

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

**CHENNAI-602105**

**Online Quiz System with Leaderboard**

**A CAPSTONE PROJECT REPORT**

*Submitted in the partial fulfillment for the completion of the course*

**CSA4307 INTERNET PROGRAMMING FOR CLIENT SERVER MODEL**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**Submitted by**

**Krishna Charan. K**,

**Under the Supervision of**

**L.REETHA**

**MAY 2025**

**DECLARATION**

I, **Krishna Charan. K**, students of **Bachelor of Engineering in the Department** of Computer Science and Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha School of Engineering, Chennai, hereby declare that the work presented in this Capstone Project Work entitled **Online Quiz System with Leaderboard** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics.

**Krishna Charan. K**

(192311316)

Date:

Place:

**CERTIFICATE**

This is to certify that the project entitled **“Online Quiz System with Leaderboard”** submitted by **Jaivignesh.S** has been carried out under my supervision. The project has been submitted as per the requirements in the current semester of B.E. Computer Science and Engineering.

Supervisor

L.Reetha

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TOPICS** | **PAGE NO.** |
| 1 | **Abstract** | **1** |
| 2 | **Introduction** | **2** |
| 3 | **Project Description** | **3** |
| 4 | **Problem Description** | **5** |
| 5 | **Tool Description** | **6** |
| 6 | **Block Diagram** | **9** |
| 7 | **Operations**  Store User Information  Store Quiz Questions and Answers  Store Quiz Results and Leaderboard Data. | **10** |
| 7 | **Approach / Module Description / Functionalities**  The idea is to form an individual functions for every operation. All the functions are unified together to form software. | **12** |
| 8 | **Implementation**  Coding | **14** |
| 9 | **Result**  Output with Screenshots | **22** |
| 10 | **Conclusion**  Future Enhancement  **References** | **25** |

**ABSTRACT**

This capstone project presents the development of an Online Quiz System with Leaderboard, designed to offer an interactive platform for assessing users’ knowledge while fostering healthy competition through real-time score tracking. The system allows users to attempt quizzes on various topics, receive immediate feedback with explanations for incorrect answers, and view their scores on a dynamic leaderboard. It is aimed at enhancing learning outcomes in educational and training environments by combining assessment with gamification elements.

The application includes core features such as user registration, timed quizzes, automated grading, and persistent storage of scores. The leaderboard encourages repeated participation by showcasing top performers, thus motivating learners to improve. Explanations provided after each incorrect answer support immediate learning reinforcement. Built with web technologies like HTML, CSS, and JavaScript, and using local storage for maintaining leaderboard data, the system provides a fast and responsive user experience without requiring a backend connection for basic functionality.

This project highlights the practical implementation of front-end web development concepts and emphasizes the importance of user engagement in digital learning tools. It serves as a foundation for future enhancements such as question randomization, backend integration for user management, and support for multiple quiz categories.

**INTRODUCTION**

The rapid evolution of digital technology and increased access to the internet have significantly transformed the landscape of education and skill assessment. Online learning platforms and assessment tools have become essential for both academic institutions and organizations to evaluate knowledge, provide feedback, and engage learners more effectively. This capstone project aims to develop a comprehensive Online Quiz System with Leaderboard, leveraging modern web technologies to create an interactive and motivating environment for learning and assessment.

The primary objective of this project is to provide users with a seamless and user-friendly platform to take quizzes on various subjects, receive immediate feedback, and compare their performance with others through a real-time leaderboard. By integrating features such as automatic grading, explanation of incorrect answers, and persistent score tracking, the system enhances user engagement and supports active learning.

Key features include a responsive quiz interface, instant result evaluation, detailed feedback with explanations, and a dynamic leaderboard that promotes healthy competition among users. The system ensures efficient data handling and offers a scalable foundation for future upgrades, such as category-wise quizzes, time-based challenges, user authentication, and cloud-based data storage. This project not only addresses common challenges in online assessment systems but also demonstrates the practical application of front-end web development skills in building meaningful educational tools.

**PROJECT DESCRIPTION**

This project involves the design and development of a web-based Online Quiz System with Leaderboard, built using front-end web technologies such as HTML, CSS, and JavaScript, with optional support for backend technologies like Node.js and MongoDB for enhanced functionality. The system provides an engaging and user-friendly platform for conducting online quizzes, automatically evaluating responses, offering instant feedback with explanations, and maintaining a dynamic leaderboard to foster healthy competition among participants.

**Key Features and Functionalities:**

**1. User Identification and Quiz Participation:**

Users can enter their names to participate in quizzes. The system captures and tracks individual performance without requiring full account registration, making it quick and accessible for users to begin a quiz session.

**2. Automated Scoring and Answer Explanations:**

After submitting the quiz, users receive an immediate evaluation of their responses. Explanations are shown for incorrect answers to promote better understanding and reinforce learning in real time.

**3. Dynamic Leaderboard:**

A built-in leaderboard displays the top performers based on quiz scores. The leaderboard updates automatically and stores results using local or cloud-based storage, encouraging repeated participation and gamifying the learning experience.

**4. Scalable Quiz Interface:**

The system is designed to support multiple questions with flexible structures, including single or multiple-choice formats. The modular approach allows easy addition of new quizzes or subjects for broader usage.

