QUIZ WEB APPLICATION

Dipti Rathod
California State University - San Bernardino

Follow this and additional works at: https://scholarworks.lib.csusb.edu/etd

Part of the Other Computer Engineering Commons

Recommended Citation
Rathod, Dipti, "QUIZ WEB APPLICATION" (2023). Electronic Theses, Projects, and Dissertations. 1818. https://scholarworks.lib.csusb.edu/etd/1818

This Project is brought to you for free and open access by the Office of Graduate Studies at CSUSB ScholarWorks. It has been accepted for inclusion in Electronic Theses, Projects, and Dissertations by an authorized administrator of CSUSB ScholarWorks. For more information, please contact scholarworks@csusb.edu.
QUIZ WEB APPLICATION

A Project
Presented to the
Faculty of
California State University,
San Bernardino

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
in
Computer Science

by
Dipti Rathod
December 2023
QUIZ WEB APPLICATION

A Project
Presented to the
Faculty of
California State University,
San Bernardino

by
Dipti Rathod

December 2023
Approved by:

Dr. Ronald Salloum, Advisor, Computer Science and Engineering

Dr. Jennifer Jin, Committee Member

Dr. Yan Zhang, Committee Member
ABSTRACT

The Quiz web application is designed to facilitate the process of quiz creation and participation. This web application mainly consists of three roles: Admin, Instructor, and Student. Each role has specific features, functionalities, and permissions. With a user-friendly interface, the admin role can handle the departments, courses, and instructors. This web application also ensures smooth quiz management, allowing the instructors to schedule the upcoming quizzes, create the questions, and manage the students with ease. Student roles have features like taking quizzes and seeing their results. Additionally, this web application includes a significant feature to prevent cheating during online tests, ensuring a fair and accurate assessment of student's knowledge.
ACKNOWLEDGEMENTS

I want to express my gratitude to Dr. Ronald Salloum for helping as my project advisor for the quiz web application. Dr. Ronald Salloum provided continuing support to complete my project successfully. I am also grateful to Dr. Jennifer Jin and Dr. Yan Zhang for assisting as committee members.

I want to express my sincere gratitude for the effort and invaluable support that each of the three committee members contributed to allow me to implement the project successfully.

Thanks to my family and friends for their incredible support during the development of the quiz web application. Without their encouragement, this project would not have been possible.
# TABLE OF CONTENTS

ABSTRACT .................................................................................................................. iii

ACKNOWLEDGEMENTS.............................................................................................. iv

LIST OF FIGURES ...................................................................................................... vii

CHAPTER ONE: INTRODUCTION .............................................................................. 1
  Background ............................................................................................................... 1
  Significance ........................................................................................................... 1

CHAPTER TWO: SYSTEM REQUIREMENTS ............................................................... 2
  Hardware Requirements ........................................................................................ 2
  Software Requirements ....................................................................................... 2

CHAPTER THREE: TOOLS AND TECHNOLOGIES ................................................ 3
  React ..................................................................................................................... 3
  Node.js .................................................................................................................. 3
  Express.js ............................................................................................................ 4
  Sequelize ............................................................................................................... 4
  MySQL ................................................................................................................ 5
  PhpMyAdmin ....................................................................................................... 5
  Xampp .................................................................................................................. 6

CHAPTER FOUR: SYSTEM DESIGN .......................................................................... 8
  UML Representation ............................................................................................ 8
  Use Case Representation ................................................................................... 9
  Activity Representation ..................................................................................... 10
  ER Diagram ........................................................................................................ 10
LIST OF FIGURES

Figure 1 PhpMyadmin ................................................................. 6
Figure 2 Xampp ......................................................................... 7
Figure 3 Architecture Diagram ................................................... 8
Figure 4 Use Case Diagram ......................................................... 9
Figure 5 Activity Diagram .......................................................... 10
Figure 6 ER Diagram ................................................................ 11
Figure 7 Database Connection ................................................... 15
Figure 8 Database .................................................................... 16
Figure 9 Admin Register Page .................................................... 17
Figure 10 Login Page ................................................................. 18
Figure 11 Login Page (Authentication Fail) ................................. 18
Figure 12 Admin Dashboard ....................................................... 19
Figure 13 View / Delete Courses .................................................. 20
Figure 14 Add / Edit Courses ....................................................... 20
Figure 15 View/ Delete Department ............................................. 21
Figure 16 Add/ Edit Department .................................................. 21
Figure 17 View / Delete Instructor .............................................. 22
Figure 18 Add/ Edit Instructor ..................................................... 22
Figure 19 View Students ............................................................ 23
Figure 20 Logout Component ...................................................... 23
Figure 21 Instructor Dashboard .................................................. 24
Figure 22 Add/ Edit Student.......................................................... 25
Figure 23 View / Delete Student....................................................... 25
Figure 24 Add / Edit Quiz ............................................................... 26
Figure 25 View Past / Evaluate Quiz ............................................... 26
Figure 26 View the Result of the Student . ................................. 27
Figure 27 View / Delete Upcoming Quiz .................................... 27
Figure 28 Add / Edit Question......................................................... 28
Figure 29 View/Delete Question ..................................................... 28
Figure 30 Student Dashboard.......................................................... 29
Figure 31 View Upcoming Quiz ...................................................... 30
Figure 32 View Past Quiz ............................................................... 30
Figure 33 Take Quiz ..................................................................... 31
Figure 34 After submitting the Quiz .............................................. 31
Figure 35 Frontend index.js............................................................ 35
Figure 36 Backend server.js .......................................................... 35
CHAPTER ONE:
INTRODUCTION

