PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

Bandepalli Surya Anjani Kumar (Reg. No – 40110156)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

OCT 2022

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by

NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI

CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **BANDEPALLI SURYA ANJANI KUMAR (40110156)** carried out the project entitled "**ONLINE QUIZ WEB APPLICATION**" under my supervision from Aug 2022 to Oct 2022.

Internal Guide

Mr. Amandeep Singh K, B.E., M.tech, (Ph. D)

Head of the Department

Dr. L. Lakshmanan, M.E., Ph. D

Dr. S. Vigneshwari, M.E., Ph. D

Submitted for Viva-voce Examination held on _____

Internal Examiner **External Examiner**

DECLARATION

I, **BANDEPALLI SURYA ANJANI KUMAR (Reg. No – 40110156),** hereby declare that the project report entitled "**ONLINE QUIZ WEB APPLICATION"** was done by me under the guidance of Mr. K. Amandeep Singh**.** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE:

PLACE: Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to the **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph. D**, Dean, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D.,** and **Dr. L. Lakshmanan, M.E., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide, **Mr. Amandeep Singh K, B.E., M.tech, (Ph.D**) for his valuable guidance, suggestions, and constant encouragement that paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING-CERTIFICATE

ABSTRACT

The purpose of this project is to create a responsive and user-friendly quiz app that can serve both teachers and students.

There will be a limited number of questions. For every correct answer, the student will get a credit score. They can also see their progress feedback while attempting the quiz.

Teachers can

- Add
- Update
- Remove questions and
- Read students' scores.

Through this, students can assess and improve their knowledge, and teachers are relieved from the manual grading process.

Additional features like

- A Timer for each question
- Tab-switch prevention
- Mobile-friendly design
- Question order randomizer are applied.

The technologies used in the front end were HTML5, CSS3, Bootstrap 5, and JavaScript. They united with the back end built using NodeJS, NPM, ExpressJS, REST API, and MongoDB in unison with Mongoose. The web application was deployed to the Internet using Heroku.

The main objective is to enable students to practice and teachers to analyze the students' understanding of a particular topic.

CONTENTS

Index	Table of contents	Page Number
	Abstract	**i**
	List of Figures	**ii**
	List of Abbreviations	**iv**
1	**Chapter 1** INTRODUCTION	**1**
1.1	Importance of taking a Quiz	1
1.2	Introduction to Web Development	1
1.3	Introduction to Front-end Web Development	2
1.4	Introduction to Back-end Web Development	2
1.5	What is an API?	2
1.6	What is a Database?	2
1.7	What is RWD?	2
1.8	What is Deployment?	3
2	**Chapter 2** AIM AND SCOPE OF PRESENT INVESTIGATION	**4**
2.1	Present Investigation	4
2.2	Aim	4
2.3	Scope	5
3	**Chapter 3** METHODS AND ALGORITHMS USED	**6**
3.1.1	Hardware requirements to develop the project	6
3.1.2	Software requirements to develop the project	6

3.1.3	Technologies used	7
3.1.3.1	Front-end	7
3.1.3.2	Back-end	10
3.1.3.3	Deployment	12
3.1.4	Source Code	13
4	**Chapter 4** RESULTS AND ANALYSIS	**27**
4.1	Client requirements	27
4.2	Testing conditions	28
4.3	Analysis	28
5	**Chapter 5** CONCLUSION	**30**
	References	31
	Snapshots of the App	32

LIST OF FIGURES

Figure No.	Figure Title	Page Number
2.1	Kahoot's last year stats	5
3.1	Bootstrap CDN	7
3.2	jQuery CDN	8
3.3	Tab-Switch Prevention Function	8
3.4	Random Question Number Generator Function	8
3.5	Score Incrementor Function	9
3.6	Score Table Generator Function	9
3.7	Timer Function	9
3.8	Instantiation of Body-Parser	10
3.9	Instantiation of Mongoose	10
3.10	Routing in Express	11
3.11	Schema for Questions	12
3.12	Schema for Students	12
3.13	Mongoose Query to Add Questions to Database	12

3.14	Mongoose Query to Delete Questions in Database	12
3.15	HTML Code template for all Pages	13
3.16	A Gist of the CSS Code	16
3.17	Asynchronous JavaScript for Fetching Questions from Database	22
3.18	Function to Initiate the Start of the Quiz	23
3.19	Function to Trigger the End of the Quiz	23
3.20	Function for formatting the Received JSON data from the Database and Timer	23
3.21	Function to Increment Score on Correct Answer	25
3.22	DOM Manipulation to Update Score on Client-Side	25
3.23	Initializing all npm packages	26
3.24	Mongoose Query to Connect with Local Database	26

LIST OF ABBREVIATIONS

ABBREVIATION	*EXPANSION*
1. HTML	Hyper Text Markup Language
2. CSS	Cascading Style Sheets
3. API	Application Programming Interface
4. HTTP	Hypertext Transfer Protocol
5. REST	Representational State Transfer
6. NPM	Node Package Manager
7. RWD	Responsive Website Development
8. JS	JavaScript
9. ES6	ECMAScript 6
10. DB	Database
11. DBMS	Database Management Systems
12. CDN	Content Delivery Network

CHAPTER 1

INTRODUCTION

There are several ways to educate and reinforce what we have learnt in the realm of education. Quizzes are one resource that is growing increasingly popular, particularly in online education. It is without dispute that exams and quizzes are valuable. Most educational institutions make use of quizzes as a way to make learning fun.

1.1 Importance of taking a Quiz

Reading information as a way of learning does have its advantages. But reading information and then taking a quiz is much more effective. Forcing your brain to retrieve data ensures that it becomes "embedded" for use in the future. Quizzes identify gaps in knowledge and highlight any areas that need more revision. Given the plethora of studies demonstrating the value of quizzes, it is crucial to make sure that students and the educational community as a whole adopt the practice whenever possible.

Workshops, educational institutions, and workplace engagement areas might use this project as a tool to assess individuals. The app is designed to facilitate the users to complete quick tests also on mobile devices like smartphones and tablets.

1.2 Introduction to Web Development

Web developers often work for clients who are trying to get their product or service onto the web. The work is typically very project focused and involves collaborating with a team that helps to coordinate the client’s needs into the end product. The client could be an individual, a tech company, an organization, or a government. The work could involve

- Front-end (Client-Side Development)
- Back-end (Server-Side Development) [or]
- Full-stack web development.

1.3 Introduction to Front-end Web Development

The front-end is the stuff you see on the website in your browser, including the presentation of content and user interface elements like the navigation bar. Front-end developers use HTML, CSS, JavaScript, and their relevant frameworks to ensure that content is presented effectively and that users have an excellent experience.

1.4 Introduction to Back-end Web Development

The back-end refers to the guts of the application, which live on the server. The back-end stores and serves program data to ensure that the front end has what it needs. This process can become very complicated when a website has millions of users. Back-end developers use programming languages like JavaScript, Java, Python, and Ruby to work with data.

1.5 What is an API?

API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you're using an API. Modern APIs adhere to standards (typically HTTP and REST), that are developer-friendly, easily accessible and understood broadly.

1.6 What is a Database?

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

1.7 What is RWD?

Responsive web design (RWD) is about creating web pages that look good on all

devices!
A responsive web design will automatically adjust for different screen sizes and viewports. Some frameworks of CSS like Bootstrap are responsive by default.

