**COLLEGE CODE: 9214**

**COLLEGE NAME: RVS SCHOOL OF ENGINEERING AND TECHNOLOGY**

**DEPARTMENT: COMPUTER SCIENCE AND ENGINEERING**

**STUDENT NM ID:**
**A73550EC6665E72857240A81EFD7EE84**

**ROLL NO: 921423104037**

**DATE : 01-10-2025**

**COMPLETED PROJECT NAMED AS**

**Phase 5**

**NAME: ONLINE QUIZ APPLICATION**

**SUBMITTED BY**

NAME: PRABAKARAN M

**MOBILE NO : 9884621302**

# Phase 5: Project Demonstration & Documentation

---

**1) Final Demo Walkthrough**

The Online Quiz Application is a full-stack web-based platform built using the MERN stack (MongoDB, Express.js, React.js, Node.js).
It allows users to register, log in, take quizzes, and instantly view their results through a clean and responsive interface.

---

Step-by-Step Flow:

1. The user opens the Online Quiz Application in their browser.
2. On the Home Page, they can navigate to Register or Login.
3. The user registers by entering their name, email, and password.
4. After successful registration or login, the user is redirected to the Dashboard.
5. The dashboard displays a list of available quizzes, each showing a title and description.
6. The user selects a quiz and clicks the Start Quiz button.
7. The quiz begins, showing one question at a time with multiple-choice options.
8. The user selects answers and navigates between questions using Next and Previous buttons.
9. After answering all questions, the user clicks the Submit button.
10. The application processes the responses and displays the final score and percentage.
11. The result page provides an option to retake the quiz or return to the dashboard.
12. The admin can add new quizzes or questions through backend APIs.
13. All user data and quiz results are stored securely in MongoDB Atlas.

---

Key Features:

- User registration and authentication with JWT security.
- Dynamic quiz questions retrieved from the MongoDB database.
- Real-time score calculation and result display.
- Responsive UI built with React.js and Tailwind CSS.
- Backend powered by Node.js and Express.js for API management.

**2) Objective**

To design and develop a **dynamic and user-friendly Online Quiz Application** using the **MERN stack (MongoDB, Express.js, React.js, and Node.js)** that enables users to take quizzes, view instant results, and manage quiz data efficiently.
The project aims to create an interactive learning platform that enhances user engagement and provides real-time evaluation through seamless frontend and backend integration.

---

**3) Technologies Used**

- **HTML** – Used for structuring the quiz interface and forms
- **CSS (Tailwind CSS)** – Provides responsive and modern styling
- **JavaScript (React.js)** – Handles dynamic rendering and user interaction
- **Node.js & Express.js** – Backend server for API handling and quiz management
- **MongoDB Atlas** – Cloud database for storing users, quizzes, and scores
- **JWT (JSON Web Token)** – Ensures secure authentication

---

**4) System Overview**

The system follows a **client–server architecture**. The **frontend (React.js)** interacts with the **backend (Node.js + Express)** through API requests, while all data is stored and managed in **MongoDB Atlas**.

**Architecture**

[User Interface - React.js + Tailwind CSS]
↓
[API Logic - Node.js + Express.js]
↓
[Database - MongoDB Atlas]

---

## 5) Advantages

- Fast, scalable, and secure MERN-based system
- User authentication ensures personalized access
- Real-time score calculation and result display
- Data stored safely in cloud database (MongoDB Atlas)
- Responsive design compatible with all devices
- Easy to extend with more quizzes or features

---

## 6) Real-World Applications

- Online education and e-learning platforms
- College and school quiz systems
- Employee training assessments
- Skill evaluation and certification tests
- Interactive learning modules

---

## 7) Source Code

### Frontend – React.js

### App.jsx

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import Home from './pages/Home';
import Login from './pages/Login';
import Register from './pages/Register';
import Dashboard from './pages/Dashboard';
import Quiz from './pages/Quiz';
import Navbar from './components/Navbar';

function App() {
  return (
    <Router>
      <Navbar />
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/login" element={<Login />} />
        <Route path="/register" element={<Register />} />
        <Route path="/dashboard" element={<Dashboard />} />
        <Route path="/quiz/:id" element={<Quiz />} />
      </Routes>
    </Router>
  );
}
export default App;
```

## Navbar.jsx

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import Home from './pages/Home';
import Login from './pages/Login';
import Register from './pages/Register';
import Dashboard from './pages/Dashboard';
import Quiz from './pages/Quiz';
import Navbar from './components/Navbar';

function App() {
  return (
    <Router>
      <Navbar />
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/login" element={<Login />} />
        <Route path="/register" element={<Register />} />
        <Route path="/dashboard" element={<Dashboard />} />
        <Route path="/quiz/:id" element={<Quiz />} />
      </Routes>
    </Router>
  );
}
export default App;
```