Background

During the COVID pandemic, educational institutions had to arrange online classes but had limited options to evaluate students' knowledge through online web applications. To address this problem, I developed a quiz web application for educational institutions seeking efficient ways to manage departments, courses, instructors, students, and quizzes.

Significance

The primary significance of the quiz web application is to simplify the process of quiz management for instructors by allowing them to create quizzes and questions according to their courses. The quiz web application provides an automated evaluation functionality, and with this feature, instructors need not worry about evaluating student's responses for each quiz. The web application automatically assesses the student's answers and calculates a score based on the quiz's marking criteria. The user interface of this project is straightforward so that any regular user can use it very efficiently. Moreover, the quiz web application also provides a feature that offers prevention of cheating during the online test, ensuring academic integrity.
CHAPTER TWO:
SYSTEM REQUIREMENTS

Hardware Requirements

Memory (RAM): 4GB
Storage: 64GB
Hard Disk: 160 GB
Processor: Intel i3 Processor

Software Requirements

Operating System: Windows
IDE: Visual Studio Code
Programming Languages: React, Node.js, JavaScript, HTML, SCSS
Database: MySQL
ORM: Sequelize
Tools: XAMPP, phpMyAdmin, npm, npx, GitHub, nodemon
Server: Apache Tomcat
CHAPTER THREE:
TOOLS AND TECHNOLOGIES

React
A JavaScript package called React renders user interfaces (UI). The user interface comprises small components like buttons, text, and graphics [1]. Everything on the website’s screens can be broken into components. I have used the 16.8.4 version of React, initialized the development with npx (Node Package Executes), and installed the required libraries using npm (Node Package Manager). I developed single-page web applications using a component-based structure that provides the virtual DOM (Document Object Model) and implemented user-friendly navigation using the react-router-dom library.

Node.js
The JavaScript runtime environment Node.js is cross-platform and open-source. The V8 JavaScript engine used by Google Chrome is operated outside the browser by Node.js. This makes Node.js extremely robust. A Node.js application does not generate a new thread for each request but operates in a single process. JavaScript code cannot block running servers due to a set of asynchronous I/O primitives included in Node.js’s standard library. Node.js performs I/O operations like reading from a network and accessing a database or filesystem without blocking the thread or wasting CPU cycles [2].
I have used Node LTS 18.18.0 version for a quiz web application. Node.js allows the implementation of a robust backend server. I also used the nodemon package for the contiguous development of the server.

Express.js

Express.js is a Node.js online application framework that is simple to use and adaptable and provides powerful functionalities for web applications. I can quickly build a robust API using various HTTP utility methods and middleware. Express.js maintains Node.js characteristics while offering an essential layer of web application functionality [3]. In the quiz web application, I utilized Express.js to implement Restful APIs, routing, and the MVC (Model View Controller) architecture for the backend.

Sequelize

Sequelize is the latest ORM (Object Relational Mapping) for Oracle, Postgres, MySQL, MariaDB, SQLite, and SQL Server, which runs on TypeScript and Node.js. It supports read replication, eager and lazy loading, relations, and transactions. It easily defines the models and makes automatic database synchronization optional. It also establishes associations between models and handles the heavy lifting of data [4]. I used Sequelize with MySQL database for implementing models and their relations to design and implement a scalable database.
MySQL

A robust, multithreaded, multiuser, and fast SQL (Structured Query Language) database server is provided by the MySQL software. MySQL Server is designed to be embedded into widely distributed software and used in essential, high-load production systems [5].

I used MySQL database in the quiz web application because it provides a reliable relational database management system. MySQL follows the ACID (Atomicity, Consistency, Isolation, and Durability) property, which ensures consistent data in the database even during system failures.