**5. Leaderboard Management and Refresh:**

Administrators or users have the option to clear or reset the leaderboard, ensuring the system can be reused for new groups or quiz rounds. This helps maintain relevance and clarity in score tracking.

**PROBLEM DESCRIPTION**

The primary goal of this project is to develop a functional and interactive software application that facilitates online quiz participation while promoting engagement through a real-time leaderboard. The system addresses various challenges associated with traditional quiz and assessment methods, such as limited accessibility, lack of immediate feedback, and minimal user motivation due to absence of competitive features.

In today’s digital learning environment, there is a growing demand for platforms that can deliver engaging and effective knowledge assessments. However, many existing systems fail to provide immediate feedback or encourage continuous improvement. Moreover, learners often lack the motivation to participate actively in quizzes that do not track performance or recognize achievement. This project aims to overcome these limitations by developing an Online Quiz System with Leaderboard, offering an accessible, educational, and competitive experience using modern web technologies.

**Project Objectives:**

1. **Enhance User Engagement:**

Design a responsive and intuitive quiz interface that allows users to answer questions easily and view results instantly, encouraging more active participation.

1. **Immediate Feedback for Learning:**

Provide real-time evaluation of answers, along with detailed explanations for incorrect responses, to reinforce learning and promote better understanding of concepts.

1. **Encourage Healthy Competition:**

Implement a dynamic leaderboard that showcases top scores, motivating users to improve and track their progress over time.

1. **Enable Scalable and Reusable Assessments:**

Develop a flexible system that can accommodate multiple quizzes or subjects and be reused across different user groups, classrooms, or training sessions.

**TOOLS DESCRIPTION**

The Online Quiz System with Leaderboard utilizes standard web technologies—HTML, CSS, and JavaScript—to create an interactive, responsive, and lightweight application. The system may optionally incorporate backend support using Node.js and MongoDB to enable persistent data storage and scalability. Each technology contributes to the functionality, usability, and performance of the platform, ensuring a seamless experience for learners and quiz administrators.

**HTML/CSS:** Structure and styling of the web pages, ensuring responsiveness across devices.

**JavaScript:** Handles quiz logic, answer validation, leaderboard functionality, and real-time user feedback.

**Node.js (optional):** Manages server-side operations, such as storing quiz data and managing user sessions.

**MongoDB (optional):** Stores user scores, quiz content, and leaderboard data in a scalable and efficient format..

**User Interface:**

The user interface (UI) is designed to be clean, intuitive, and responsive, suitable for users of all age groups. It is structured into two main sections: Participant Interface and Admin Interface (if backend is implemented)

**1. Participant Interface:**

* **Home Page:**

Welcomes users and prompts them to enter their name before starting the quiz. Displays quiz instructions and time limits if applicable.

* **Quiz Page:**

Presents 15 multiple-choice questions one at a time or as a full list. Users can select answers and submit the quiz when ready.

* **Results Page:**

Displays the total score, correct answers, and detailed explanations for each incorrect response. This supports immediate learning and feedback.

* **Leaderboard:**

Shows a dynamically updated list of top scorers. Users can view their rank in real-time, encouraging competition and reattempts.

**2. Admin Interface (Optional for Advanced Version):**

* **Quiz Management:**

Enables creation, editing, and deletion of quiz questions and answers.

* **Leaderboard Control:**

Allows administrators to reset or clear the leaderboard for new quiz rounds or participants.

**BLOCK DIAGRAM**

Start

Online Quiz System With Leaderboard

Login

Show Quiz

Attempt Quiz

View Results

Leaderboard

Logout

End

**OPERATIONS**

In the Online Quiz System with Leaderboard project, several key operations are implemented to manage users, quiz data, responses, and leaderboard scores. Below are the main operations and the corresponding data storage requirements necessary for efficient system functionality and user engagement:

**1. Store User Information:**

* **User Name:**

Stores the participant's name to personalize the quiz experience and identify scores on the leaderboard.

* **Unique User ID (optional):**

In advanced implementations with backend support, each user may be assigned a unique identifier for tracking performance over multiple sessions or quizzes.

* **Score History :**

If persistent storage is used, the system can keep a record of users’ past scores for analytics or performance review.

**2. Store Quiz Questions and Answers:**

* **Question ID:**

Each question is assigned a unique identifier for management and tracking.

* **Question Text:**

Stores the actual quiz question presented to the user.

* **Answer Options:**

An array of possible choices (e.g., A, B, C, D) for each question.

* **Correct Answer:**

Used to validate user responses and calculate the final score.

* **Explanation:**

A brief description explaining the correct answer, shown when the user selects an incorrect option to support immediate learning.

**3. Store Quiz Results and Leaderboard Data:**

* **User Name / ID:**

Links each score entry to a user.

* **Score:**

The number of correct answers submitted by the user.

* **Timestamp:**

Stores the date and time when the quiz was completed, which can be used to rank scores or show recent activity.

* **Leaderboard Entries:**

A collection of high scores sorted in descending order. This data is used to populate the leaderboard interface dynamically.

