzes/{id}/questions` → Map questions to quiz
- `GET /quizzes/{id}/participants` → Get all quiz participants list with status
- `GET /quizzes/{id}/response/{user-id}` → Get quiz response of a particular user
- More if required

Rate Limiting Applied: ✅

🛠️ Quiz Attempt (User)

- `GET /my-quizzes` → List quizzes with login user quiz status

- `POST /quizzes/{id}/start` → Start a quiz
- `POST /quizzes/{id}/submit` → Submit quiz responses
- `GET /quizzes/{id}/response` → Get quiz response of a login user
- More if required

Rate Limiting Applied: ✅

🛡️ Rate Limiting Details

🔹 **Each user can make a maximum of 100 requests per second.**
🔹 If exceeded, return **429 Too Many Requests** with a retry header.

📌 Implementation Suggestions

- **Spring Boot**: Use `Bucket4j` or `Spring RateLimiter`
- **Django**: Use `django-ratelimit` middleware
- **FastAPI**: Use `slowapi` middleware
- **Golang**: Use `golang.org/x/time/rate` package

📌 Evaluation Criteria

✅ **Code Quality** (clean, modular, maintainable)
✅ **Database Design** (relations, indexing, queries, data-type)
✅ **Security Practices** (password hashing, JWT authentication, rate limiting)
✅ **GitHub Submission** (meaningful commits, README file)

📢 Submission Guidelines

1️⃣Share your project to a **GitHub repository** (**public access**)
2️⃣Include a **README.md** with setup instructions
4️⃣**Deadline**: **18th Feb 2025**