PhpMyAdmin

An open-source software tool called phpMyAdmin was created in PHP to manage MySQL remotely. Several types of MySQL operations are supported by phpMyAdmin. Using the user interface, any SQL query can be quickly executed while managing databases, tables, columns, relations, indexes, users, and permissions [6].

I can quickly check the data, tables, and their relations using phpMyAdmin for the quiz web application (Figure 1).
Xampp

Xampp is a user-friendly Apache distribution that includes MariaDB, PHP, and Perl. The Xampp open-source software is designed to be extremely simple to use and install [7]. In this project, I used MySQL and Apache Moules (shown in Figure 2).
Figure 2. Xampp
CHAPTER FOUR:
SYSTEM DESIGN

UML Representation

UML, or Unified Modeling Language, is standard visualization for software design. The quiz web application’s overall architecture is shown in Figure 3 [8].

Figure 3. Architecture Diagram
Use Case Representation

The interactions between a user and a system can be represented with the help of a use case diagram. Figure 4 shows the interactions of three leading roles in the quiz web application [8].

Figure 4. Use Case Diagram
Activity Representation

Activity diagrams are graphical presentations of workflows of step-by-step activities and actions. Figure 5 demonstrates all activities and actions in the quiz web application [8].

Figure 5. Activity Diagram

ER Diagram

An ER mode is also known as entity–relationship model or ER model, which describes relations between entities. Entities are represented by the tables or models in a database. Figure 6 illustrates the relations of each entity in the quiz web application database [8].
Figure 6. ER Diagram
CHAPTER FIVE:
SYSTEM ANALYSIS

Proposed System

The quiz web application has three roles: Admin, Student, and Instructor. Each role has specific permissions to access the platform. The admin role in the quiz web application includes managing various aspects of the educational institution, such as departments, courses, and instructors. The instructor organizes students, quizzes, and questions with their respective courses and departments. The students can see the upcoming quizzes, past quizzes, and scores of the past quizzes and take quizzes.

Web Application

Features for Admin

Since the admin's information already exists in the quiz web application database, I implemented a login component using the JWT authentication package to make sure that the admin can easily log in to the quiz web application. The admin will define Department (create department, edit department, view department, delete department), Course (add course, edit course, view course, delete course), and Instructor (add instructor, edit instructor, view instructor, delete instructor). Additionally, Admin can view the list of students with their respective courses and departments. Having this feature, the admin
can easily manage the various modules of the educational institution and ensure smooth functioning.

Features for Instructor

I used a login component for Instructors whose information is already in the quiz web application database. Here, the Instructor will define Student (add student, edit student, view student, delete student), Quiz (add quiz, edit quiz, view quiz, delete quiz), Question (add question, edit question, view question, delete question). Instructors can access the results of each quiz with student details.

Features for Student

I used a login component for Students whose information is already in the quiz web application database. Here, the Student will see the upcoming quizzes, view past quizzes with scores, and give the quiz response within the respective time limits of the quiz. One of the features of the quiz web application is that if a student changes their active tab while taking the quiz, the web application automatically redirects to the login page. It disqualifies the student from the current exam. Another feature of the quiz web application is that if students take a quiz and the time runs out, their responses will be automatically submitted. This eliminates the need for the student to worry about manually submitting their quiz.
CHAPTER SIX:
IMPLEMENTATION

Primary Implementation

Initially, I installed Node.js version v18.18.0 from the official website and set up the development environment for the project. Next, launch the command prompt and execute "npx create-react-app quiz_web" to install and set up React. After that, set up Apache, MySQL, and PHP components in Xampp installation.

To design the user interface, I created the components and stylesheets using HTML, SCSS, React Router Dom, Redux, Axios, and React Bootstrap packages to build business logic for the front end with a user-friendly interface.

Using Node.js and Express.js, I developed RESTful APIs for backend setup that could carry out a variety of operations, such as maintaining departments, courses, authentications, and additionally, while also defining their routes in the appropriate file. Additionally, I ensure that backend APIs handle authorization and authentication correctly. I established the connection between the backend server and database to carry out the CRUD (Create, Read, Update, and Delete) operations utilizing the Sequelize drivers in the quiz web application. (shown in Figure 10).
I utilized the "HTTP" module to send requests from the frontend to the backend APIs in order to establish a connection between the frontend and backend. The AXIOS package, which allows for the execution of asynchronous activities within API calls, is what I used to develop Restful API services.

In addition, to use the database, configure the Apache and MySQL servers in the Xampp program to enable localhost website development and testing.
Figure 8 demonstrates the overall database with tables and their respective attributes, constraints, and relations.
CHAPTER SEVEN:
USER INTERFACE

Login/Register

Open the quiz web portal in the web browser by navigating to “localhost:3000/register”. Figure 9 indicates the register page for the admin. I used the datta [9], a react template, to achieve an effective user interface.