1.8 What is Deployment?

Deployment is the process of making software available to be used on a system by users and other programs.

Chapter 3 of this documentation provides thorough descriptions of the aforementioned subjects and the chosen approach and methods that were used for this project.

CHAPTER 2

AIM AND SCOPE OF THE PRESENT INVESTIGATION

2.1 PRESENT INVESTIGATION:

There are currently a number of quiz applications online. However, most of them are complex and have a crowded user interface. The remaining ones are large corporations that either utilize cookie-based data collection to invade users' privacy or use paid subscription schemes. It was observed that there was a need for an open source, responsive quiz app.

2.2 AIM:

The primary goals of this project were to create a web application with the following features:

1. **Open-Source**
The application would be completely transparent to the users. All programmers will now have access to the code and can modify it as required.

2. **Responsive Design**
Apart from ensuring good usability, the website should render properly on all screen sizes and resolutions. If the information is being viewed on a tablet, phone, or PC, it should take into account the variety of devices and device sizes, enabling automatic adaptation to the screen.

3. **Data Storage**
The capability to save quiz data and export it as needed was essential. It was crucial to have the ability to retrieve and deliver data promptly.

4. **Dynamic Web Application**
The application must adapt to user input, system events, and data in order to modify its content, presentation, and functionality.

5. Malpractice Prevention

Candidates appearing for the exam may try to overcome security aspect of online exam in various ways. In addition, a question order randomizer algorithm, tab-switch prevention, and a timer for each question were implemented. Authentication for educators was also put into place.

2.3 SCOPE:

Teachers are always in search of different ways to monitor their students' performance in real-time. Quizzes might not be the best way to help your class learn, but they are still useful in memorizing facts and assessing a student's knowledge at the end of the lecture.
Consequently, they will be a real requirement for the education sector as well as in the corporate world.

Kahoot, a leading provider of quiz-app services, has served over 100 billion players in the previous year. This demonstrates the market's viability.

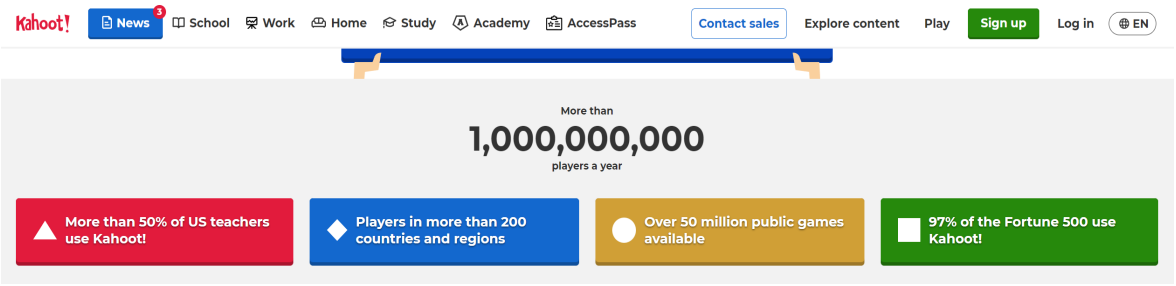


Figure 2. 1 Kahoot's last year stats

CHAPTER 3

METHODS AND ALGORITHMS USED

3.1.1 HARDWARE REQUIREMENTS TO DEVELOP THE PROJECT:

1. CPU: Intel i3 7th Gen or higher
2. RAM: 4 GB or higher
3. Storage: 10 GB available space recommended
4. Internet Connection: Required

3.1.2 SOFTWARE REQUIREMENTS TO DEVELOP THE PROJECT:

1. VS Code – (Recommended Code Editor)

Extensions that have to be installed:

- Prettier (Code Formatter)
- Auto Close Tag
- IntelliCode
- jQuery Code Snippets

2. MongoDB – v6.0.1 (Atlas Account for Cloud Database and Compass GUI if required)

3. NodeJS – v16.16.0 or later

4. NPM – v8.11.0 or later

Dependencies that have to be installed:

- **body-parser v1.20.0**
- **express v4.18.1**
- **mongoose v6.5.4**
- **nodemon (for server auto-refresh after changes)**
- **git v2.37.0**

5. Heroku – v7.63.0 or later

3.1.3 TECHNOLOGIES USED:

3.1.3.1 FRONT-END

1. HTML:

The preferred markup language for Web pages is HTML. This project made use of HTML5, the most recent version of HTML.

2. CSS:

The language we employ to style an HTML document is CSS. The latest release of CSS, CSS3, was used to dictate how HTML components should be rendered in this project.

Bootstrap:

A free and open-source CSS framework called Bootstrap which is designed for front-end web development that prioritizes mobile responsiveness. It was incorporated into the project as well.

```
<link
      rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/dist/css/bootstrap.min.css"
      integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
      crossorigin="anonymous"
/>
```

Figure 3. 1 Bootstrap CDN

3. JavaScript

The Web's primary programming language is JavaScript. The newest version, ES6, was applied to the project.

jQuery:

jQuery is a JavaScript library created to make event handling, CSS animation, and

HTML DOM tree navigation and manipulation simple. The project also made advantage of it.

```
<script
      src="https://code.jquery.com/jquery-
3.2.1.slim.min.js"
      integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXp
G5KkN"
      crossorigin="anonymous"
></script>
```

Figure 3. 2 jQuery CDN

ALGORITHMS USED:

```
function focus() {
  var count = 3;
  window.addEventListener("blur", function () {
    count--;
    if (count === 0) {
      localStorage.setItem("mostRecentScore", score);
      return window.location.replace("end.html");
    }
    alert(
      "Tab Switch is Prohibited!, You have " +
        (count - 1) +
        " accidental-switch(es) left!"
    );
  });
}
```

Figure 3. 3 Tab-Switch Prevention Function

```
const questionIndex = Math.floor(Math.random() *
availableQuestions.length);
  currentQuestion = availableQuestions[questionIndex];
  question.innerText = currentQuestion.question;
  availableQuestions.splice(questionIndex, 1);
```

Figure 3. 4 Random Question Number Generator Function

```
incrementScore = (num) => {
  score += num;
  scoreText.innerText = score;
};
```

Figure 3. 5 Score Incrementor Function

```
function buildTable(data) {
  var table = document.getElementById("myTable");
  for (var i = 0; i < data.length; i++) {
    var row = `<tr>
          <td>${i + 1}</td>
          <td>${data[i].name}</td>
          <td>${data[i].score}</td>
        </tr>`;
    table.innerHTML += row;
  }
}
```

Figure 3. 6 Score Table Generator Function

```
var count = 16;
var interval = setInterval(function () {
  document.getElementById("timer").innerHTML = count - 1;
  count--;
  if (count === 0) {
    getNewQuestion();
    count = 16;
  }
}, 1000);
```

Figure 3. 7 Timer Function

3.1.3.2 BACK-END

1. NodeJS

Node.js is a back-end JavaScript runtime environment that is open-source, cross-platform, and runs on a JavaScript Engine. It executes JavaScript code outside of a web browser. Over Django, Ruby on Rails, and Flask, this was chosen.