**MODULE DESCRIPTION**

The Online Quiz System with Leaderboard is developed using a modular architecture where each key functionality is encapsulated within an independent module. This design approach promotes maintainability, scalability, and ease of integration for future upgrades or enhancements. Each module is responsible for a specific aspect of the system and works in unison to deliver a seamless and efficient quiz-taking experience. Below is a breakdown of the main modules and their functionalities:

**1. User Interaction Module:**

**Functionality:** Handles the initial interaction personalized including name input, quiz initiation, and basic session tracking.

**Features:**

* Input field for users to enter their name before starting the quiz.
* Instructions or guidelines about the quiz format and scoring.
* Optional timer setup to add time-based challenge.
* Ensures a smooth entry point for users and stores session data for personalized experience.

**2. Quiz Management Module:**

**Functionality:** Controls the delivery of quiz content and user interaction with questions.

**Features:**

* Dynamic loading of 15 multiple-choice questions from a predefined database or JSON file.
* Real-time user answer tracking and navigation (next/previous or single-page mode).
* Submission validation and prevention of multiple submissions.
* Designed to provide a responsive and intuitive quiz-taking interface.

**3. Evaluation and Feedback Module:**

**Functionality:** Manages the answer evaluation process and generates feedback.

**Features:**

* Instant checking of user responses against the correct answers.
* Displays the final score upon submission.
* Provides detailed explanations for each incorrect answer to support learning.
* Enhances the educational value of the quiz by promoting understanding, not just grading.

**4. Leaderboard Module:**

**Functionality:** Maintains a real-time leaderboard displaying top scores.

**Features:**

* Stores each user’s score along with their name and timestamp.
* Ranks participants in descending order based on score.
* Provides a "Refresh" feature to update the leaderboard dynamically.
* Optional feature to clear/reset the leaderboard for new quiz rounds.
* Encourages user engagement through competition and recognition.

**IMPLEMENTATION**

**HTML Code:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Online Quiz</title>

<style>

body {

margin: 0;

font-family: Arial, sans-serif;

background: url('Quiz1.jpeg') no-repeat center center fixed;

background-size: cover;

}

#login-box {

width: 400px;

margin: 100px auto;

background: rgba(255, 255, 255, 0.95);

padding: 30px;

border-radius: 12px;

box-shadow: 0 0 10px rgba(0,0,0,0.3);

text-align: center;

}

#login-box h1 {

margin-bottom: 10px;

}

#login-box p {

margin-bottom: 20px;

color: #333;

}

input[type="text"], input[type="email"], input[type="password"] {

padding: 10px;

width: 90%;

margin: 8px 0;

border: 1px solid #ccc;

border-radius: 5px;

}

select {

padding: 10px;

width: 95%;

margin-top: 10px;

border: 1px solid #ccc;

border-radius: 5px;

}

button {

padding: 10px 20px;

margin-top: 15px;

cursor: pointer;

background-color: #28a745;

color: white;

border: none;

border-radius: 5px;

}

#quiz-box, #leaderboard {

display: none;

background: #fff;

margin: 20px auto;

padding: 20px;

width: 80%;

border-radius: 10px;

box-shadow: 0 2px 8px rgba(0,0,0,0.1);

}

.question {

margin-bottom: 20px;

}

.explanation {

display: none;

background: #f8d7da;

color: #721c24;

padding: 10px;

border-left: 5px solid #f5c6cb;

border-radius: 5px;

margin-top: 5px;

}

#leaderboard-list li {

background: #eee;

padding: 8px;

margin: 5px 0;

border-radius: 5px;

}

#logout-btn, #okay-btn {

margin-top: 20px;

padding: 10px 20px;

cursor: pointer;

background-color: #007bff;

color: white;

border: none;

border-radius: 5px;

}

#logout-btn {

background-color: #dc3545;

display: none;

}

</style>

</head>

<body>

<div id="login-box">

<h1>Online Quiz</h1>

<p>Test your knowledge in different levels: Easy, Medium, or Hard!</p>

<input type="text" id="name" placeholder="Your Name">

<input type="email" id="email" placeholder="Your Email">

<input type="password" id="password" placeholder="Your Password">

<select id="category">

<option value="">Select Category</option>

<option value="easy">Easy</option>

<option value="medium">Medium</option>

<option value="hard">Hard</option>

</select>

<button onclick="loginUser()">Start Quiz</button>

</div>

<div id="quiz-box">

<h2>Quiz</h2>

<div id="quiz-content"></div>

<button onclick="submitQuiz()">Submit Quiz</button>

</div>

<div id="leaderboard">

<h2>Leaderboard</h2>

<ul id="leaderboard-list"></ul>

<button id="okay-btn" onclick="showLogout()">Okay</button>

<button id="logout-btn" onclick="logout()">Logout</button>

</div>

<script>

const allQuestions = {

easy: [

{

question: "What does HTML stand for?",

options: ["Hyper Trainer Marking Language", "Hyper Text Marketing Language", "Hyper Text Markup Language", "Hyper Tool Multi Language"],

answer: 2,

explanation: "HTML stands for Hyper Text Markup Language, the standard for web pages."

},

{

question: "Which protocol is used to transfer web pages on the Internet?",

options: ["FTP", "SMTP", "HTTP", "SNMP"],

answer: 2,

explanation: "HTTP is used to transfer web pages from server to browser."