![Admin Register Page]

Figure 9. Admin Register Page

Figure 10 indicates the login page for the admin, instructor, and student below, and Figure 11 shows the authentication failed error when a user enters the wrong email or password.
Figure 10. Login Page

Figure 11. Login Page (Authentication Fail)
Admin

Once the admin successfully logs in, they will be automatically directed to the Dashboard screen. The admin can access various options such as courses, departments, instructors, and students on this screen using the dynamic sidebar. For example, the admin can view and delete courses (as shown in Figure 13) or create and edit courses (as shown in Figure 14). Similarly, the admin can manage departments by viewing and deleting departments (as shown in Figure 15) or creating and editing departments (as shown in Figure 16). The admin can also check and delete the information of instructors (as shown in Figure 17) and create and edit instructors (as shown in Figure 18). Additionally, the admin can view all students (as shown in Figure 19) and log out from the quiz web application by clicking the logout button (as shown in Figure 20).

Figure 12. Admin Dashboard
Figure 13. View / Delete Courses

Figure 14. Add / Edit Courses
Figure 15. View/ Delete Department

Figure 16. Add/ Edit Department
Figure 17. View / Delete Instructor

Figure 18. Add/ Edit Instructor
Figure 19. View Students

Figure 20. Logout Component
Instructor

Once the instructor successfully logs in, they will be automatically directed to the Dashboard screen (as shown in Figure 21). The instructor can access various screens, such as students, quizzes, and questions, using the dynamic sidebar on this screen. The instructor can create and edit students (as shown in Figure 22) or view and delete students (as shown in Figure 23). Similarly, the instructor can manage quizzes by creating and editing quizzes (as shown in Figure 24), viewing past quizzes (as shown in Figure 25), seeing the student's results of each quiz (as shown in Figure 26), or checking the upcoming quizzes and delete quizzes (as shown in Figure 27). The instructor can also create and edit the information of each question (as shown in Figure 28), see the questions, or delete questions (as shown in Figure 29).

![Instructor Dashboard](image)

Figure 21. Instructor Dashboard
Figure 22. Add/ Edit Student

Figure 23. View / Delete Student
Figure 24. Add / Edit Quiz

Figure 25. View Past / Evaluate Quiz
Figure 26. View the Result of the Student

Figure 27. View / Delete Upcoming Quiz
Figure 28. Add / Edit Question

Figure 29. View/Delete Question
Once the student successfully logs in, they will be automatically directed to the Dashboard screen (as shown in Figure 30). Using the dynamic sidebar, students can access upcoming quizzes screens (as shown in Figure 31), past quizzes (as shown in Figure 32), and take quizzes (as shown in Figure 33). Figure 34 indicates the screen that appears after the student successfully submits the quiz.

Figure 30. Student Dashboard
Figure 31. View Upcoming Quiz

Figure 32. View Past Quiz
Figure 33. Take Quiz

Figure 34. After submitting the Quiz
The web application’s robust and compatible backend technology allows instructors to create and customize quizzes effectively. An automatic evaluation feature, which ensures accuracy in the evaluation process, is an essential benefit for instructors.

The quiz web application addresses cheating in online tests by implementing strategic measures such as time limitations and tracking the test screen, which indicates that students' knowledge can be evaluated fairly. With a responsive and user-friendly interface, students gain access to a convenient and accessible system.

The quiz web application is a valuable solution for remote learning and online education, breaking geographical barriers. By leveraging modern technology, instructors can engage students dynamically, shaping a more interactive and engaging future for education.
Currently, students can only see their results in the quiz web application. However, for future enhancement, it may be possible for students to see their quiz answers after the results are determined and get feedback from the instructor for their wrong answers, which might assist students in recognizing their areas of weakness and enhancing their knowledge of the course.

Additionally, the web application could be implemented using cloud services as a SAAS-based web portal.

Another potential enhancement for the quiz web application is implementing additional security measures to prevent cheating during online tests. One solution could be to track students' movements while taking the quiz and flag any suspicious behavior. This could help ensure the integrity of the assessment process and provide a more secure testing environment for students.
APPENDIX A:

BASIC CODE
Figure 35 indicates the Index.js file of a quiz web application, which is the root file in the component-based structure that allows all child components to execute with user-specific requirements.

![Figure 35. Frontend index.js](image)

Figure 36 indicates the server.js file of the quiz web application, which is the root file in the backend project, starts the server and establishes the connection with the database.

![Figure 36. Backend server.js](image)
REFERENCES