NPM

NPM is the package manager for JavaScript. It provides a command-line interface called npm. The npm packages used are:

- **body-parser v1.20.0**

```
const bodyParser = require("body-parser");
```

Figure 3. 8 Instantiation of Body-Parser

- **mongoose v6.5.4**

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

Figure 3. 9 Instantiation of Mongoose

ExpressJS

Express.js is a free and open-source web application framework for Node.js that helps developers create RESTful APIs. It is made for creating APIs and web applications. It was used to get and post requests from the application's front end to the back end.

```
const express = require("express");
const app = express();
app.get("/", function (req, res) {
  res.sendFile(__dirname + "/index.html");
});

app.get("/quiz.html", function (req, res) {
  res.sendFile(__dirname + "/quiz.html");
});

app.get("/api/questions", async function (req, res) {
  let response = await Quiz.findOne({ quizId: 0 },
"questions").exec();
```

```
  res.setHeader("Content-Type", "application/json");
  questionsr = response.questions;
  res.end(JSON.stringify(questionsr));
});
app.get("/examinerLogin.html", function (req, res) {
  res.sendFile(__dirname + "/examinerLogin.html");
});

app.get("/questionsAdded.html", function (req, res) {
  res.sendFile(__dirname + "/questionsAdded.html");
});

app.get("/index.html", function (req, res) {
  res.sendFile(__dirname + "/index.html");
});

app.get("/scoresViewer.html", function (req, res) {
  res.sendFile(__dirname + "/scoresViewer.html");
});
```

Figure 3. 10 Routing in Express

2. MongoDB

MongoDB is a cross-platform document-oriented database application that is open source. MongoDB, a NoSQL database application, employs documents that resemble JSON and may or may not include schemas. The database for this project was created using it.

```
const questionsSchema = {
  quizId: Number,
  questions: [
    {
      qno: Number,
      question: String,
      choice1: String,
      choice2: String,
      choice3: String,
```

```
      choice4: String,
      answer: Number,
    },
  ],
};
```

Figure 3. 11 Schema for Questions

```
const userSchema = {
  name: String,
  score: Number,
};
```

Figure 3. 12 Schema for Students

```
Quiz.create(questionsall, function (err) {
    if (err) {
      console.log(err);
    } else {
      console.log("New Questions Added");
    }
  });
```

Figure 3. 13 Mongoose Query to Add Questions to Database

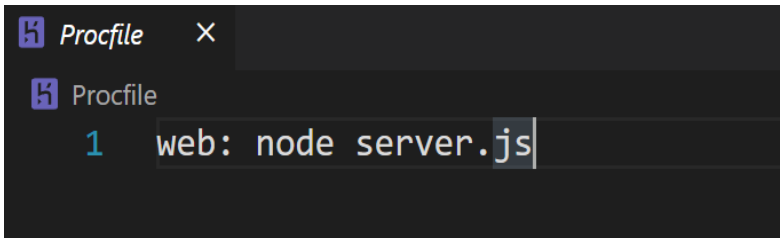
```
Quiz.deleteMany({ answer: { $gte: 0 } }).catch(function
(error) {
    console.log(error);
  });
```

Figure 3. 14 Mongoose Query to Delete Question in Database

3.1.3.3 DEPLOYMENT

1. Heroku

A platform as a service (PaaS) called Heroku allows programmers to create, launch, and manage apps fully in the cloud. This NodeJS application was published on the internet using it. Proc File is the file specific to Heroku.



Procfile for Heroku

3.1.4 SOURCE CODE:

HTML Code template for all pages:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width,
initial-scale=1.0" />
    <title>Examinee Home | Quick Quizizz</title>

    <!-- Our Coded CSS File -->
    <link rel="stylesheet" href="css/styles.css" />

    <!-- Bootstrap for CSS -->
    <link
      rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@4.0.0/di
st/css/bootstrap.min.css"
      integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAi
S6JXm"
      crossorigin="anonymous"
    />

    <!-- Montserrat Font -->
    <link rel="preconnect"
href="https://fonts.googleapis.com" />
```

```
    <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin />
    <link
      href="https://fonts.googleapis.com/css2?family=Montse
rrat:wght@100;500&display=swap"
      rel="stylesheet"
    />

    <!-- Favicon for Webpage -->
    <link
      rel="shortcut icon"
      href="images/Favicon.png?v=1.1"
      type="image/x-icon"
    />
  </head>
  <body>
    <!-- Navigation Bar on Top -->
    <nav class="navbar navbar-dark bg shadow-nav"
style="width: 102.9%">
      <a href="index.html" class="mx-auto h1 fs-30"
        ><span class="navbar-brand" id="brand">
          <img
            class="d-inline-block navbar-img mr-2"
            src="images/Favicon.png"
            height="40"
            width="40"
          />
          Quick Quizizz
        </span>
      </a>
    </nav>

    <!-- Caption -->
    <!-- <h1 class="caption">Ready to Get Started?</h1> -->

    <div class="row">
      <!-- Name -->
```

```
      <div class="wrapper col-sm-12 col-md-6 bg1">
        <div class="pad">
          <h2 class="page">Examinee Home : </h2>
          <h2 class="caption" id="blue">What's your
Name?</h2>
          <form
            class="name-form flex2"
            action="quiz.html"
            name="welcome_form"
            onsubmit="submitForm()"
          >
            <input
              maxlength="8"
              required
              type="text"
              name="name"
              autocomplete="off"
              placeholder="[MAX 8 CHARACTERS]"
              autofocus="autofocus"
            />
            <button type="submit">Let's get
started!</button>
            <a href="examinerLogin.html" id="create"
class="page"
              >Examiner Login →</a
            >
          </form>
        </div>
      </div>

      <!-- Image -->
      <div class="wrapper col-sm-6 d-none d-md-block bg2
nav-shadow">
        <div class="button-cover">
          <a href="scoresViewer.html" id="create2"
class="page" onclick="GenerateTable()"
            >Scores →</a
```

```
          >
        </div>
      </div>
    </div>

    <!-- Footer -->
    <footer class="page-footer bg-dark footer fixed-bottom"
style="width: 102.9%">
      <!-- Copyright -->
      <div class="footer-copyright text-center py-3">
        <a href="https://www.linkedin.com/in/bandepalli-
surya"> &copy; Surya</a>
      </div>
    </footer>
  </body>
</html>
```

Figure 3. 15 HTML Template format for all Pages

A Gist of the CSS Code:

```
:root {
  overflow-x: hidden;
}

body {
  position: relative;
  padding: 0;
  margin: 0;
  overflow-x: hidden;
  width: 100%;
}

/* Font-size of Navbar */
#brand {
  font-size: 2.75rem;
}

/* Changing Navbar Title Font */
.navbar {
```

```
  font-family: "Montserrat", sans-serif;
}

.shadow-nav {
  z-index: 9;
  -webkit-box-shadow: -4px 11px 74px -15px rgba(0, 0, 0, 0.7);
  -moz-box-shadow: -4px 11px 74px -15px rgba(0, 0, 0, 0.7);
  box-shadow: -4px 11px 74px -15px rgba(0, 0, 0, 0.7);
}