},

{

question: "What does CSS stand for?",

options: ["Computer Style Sheets", "Creative Style Syntax", "Cascading Style Sheets", "Colorful Style Sheets"],

answer: 2,

explanation: "CSS means Cascading Style Sheets for styling HTML."

},

{

question: "Which language is used for client-side scripting in web development?",

options: ["PHP", "Python", "JavaScript", "SQL"],

answer: 2,

explanation: "JavaScript is the client-side scripting language."

},

{

question: "Which of the following is a server-side scripting language?",

options: ["HTML", "CSS", "JavaScript", "PHP"],

answer: 3,

explanation: "PHP is a server-side language used for dynamic web pages."

},

{

question: "What does URL stand for?",

options: ["Uniform Resource Locator", "Uniform Referencing Link", "Universal Research Locator", "Unique Reference Link"],

answer: 0,

explanation: "URL stands for Uniform Resource Locator, an address on the web."

},

{

question: "What is the main purpose of a web browser?",

options: ["To create web pages", "To store websites", "To translate code", "To display web pages"],

answer: 3,

explanation: "A browser displays web pages from the internet."

},

{

question: "Which one of the following is not a web browser?",

options: ["Chrome", "Firefox", "Linux", "Safari"],

answer: 2,

explanation: "Linux is an operating system, not a web browser."

}

],

medium: [

{

question: "What does JavaScript primarily run in?",

options: ["Web Browser", "Database Server", "Application Server", "Cloud Server"],

answer: 0,

explanation: "JavaScript primarily runs in web browsers to create dynamic and interactive web pages."

},

{

question: "Which of the following is a valid JavaScript variable declaration?",

options: ["let x = 10;", "int x = 10;", "variable x = 10;", "const x;"],

answer: 0,

explanation: "In JavaScript, variables can be declared using let, var, or const."

}

],

hard: [

{

question: "Which of the following is not a valid HTTP request method?",

options: ["HEAD", "OPTIONS", "TRACE", "PUTGET"],

answer: 3,

explanation: "PUTGET is not a valid HTTP request method. Valid methods include GET, POST, PUT, DELETE, HEAD, OPTIONS, and TRACE."

},

{

question: "In a RESTful API, which of the following HTTP methods is used to update a resource?",

options: ["GET", "POST", "PUT", "PATCH"],

answer: 2,

explanation: "PUT is used to update or replace a resource on the server. PATCH is also used to make partial updates to a resource."

},

{

question: "Which of the following statements is true about cookies in web development?",

options: ["Cookies are stored in the server", "Cookies have an unlimited lifespan", "Cookies can only store strings", "Cookies can only be used in GET requests"],

answer: 2,

explanation: "Cookies are stored on the client-side and can only store string data. If you want to store objects, they need to be serialized into strings (usually JSON format)."

},

{

question: "Which of the following is a characteristic of WebSockets?",

options: ["They are used only for sending data from the client to the server", "WebSockets require frequent page reloads", "They allow full-duplex communication between client and server", "WebSockets use the HTTP protocol exclusively"],

answer: 2,

explanation: "WebSockets provide a full-duplex, persistent connection between the client and server, enabling real-time data exchange without needing to reload the page."

},

{

question: "Which of the following is not an element of the HTTP request message?",

options: ["Request line", "Headers", "Body", "Response code"],

answer: 3,

explanation: "The response code is part of the HTTP response message, not the request message. A request message consists of the request line, headers, and optional body."

},

{

question: "In Node.js, which function is used to start a web server that listens on a specific port?",

options: ["http.createServer()", "server.listen()", "app.listen()", "createServer()"],

answer: 0,

explanation: "http.createServer() is used to create an HTTP server in Node.js. The server.listen() method is then used to specify the port number the server should listen on."

},

{

question: "What is the role of a proxy server in client-server communication?",

options: ["To redirect client requests to a different server", "To encrypt data between the client and the server", "To manage database connections", "To store static resources for faster delivery"],

answer: 0,

explanation: "A proxy server acts as an intermediary between the client and server, typically used for load balancing, caching, or security purposes. It can also redirect client requests to different servers."

},

{

question: "What is the purpose of the Access-Control-Allow-Origin header in HTTP responses?",

options: ["To specify which domains are allowed to make requests to the server", "To define the security level of the server", "To authenticate users from different domains", "To compress the response data"],

answer: 0,

explanation: "The Access-Control-Allow-Origin header is part of the Cross-Origin Resource Sharing (CORS) policy. It controls which domains are allowed to access the resources on the server."