## Dashboard.jsx

```
import { useEffect, useState } from 'react';
import axios from 'axios';
import QuizCard from '../components/QuizCard';

const Dashboard = () => {
  const [quizzes, setQuizzes] = useState([]);

  useEffect(() => {
    axios.get('http://localhost:5000/api/quizzes')
      .then(res => setQuizzes(res.data))
      .catch(err => console.error(err));
  }, []);

  return (
    <div className="p-6">
      <h2 className="text-2xl font-bold mb-4">Available Quizzes</h2>
      {quizzes.length === 0 ? (
        <p>No quizzes available right now.</p>
      ) : (
        <div className="grid grid-cols-3 gap-4">
```

```
        {quizzes.map(q => <QuizCard key={q._id} quiz={q} />)}
       </div>
     )}
    </div>
  );
};
export default Dashboard;
```

---

## Backend – Node.js & Express.js

### index.js

```
require('dotenv').config();
const express = require('express');
const cors = require('cors');
const connectDB = require('./config/db');

const app = express();
app.use(express.json());
app.use(cors());

// Connect MongoDB
connectDB();

// Routes
app.get('/', (req, res) => res.send('API running...'));
app.use('/api/auth', require('./routes/auth'));
app.use('/api/quizzes', require('./routes/quizzes'));

const PORT = process.env.PORT || 5000;
app.listen(PORT, () => console.log(`🚀 Server started on port ${PORT}`));
```

### db.js

```
const mongoose = require('mongoose');
const connectDB = async () => {
  try {
    await mongoose.connect(process.env.MONGO_URI);
    console.log('✅ MongoDB Connected');
  } catch (err) {
    console.error('❌ MongoDB connection failed:', err.message);
    process.exit(1);
  }
};
module.exports = connectDB;
```

### Quiz.js

```
const mongoose = require('mongoose');

const quizSchema = new mongoose.Schema({
  title: { type: String, required: true },
  description: String,
  category: String,
  questions: [
```

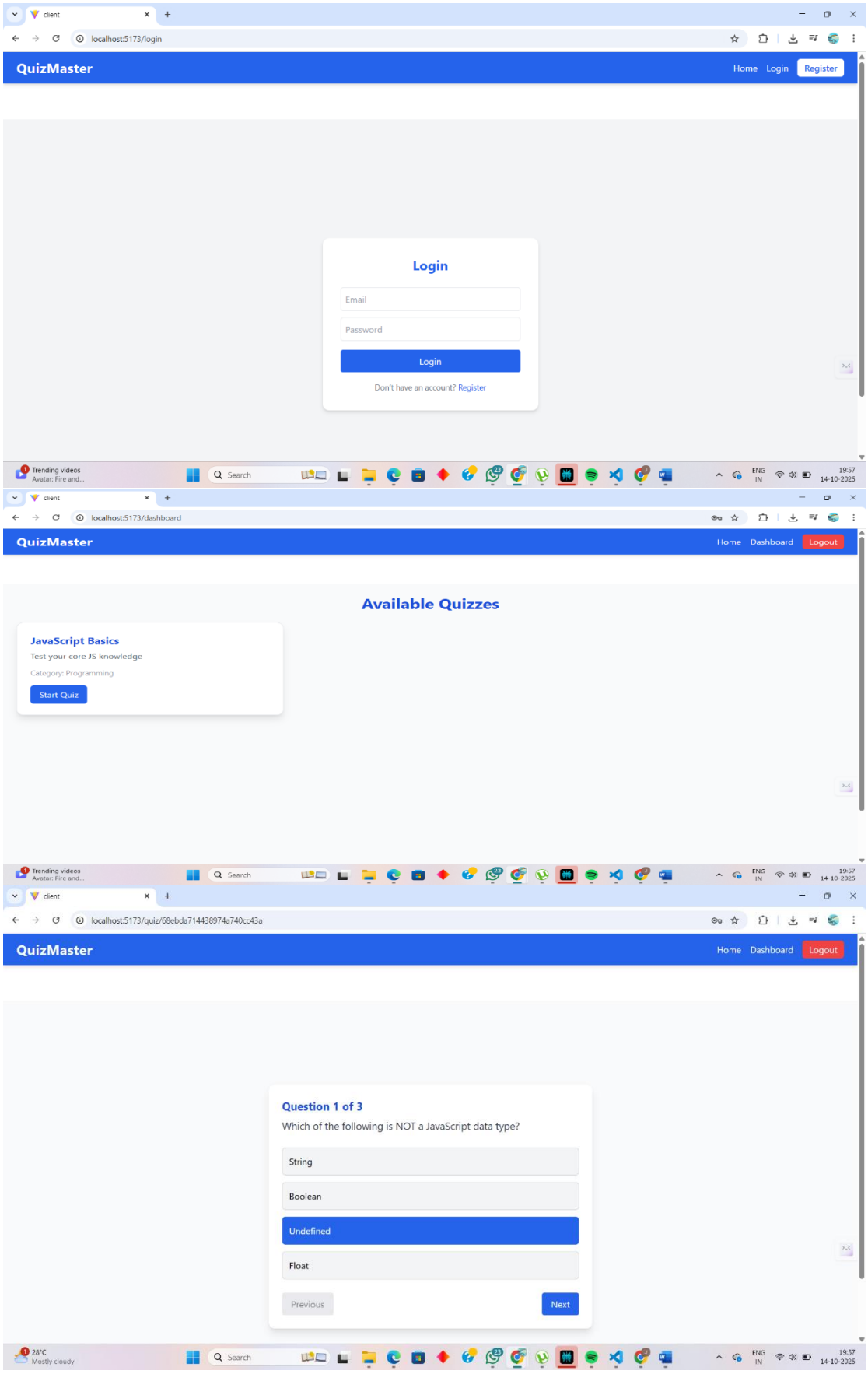
```
    {
      text: String,
      options: [String],
      correctIndex: Number
    }
  ]
});

module.exports = mongoose.model('Quiz', quizSchema);
```