/* Shadow Beneath Navbar */
.nav-shadow {
  border-radius: 5px;
  -webkit-box-shadow: -9px 17px 79px -10px rgba(0, 0, 0, 0.7);
  -moz-box-shadow: -9px 17px 79px -10px rgba(0, 0, 0, 0.7);
  box-shadow: -9px 17px 79px -10px rgba(0, 0, 0, 0.7);
}

/* Change Color of Navbar */
.bg {
  background-color: #3395ff;
}

/* Setting Font-style for Caption */
.caption {
  padding-top: 2rem;
  padding-bottom: 1rem;
  text-align: center;
  font-family: "Montserrat", sans-serif;
  font-size: 3rem;
  color: #333;
}

/* Wrapper of Content */
.wrapper {
  display: flex;
  align-items: center;
  justify-content: flex-start;
  height: 100vh;
  width: 100vw;
  flex-flow: column;
  overflow: auto;
}
```

```css
.wrapper2 {
  display: flex;
  align-items: center;
  justify-content: flex-start;
  height: 80vh;
  width: 100vw;
  flex-flow: column;
}

/* Seperates Input TextBox and Submit */
.name-form input,
.name-form button {
  display: block;
  width: 100%;
}

/* What is your name styling */
#blue {
  margin-top: 3rem;
  color: white;
}

/* Background Color for Container */
.bg1 {
  background: lightskyblue;
  background: linear-gradient(
    0deg,
    rgba(255, 248, 242, 1) 0%,
    lightskyblue 100%
  );
}

/* Setting Input Characteristics */
.name-form input {
  margin-bottom: 0.5rem;
  background-color: transparent;
  border: none;
  border-bottom: solid 2px white;
  color: white;
  font-size: 3rem;
  font-family: "Montserrat", sans-serif;
  text-align: center;
```

```css
}

/* Button Customization */
.name-form button {
  font-size: xx-large;
  padding: 5px;
  background-color: white;
  border: none;
  font-family: "Montserrat", sans-serif;
  border-radius: 10px;
  transition: 0.4s all;
}

/* Hover Effect */
.name-form button:hover {
  transform: translateY(-3px);
  cursor: pointer;
}

/* Removing Waste of Outline */
.name-form input:focus,
.name-form button:focus {
  outline: none;
}

/* Background for video */
.bg2 {
  background-color: white;
  -webkit-box-shadow: -14px 0px 55px -28px rgba(0, 0, 0, 0.84);
  -moz-box-shadow: -14px 0px 55px -28px rgba(0, 0, 0, 0.84);
  box-shadow: -14px 0px 55px -28px rgba(0, 0, 0, 0.84);

  background: rgb(251, 244, 235);
  background: radial-gradient(
    circle,
    rgba(251, 244, 235, 1) 0%,
    rgba(255, 255, 255, 1) 98%
  );
}

.footer {
  text-align: center;
  font-family: "Montserrat", sans-serif;
```

```css
}

.flex2 {
  display: flex;
  flex-flow: column;
  justify-content: center;
  align-items: center;
  padding-left: 1rem;
}

.quiz {
  margin-top: 4vh;
  width: 80vw;
  height: 80vh;
  background-color: lightskyblue;
  display: flex;
  align-items: center;
  flex-flow: column;
  flex-basis: 100%;
  flex-grow: 1;
  overflow: auto;
}

.quizheader {
  width: 100%;
  height: 10vh;
  background-color: #3395ff;
  color: lightblue;
  box-shadow: 0px 2px 5px 1px rgba(0, 0, 0, 0.4);
  z-index: 1;
  display: flex;
  justify-content: space-between;
  overflow: hidden;
}

.quizbody {
  background-color: lightskyblue;
  width: 100%;
  height: 70vh;
  padding: 1rem;
  overflow: auto;
}
```

```css
.footer2 {
  margin-top: 4.5rem;
}

.quiz-user {
  font-family: "Montserrat", sans-serif;
  display: flex;
  align-items: center;
  padding-left: 1rem;
  color: white;
  background-color: #3395ff;
  padding-right: 1rem;
}

.quiz-timer {
  font-family: "Montserrat", sans-serif;
  display: flex;
  align-items: center;
  padding-right: 1rem;
  font-weight: bolder;
  font-size: xx-large;
  padding-left: 1rem;
  color: white;
  background-color: #3395ff;
  border: 2px solid lightcoral;
  border-radius: 5px;
  margin: 3px;
}

.question {
  font-family: "Montserrat", sans-serif;
  font-size: x-large;
  color: white;
  user-select: none;
}

.choice-container {
  font-family: "Montserrat", sans-serif;
  background-color: aliceblue;
  border: 1px solid #84c5fe;
  border-radius: 5px;
  padding: 5px;
  margin-bottom: 0.25rem;
```

```
  display: flex;
  align-items: center;
  cursor: pointer;
  transition: all 0.3s ease;
}

.choice-prefix {
  font-family: "Montserrat", sans-serif;
  font-size: 1.25rem;
  color: white;
  background-color: #2668e0;
  border-radius: 5px;
  padding: 0.5rem;
}

.choice-text {
  padding: 1rem;
  width: 100%;
  user-select: none;
}

.choice-container:hover {
  background-color: #cce5ff;
  border-color: #b8daff;
}

.last {
  margin-bottom: 0.4rem;
  width: 100%;
}
```

Figure 3. 16 A Gist of the CSS Code

JavaScript:

```
const loadData = async () => {
  const res = await fetch("/api/questions");
  const data = await res.json();
  availableQuestions = data;
  startGame();
};
```

Figure 3. 17 Asynchronous JavaScript for Fetching Questions from Database

```
function startGame() {
  questionCounter = 0;
  score = 0;
  availableQuestions = [...availableQuestions];
  localStorage.setItem("mostRecentScore", score);
  focus();
  getNewQuestion();
}
```

Figure 3. 18 Function to Initiate the Start of the Quiz

```
function endQuiz() {
  localStorage.setItem("mostRecentScore", score);
  return window.location.replace("end.html");
}
```

Figure 3. 19 Function to Trigger the End of the Quiz

```
// Displays Question from Array
function getNewQuestion() {
  if (availableQuestions.length === 0 || questionCounter >=
MAX_QUESTIONS) {
    localStorage.setItem("mostRecentScore", score);
    return window.location.replace("end.html");
  }
  questionCounter++;
  //Change qn no in html
  questionCounterText.innerText = questionCounter + "/" +
MAX_QUESTIONS;

  // Selects Random Question
  const questionIndex = Math.floor(Math.random() *
availableQuestions.length);
  currentQuestion = availableQuestions[questionIndex];

  question.innerText = currentQuestion.question;

  //Displays it's Respective Choices
  choices.forEach((choices) => {
    const number = choices.dataset["number"];
    choices.innerText = currentQuestion["choice" + number];
  });

  //Remove Question to avoid repetition
```

```
  availableQuestions.splice(questionIndex, 1);

  acceptingAnswers = true;
}
choices.forEach((choice) => {
  choice.addEventListener("click", (e) => {
    if (!acceptingAnswers) return;

    acceptingAnswers = false;
    const selectedChoice = e.target;
    const selectedAnswer = selectedChoice.dataset["number"];

    //Change on Correct Answer
    var classToApply = "incorrect";
    if (selectedAnswer == currentQuestion.answer) {
      classToApply = "correct";
    }

    if (classToApply === "correct") {
      incrementScore(CORRECT_BONUS);
      localStorage.setItem("mostRecentScore", score);
    }