}

]

};

let selectedCategory = "";

function loginUser() {

const name = document.getElementById('name').value.trim();

const email = document.getElementById('email').value.trim();

const password = document.getElementById('password').value.trim();

const category = document.getElementById('category').value;

if (!name || !email || !password || !category) {

alert("Please fill in all fields and select a category.");

return;

}

selectedCategory = category;

localStorage.setItem("quizUser", JSON.stringify({ name, email }));

document.body.style.background = "#f9f9f9";

document.getElementById('login-box').style.display = 'none';

document.getElementById('quiz-box').style.display = 'block';

loadQuiz();

}

function loadQuiz() {

const quizContent = document.getElementById('quiz-content');

quizContent.innerHTML = '';

const questions = allQuestions[selectedCategory];

questions.forEach((q, i) => {

const div = document.createElement('div');

div.className = 'question';

div.innerHTML = `

<p><strong>${i + 1}. ${q.question}</strong></p>

${q.options.map((opt, j) => `

<label><input type="radio" name="q${i}" value="${j}"> ${opt}</label><br>

`).join('')}

<div class="explanation" id="exp${i}">${q.explanation}</div>

`;

quizContent.appendChild(div);

});

}

function submitQuiz() {

const user = JSON.parse(localStorage.getItem("quizUser"));

const questions = allQuestions[selectedCategory];

let score = 0;

questions.forEach((q, i) => {

const selected = document.querySelector(`input[name="q${i}"]:checked`);

const expDiv = document.getElementById(`exp${i}`);

if (selected && parseInt(selected.value) === q.answer) {

score++;

} else {

expDiv.style.display = 'block';

}

});

const leaderboard = JSON.parse(localStorage.getItem('leaderboard') || '[]');

leaderboard.push({ name: user.name, score });

leaderboard.sort((a, b) => b.score - a.score);

localStorage.setItem('leaderboard', JSON.stringify(leaderboard));

alert(`${user.name}, you scored ${score} out of ${questions.length}`);

displayLeaderboard();

}

function displayLeaderboard() {

document.getElementById('quiz-box').style.display = 'none';

document.getElementById('leaderboard').style.display = 'block';

document.getElementById('okay-btn').style.display = 'inline-block';

document.getElementById('logout-btn').style.display = 'none';

const leaderboard = JSON.parse(localStorage.getItem('leaderboard') || '[]');

const list = document.getElementById('leaderboard-list');

list.innerHTML = '';

leaderboard.slice(0, 10).forEach((entry, i) => {

list.innerHTML += `<li>#${i + 1} - ${entry.name}: ${entry.score}</li>`;

});

}

function showLogout() {

document.getElementById('okay-btn').style.display = 'none';

document.getElementById('logout-btn').style.display = 'inline-block';

}

function logout() {

localStorage.removeItem("quizUser");

document.getElementById('leaderboard').style.display = 'none';

document.getElementById('login-box').style.display = 'block';

document.body.style.background = "url('Quiz1.jpeg') no-repeat center center fixed";