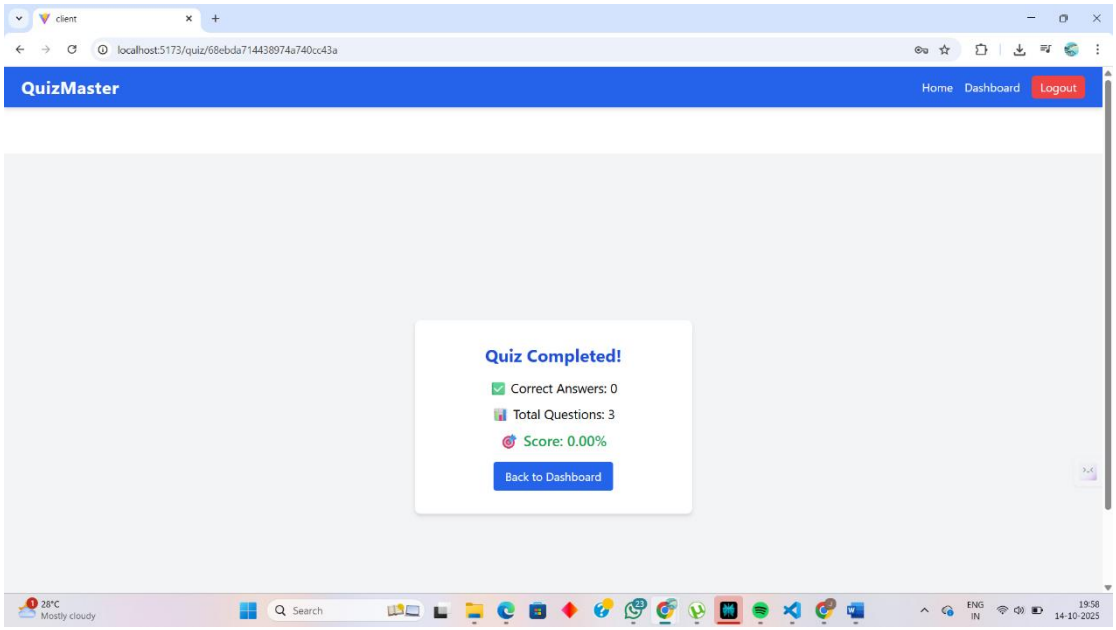
### quizzes.js

```
const express = require('express');
const Quiz = require('../models/Quiz');
const router = express.Router();

// Get all quizzes
router.get('/', async (req, res) => {
  try {
    const quizzes = await Quiz.find();
    res.status(200).json(quizzes);
  } catch {
    res.status(500).json({ msg: 'Error fetching quizzes' });
  }
});

// Create quiz
router.post('/', async (req, res) => {
  try {
    const quiz = await Quiz.create(req.body);
    res.status(201).json(quiz);
  } catch {
    res.status(500).json({ msg: 'Error creating quiz' });
  }
});

module.exports = router;
```

---

## 8) Screenshots:


client
localhost:5173/login
QuizMaster
Home
Login
Register
Login
Email
Password
Login
Don't have an account? Register
client
localhost:5173/dashboard
QuizMaster
Home
Dashboard
Logout
Available Quizzes
JavaScript Basics
Test your core JS knowledge
Category: Programming
Start Quiz
client
localhost:5173/quiz/68ebda714438974a740cc43a
QuizMaster
Home
Dashboard
Logout
Question 1 of 3
Which of the following is NOT a JavaScript data type?
String
Boolean
Undefined
Float
Previous
Next




## 9) Github Repo Link:

https://github.com/Jeziel-ent/onlinequiz.git

## 10) Conclusion:

The Online Quiz Application project successfully demonstrates the development of a fully functional web-based platform using the MERN stack.
It provides users with an interactive and efficient way to take quizzes, view instant results, and improve their knowledge.
The integration of React.js, Node.js, Express.js, and MongoDB ensures scalability, security, and performance.
Through this project, modern web technologies were effectively utilized to create a user-friendly application suitable for educational and training environments.