    //add class to parent element to display correct or wrong
    selectedChoice.parentElement.classList.add(classToApply);

    //Shows Correct answer if answer is wrong by user
    if (classToApply === "incorrect") {
      switch (currentQuestion.answer) {
        case 1:
          document.getElementById("option1").classList.remove("hid
den");
          setTimeout(() => {
            document.getElementById("option1").classList.add("hidd
en");
          }, 1000);
          break;
        case 2:
          document.getElementById("option2").classList.remove("hid
den");
          setTimeout(() => {
            document.getElementById("option2").classList.add("hidd
en");
```

```
          }, 1000);
          break;
        case 3:
          document.getElementById("option3").classList.remove("hid
den");
          setTimeout(() => {
            document.getElementById("option3").classList.add("hidd
en");
          }, 1000);
          break;
        case 4:
          document.getElementById("option4").classList.remove("hid
den");
          setTimeout(() => {
            document.getElementById("option4").classList.add("hidd
en");
          }, 1000);
          break;
      }
    }

    //remove class before changing to next question
    setTimeout(() => {
      selectedChoice.parentElement.classList.remove(classToApply);
      getNewQuestion();
    }, 1000);

    count = 16;
  });
});
```

Figure 3. 20 Function for formatting the Received JSON data from the Database and Timer

```
//Score Incrementor
incrementScore = (num) => {
  score += num;
  scoreText.innerText = score;
};
```

Figure 3. 21 Function to Increment Score on Correct Answer

```
const finalScore = document.getElementById("finalScore");
finalScore.innerText = mostRecentScore + " / 100";
```

Figure 3. 22 DOM Manipulation to Update Score on Client-Side

```
const express = require("express");
const bodyParser = require("body-parser");
var favicon = require("serve-favicon");
var path = require("path");
const mongoose = require("mongoose");
```

Figure 3. 23 Initializing all npm packages