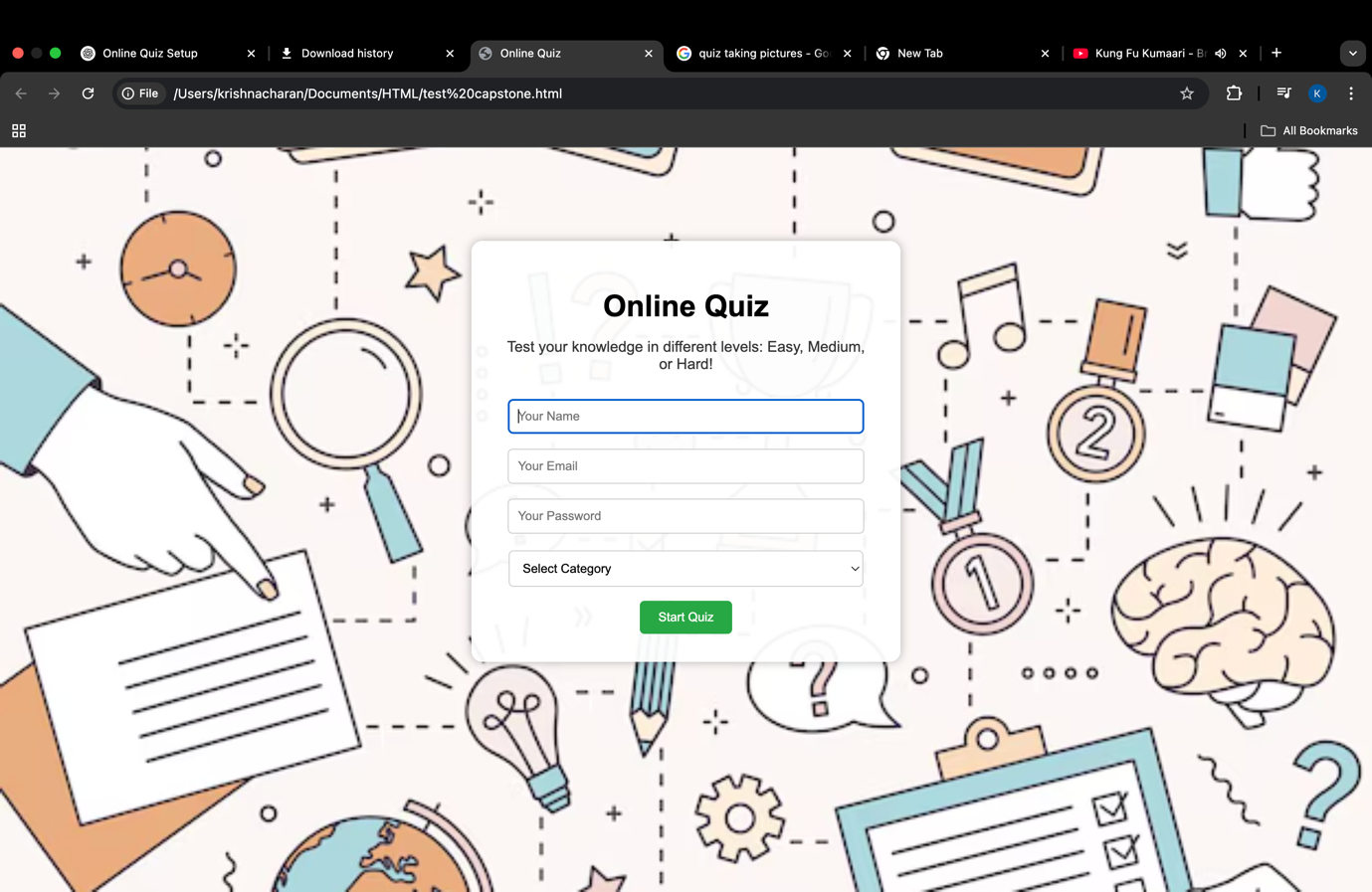
}

</script>

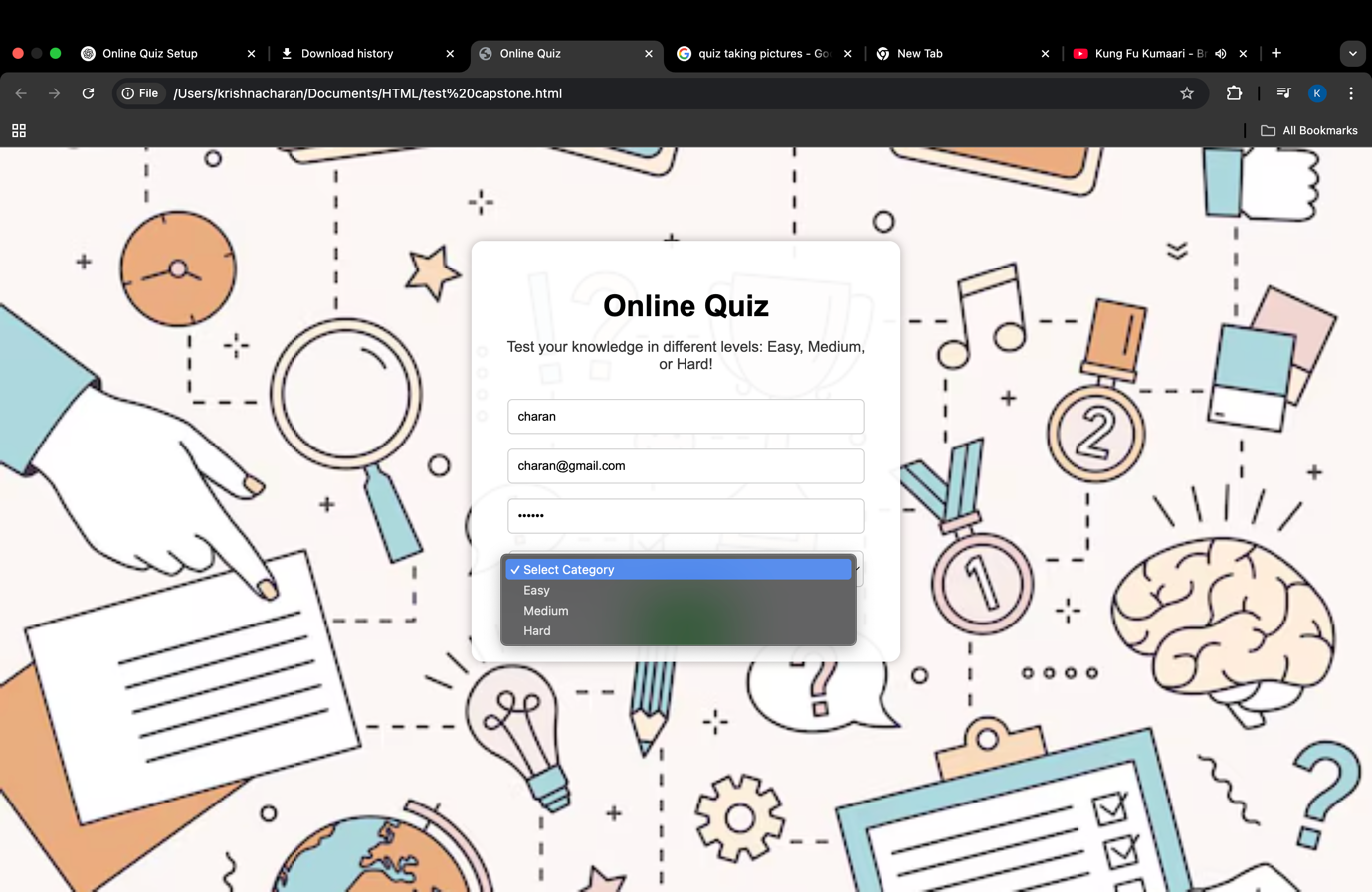
</body>

</html>

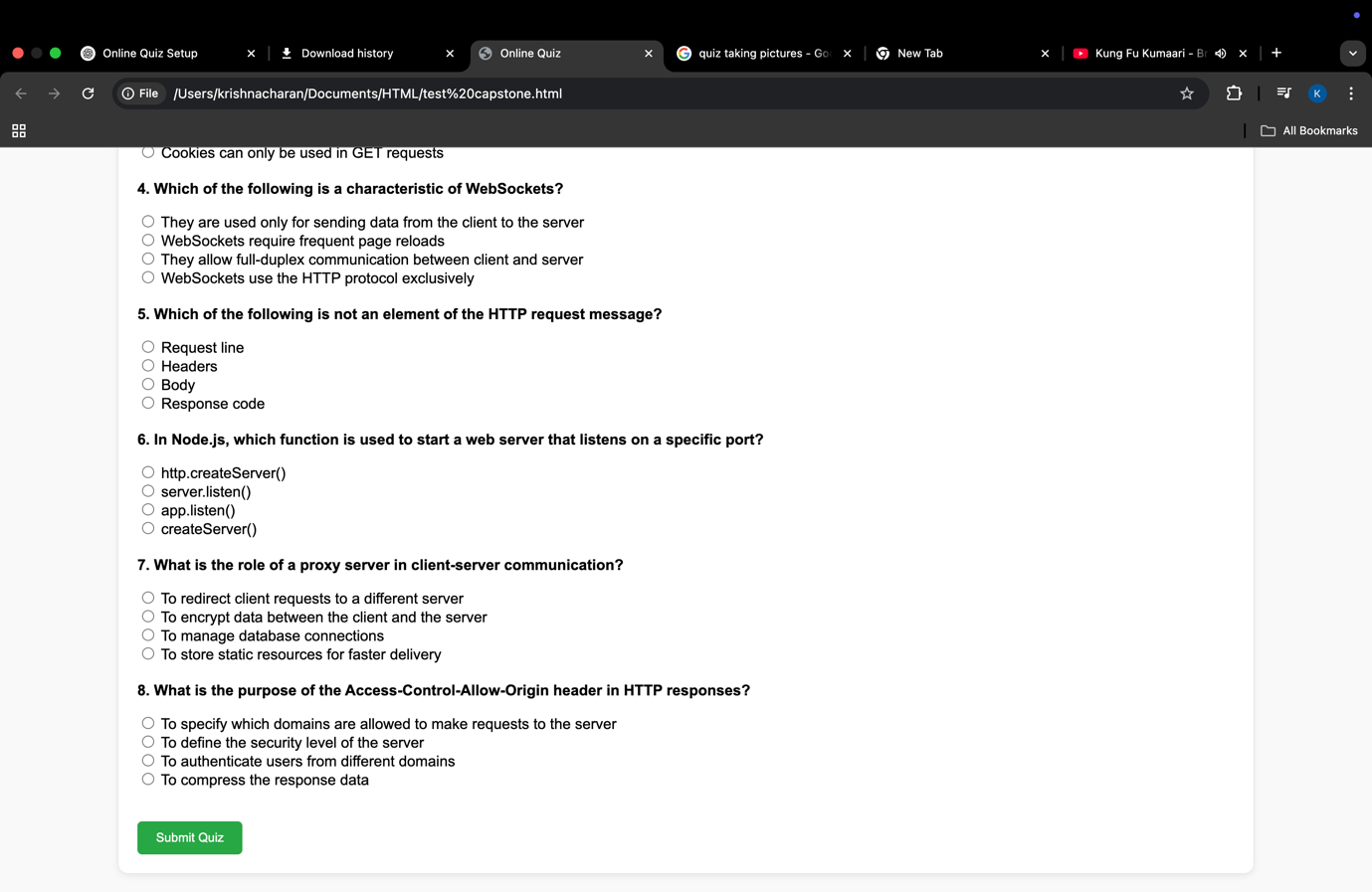
**RESULT**

****

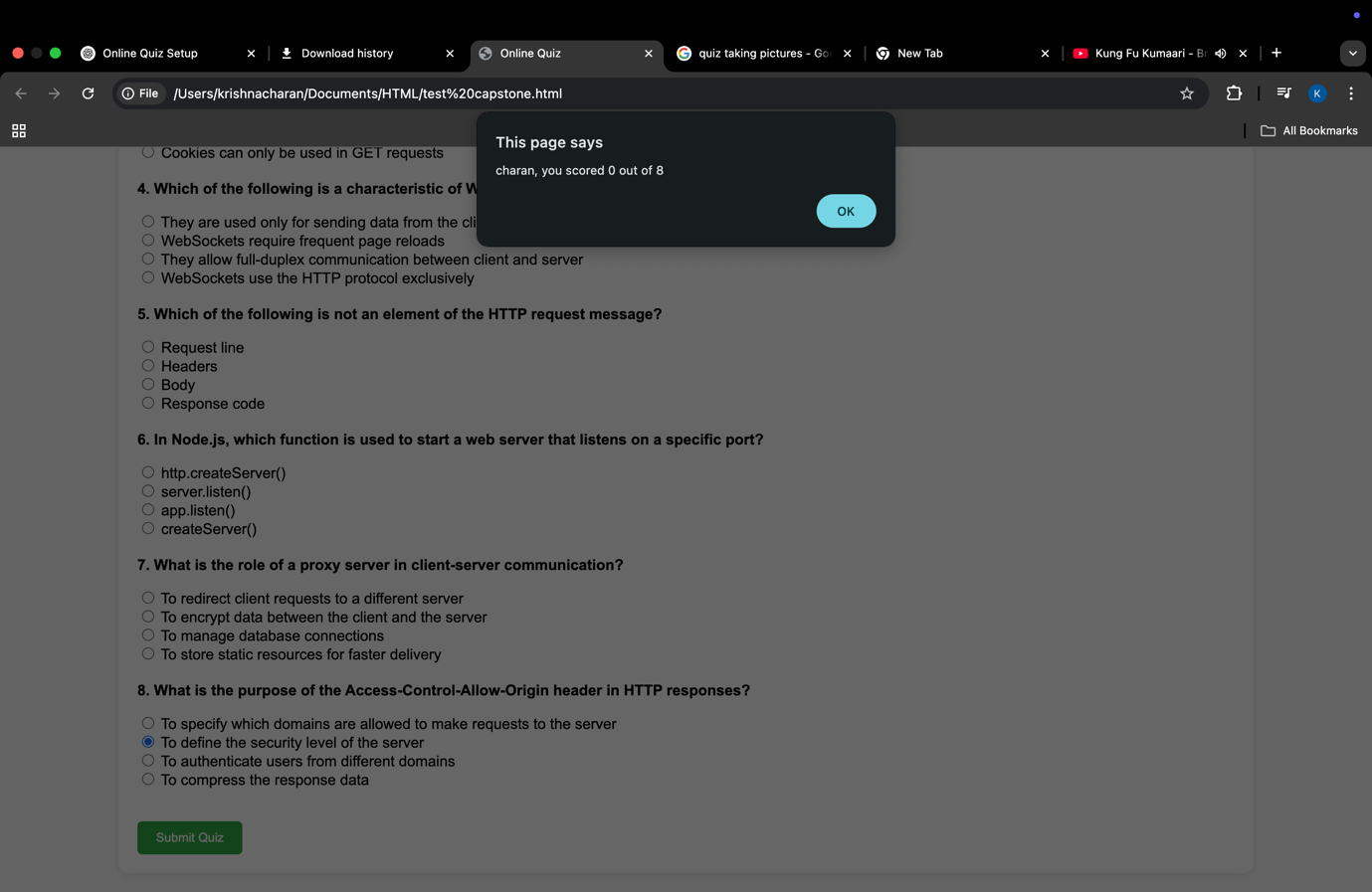
**Figure 1: Home page**

****

**Figure 2: Select Category**

****

**Figure 3: Quiz**



**Figure 4: Result**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Figure5: Leaderboard**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Figure6: Logout**

**CONCLUSION**

The Online Quiz System with Leaderboard project successfully demonstrates the development of an interactive, educational, and competitive platform designed to assess users' knowledge through an engaging online quiz format. By utilizing modern web technologies and a modular development approach, the system ensures an intuitive user experience, real-time feedback, and dynamic leaderboard functionality.

Throughout the project, we accomplished several key objectives:

* Designed a clean and responsive quiz interface that supports 15 multiple-choice questions.
* Implemented real-time evaluation logic with detailed feedback for incorrect answers, enhancing the learning experience.
* Developed a live leaderboard that records and displays user scores, promoting a competitive environment.
* Ensured scalability and maintainability through a modular codebase, supporting potential enhancements like timed quizzes, category-based questions, or user accounts.

This project not only meets the core requirements of an online quiz platform but also lays the groundwork for future expansion. Advanced features such as admin-controlled question banks, adaptive quizzes, authentication, and performance analytics can be integrated seamlessly due to the system's flexible architecture.

Overall, the Online Quiz System with Leaderboard is a practical demonstration of front-end and (optionally) full-stack web development skills, emphasizing user engagement, learning reinforcement, and system scalability. It highlights our ability to build real-world educational tools that are efficient, reliable, and learner-focused.

**REFERENCE**

1. W3Schools HTML Quiz Tutorial: https://www.w3schools.com/html/html\_quiz.asp
2. JavaScript Quiz Application Guide – freeCodeCamp: https://www.freecodecamp.org/news/javascript-quiz-tutorial/
3. MDN Web Docs – JavaScript: https://developer.mozilla.org/en-US/docs/Web/JavaScript
4. MDN Web Docs – DOM Manipulation: https://developer.mozilla.org/en-US/docs/Web/API/Document\_Object\_Model
5. LocalStorage API – MDN: https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage
6. CSS Styling Basics – W3Schools: https://www.w3schools.com/css/
7. Building an Interactive Quiz – JavaScript.info: https://javascript.info/task/quiz
8. HTML Forms Documentation – W3Schools: https://www.w3schools.com/html/html\_forms.asp
9. Creating a Leaderboard in JavaScript – GeeksforGeeks: https://www.geeksforgeeks.org/how-to-create-leaderboard-using-javascript/
10. Responsive Web Design Principles – freeCodeCamp:https://www.freecodecamp.org/learn/responsive-web-design/