```
mongoose.connect("mongodb://localhost:27017/quizDataBase", {
  useNewUrlParser: true,
});
```

Figure 3. 24 Mongoose Query to Connect with Local Database

CHAPTER 4

RESULTS AND PERFORMANCE ANALYSIS

4.1 CLIENT REQUIREMENTS:

Minimum Hardware Requirements:

1. CPU: Intel Pentium 4 or later
2. RAM: 2 GB or higher
3. Storage: 2 GB available space recommended
4. Internet Connection: Required

Software Requirements:

5. Latest version of any Browser (Chrome Recommended)

Web Testing, or website testing is checking your web application or website for potential bugs before it is made live and is accessible to general public. Web Testing checks for functionality, usability, security, compatibility, performance of the web application or website.

Testing was done in 4 phases:

1. Functionality Testing

It is the process that includes several testing parameters like user interface, APIs, database testing, security testing, client and server testing and basic website functionalities. Functional testing is very convenient and it allows users to perform both manual and automated testing. It is performed to test the functionalities of each feature on the website.

2. Usability Testing

It has now become a vital part of any web-based project. It can be carried

out like a small focus group similar to the target audience of the web application.

3. Interface Testing

   Three areas to be tested here are – Application, Web and Database Server.

4. Database Testing

Database is one critical component of your web application and stress must be laid to test it thoroughly.

Testing activities will include-

- Test if any errors are shown while executing queries
- Data Integrity is maintained while creating, updating or deleting data in database.

4.2 TESTING CONDITIONS:

Configuration: Ryzen 7 5800H 8 CPU cores, Nvidia GTX 1650, 16 GB RAM Server-Side and Client-Side were run on the local machine.

4.3 ANALYSIS

- Rather than frequently querying the database for questions, a one-time fetch solution where questions are fetched all at once was employed to avoid network bandwidth issues for users while taking the quiz.
- Overflow issues in the front-end were analyzed and fixed.
- The website's other features were all working as they should.
- The responsive home page, buttons, and hyperlinks were tested.
- Quiz page was loading with as less latency as possible.
- The sequence in which the questions were fetched was random.
- Effects on hovering were effective. The Tab-Switch feature was functioning as it should.
- As anticipated, scores and question numbers were adjusted.
- The database was operating as it should.

- Login functionality was tested and was working properly.
- Form Validation for Questions' Adder page was also functioning properly.

CONCLUSION:

- The best efforts were made to develop a web application with no bugs.
- The best coding practice was used throughout the making of the project.
- The project complied with all user criteria.
- To keep up with the latest trends, the project will be iteratively improved.
- In the future, the project will develop to include features like
    - A dashboard for Teachers
    - Student focus checker using machine learning and
    - Support for multiple quizzes.

REFERENCES:

1. Angela Yu, Web-Development Bootcamp, The App Brewery,
   https://www.udemy.com/course/the-complete-web-development-bootcamp/

2. Mosh, The Ultimate HTML/CSS Mastery Series
   https://codewithmosh.com/p/the-ultimate-html-css

3. W3Schools, Web-Dev Guidance
   https://www.w3schools.com/whatis/

4. MDN Web Docs, Mozilla
   https://developer.mozilla.org/en-US/docs/Learn

5. Documentation, Bootstrap
   https://getbootstrap.com/docs/4.5/getting-started/introduction/

6. Mosh, The Ultimate JavaScript Mastery Series
   https://codewithmosh.com/p/javascript-basics-for-beginners

7. Documentation, Mongoose
   https://mongoosejs.com/docs/api/document.html

8. Documentation, Node
   https://nodejs.org/api/

9. Heroku Dev Center, Heroku
   https://devcenter.heroku.com/categories/reference

10. Richard Kalehoff, Version Control with Git
    https://www.udacity.com/course/version-control-with-git--ud123

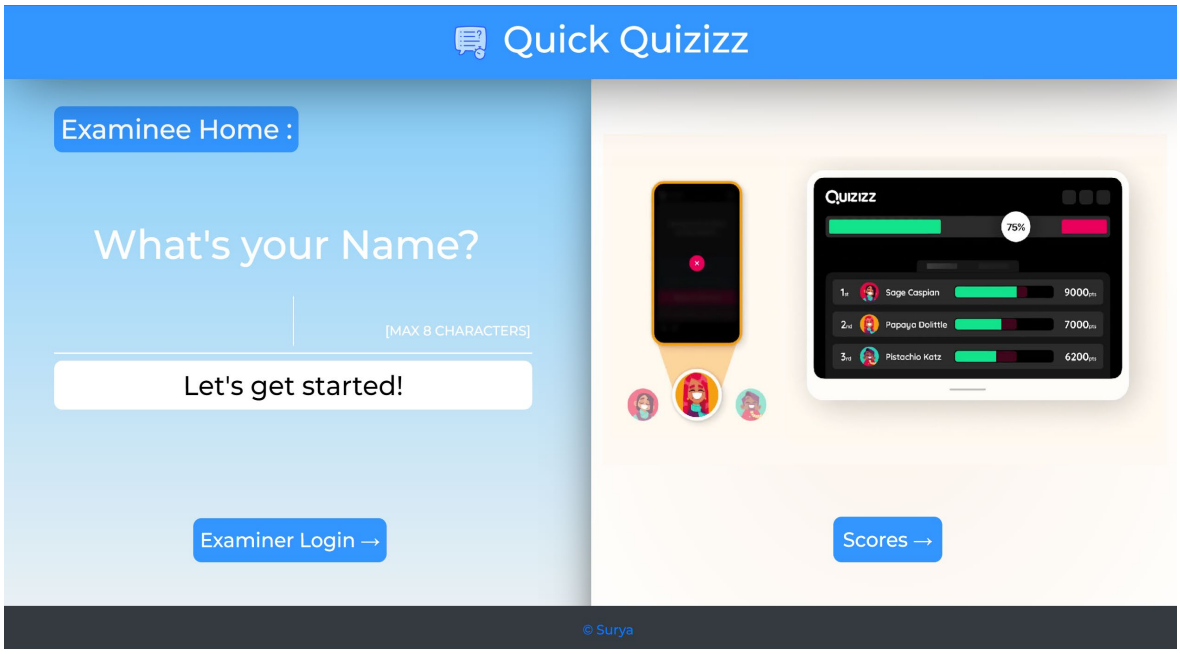
11. Deployment, Heroku
    https://quick-quizizz.herokuapp.com/

12. GitHub Repository
    https:/github.com/Surya-Kumar-03/Quick-Quizizz

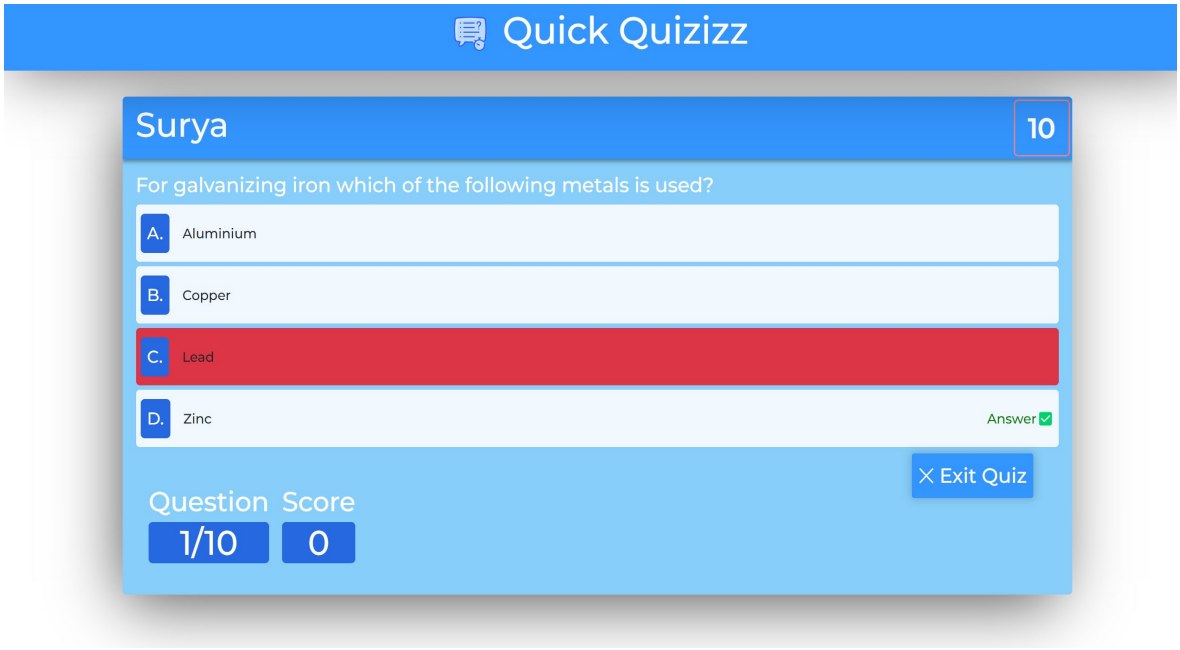
APPENDIX

SNAPSHOTS OF THE WEB–APP

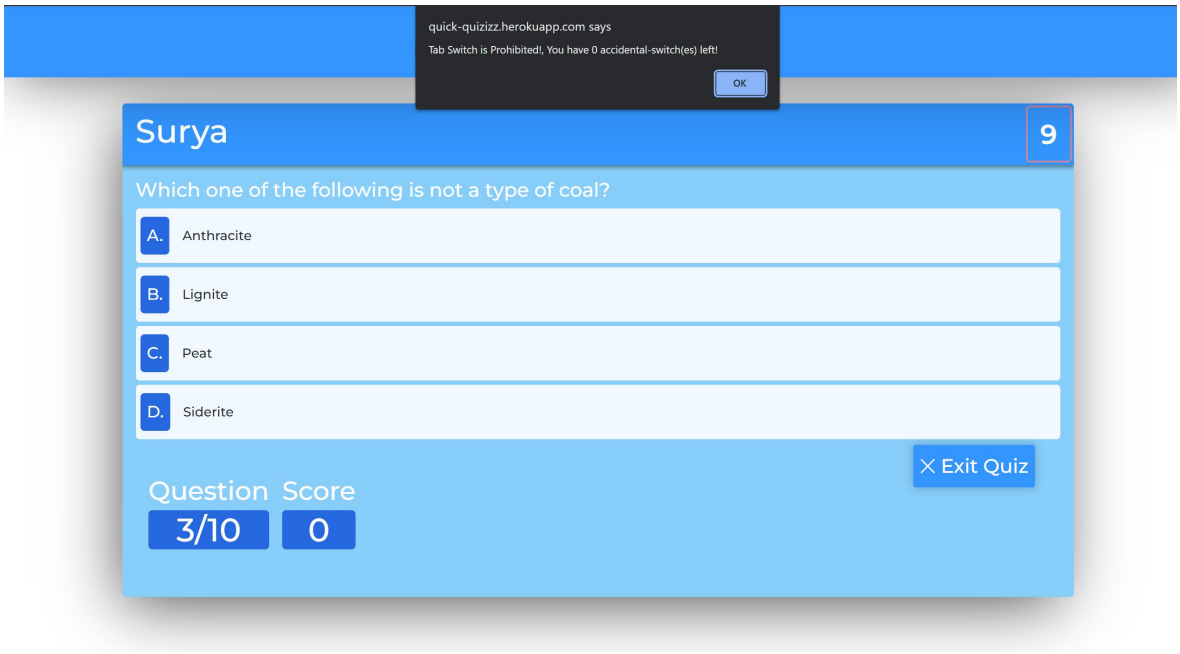
EXAMINEE HOME:



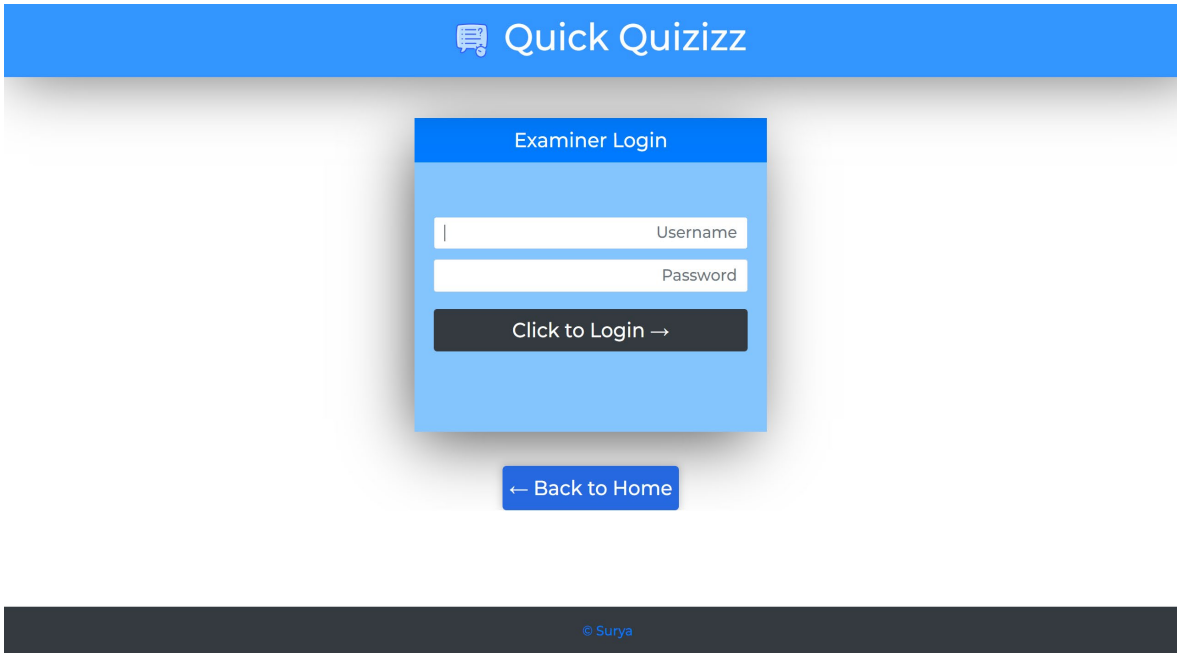
QUIZ PAGE:



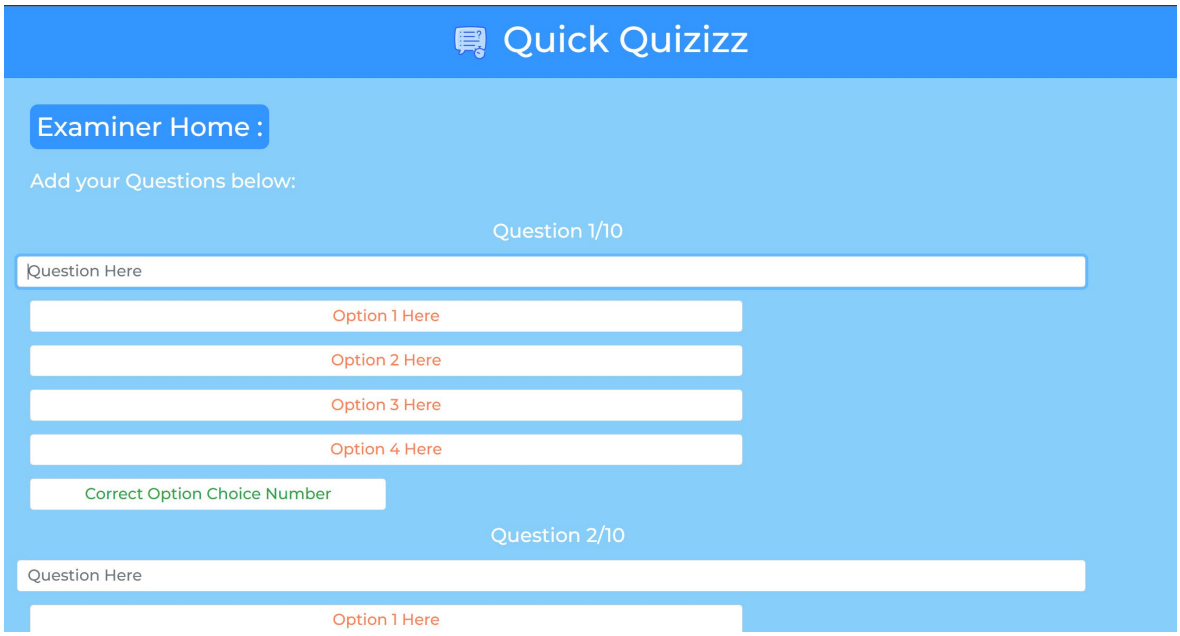
TAB SWITCH PREVENTION:



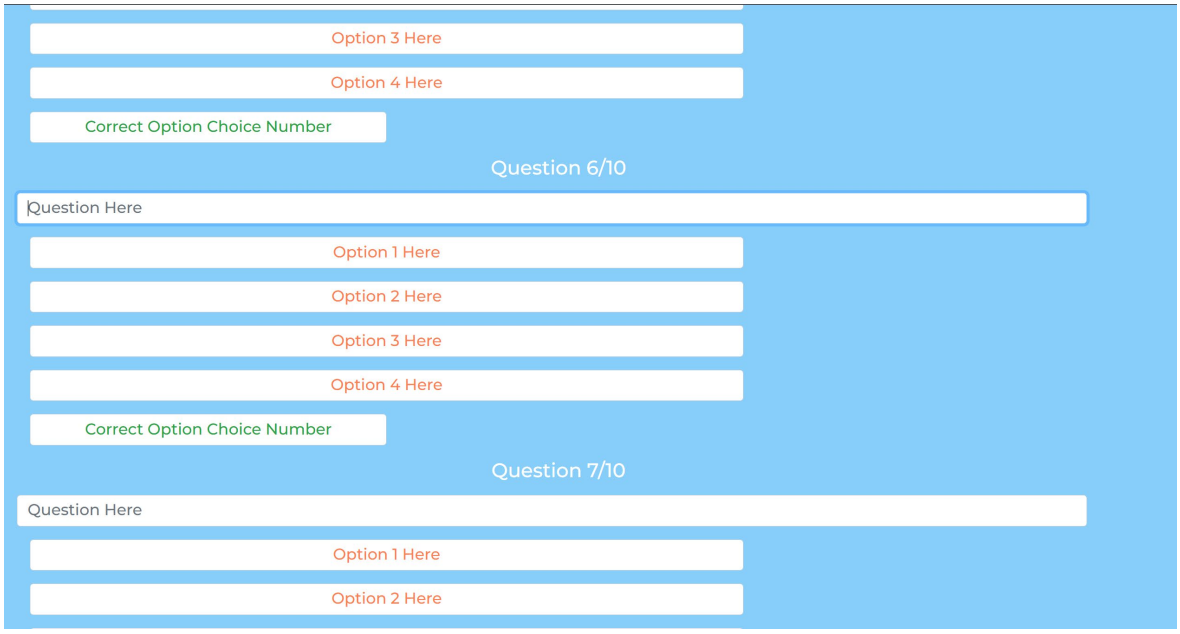
EXAMINER LOGIN:



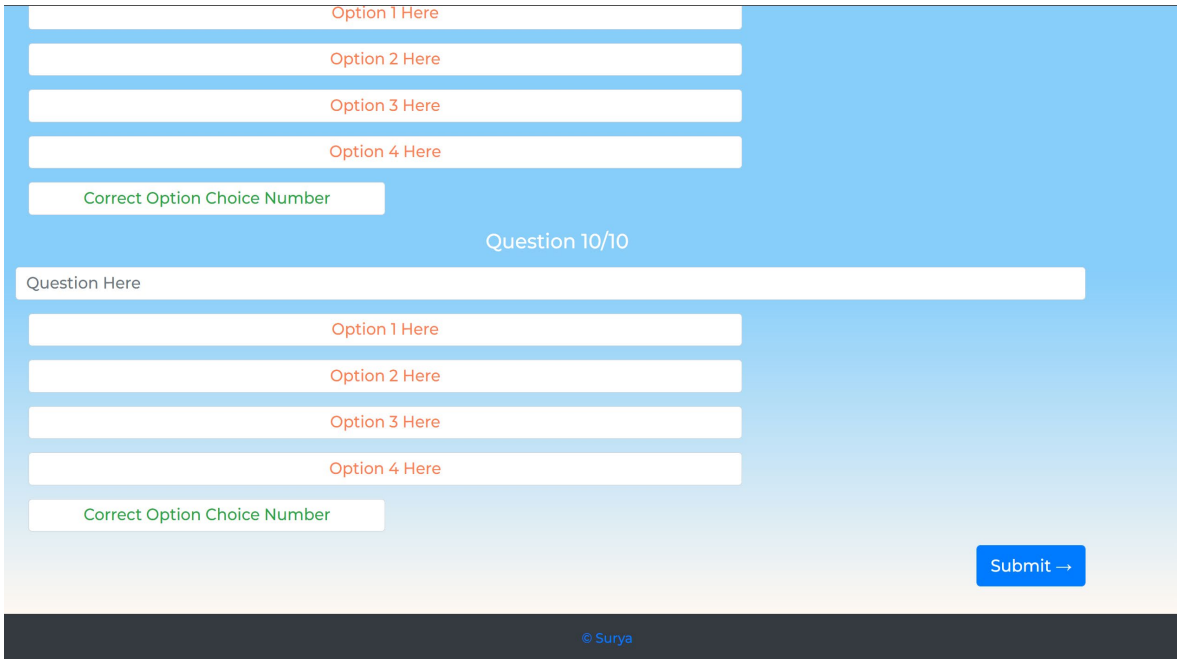
QUESTIONS' ADDER:



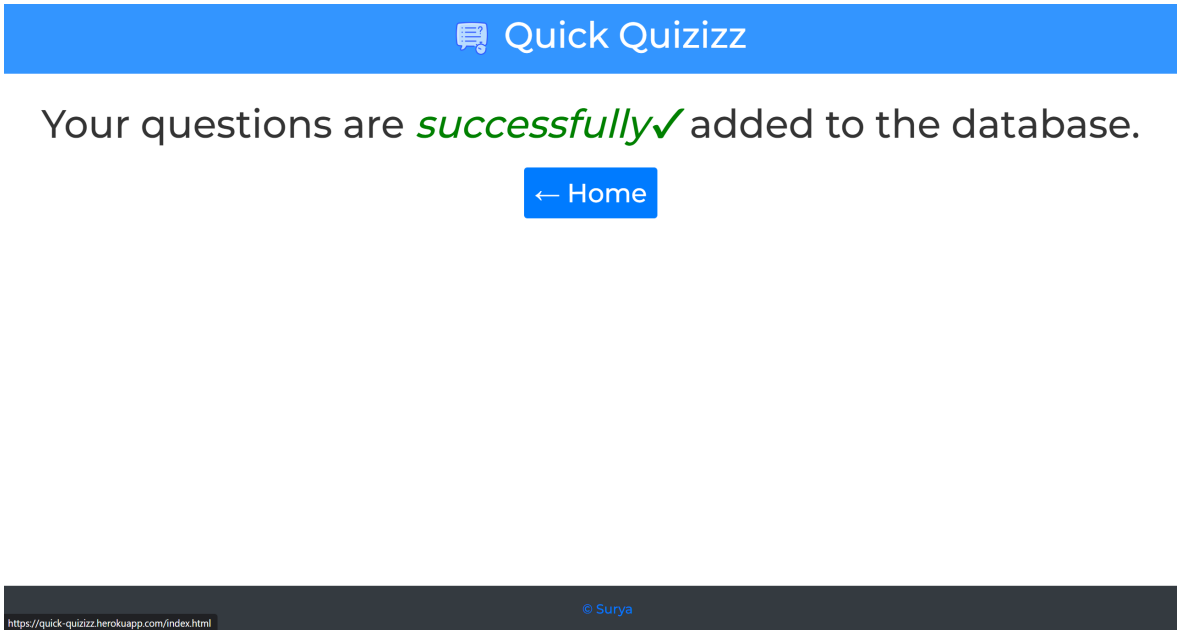
QUESTIONS' ADDER-2:



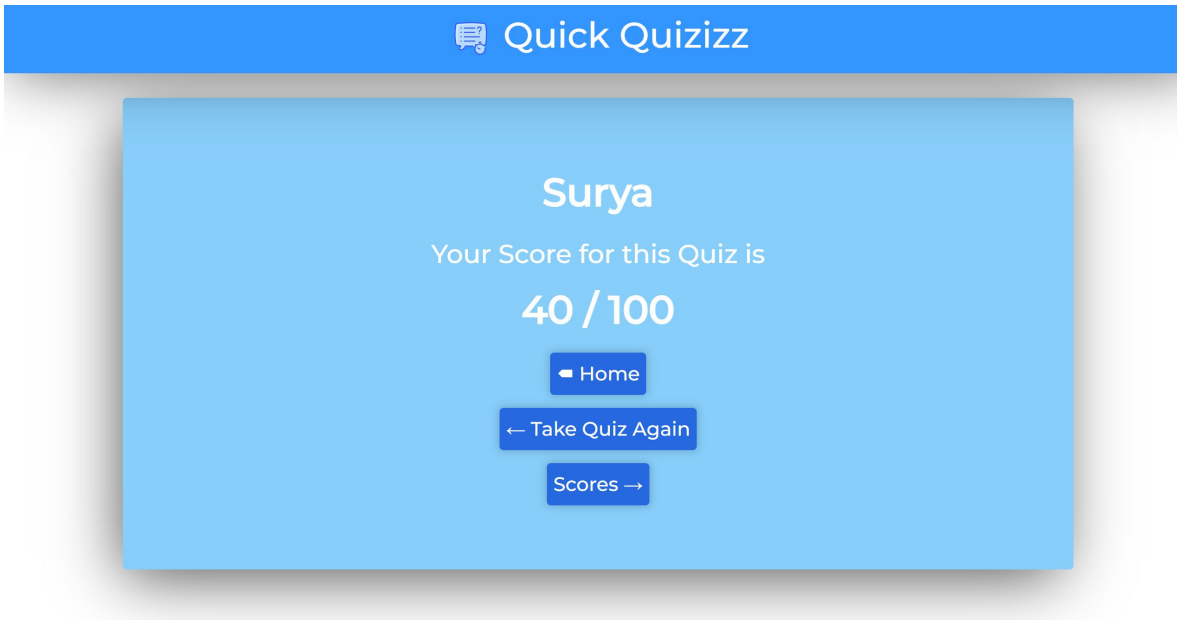
QUESTIONS' ADDER-3:



SUCCESS ACKNOWLEDGEMENT:



QUIZ-END PAGE:



SCORES-DISPLAY PAGE: