

# **Department of Computer Science Bachelor of Science in Computer Science (BSCS)**

**🎓 Project Title:**

**CAMPUS QUIZZER**

**Course Title: Software Engineering  
Section: AM  
Semester: Spring 2024**

**Submitted By:  
• Dua Ashfaq (SP24-BSCS-0079)  
• Eshaal Atif (SP24-BSCS-0087)**

**Submitted To:  
Instructor Name: ASIA SHEIKH**

**📅 Submission Date: 27-May-2025**

# **INDEX**

1. **Introduction**
2. **Project Overview** **2.1 Problem Statement** **2.2 Aims and Objectives** **2.3 Inputs & Outputs**  **• Inputs**  **• Outputs** **2.4 Project Scope** **2.5 Domain**
3. **Software Process Model** **3.1 Process Model: Waterfall Model**  **• Communication**  **• Planning**  **• Modeling**  **• Construction**  **• Deployment** **3.2 Umbrella Activities**  **• Tracking Progress**  **• Handling Risks**  **• Ensuring Quality**  **• Reviewing Work**  **• Tracking Performance**  **• Managing Changes**  **• Reusing Features**  **• Preparing Documents**
4. **System Requirements** **4.1 Functional Requirements** **4.2 Non-Functional Requirements** **4.3 Hardware & Software Requirements**
5. **Wireframing of Project**
6. **Conclusion**
7. **References**

# 1. Introduction

Campus Quizzer is a simple, user-friendly online quiz system developed to assist teachers in creating, managing, and grading quizzes efficiently. It allows students to take quizzes online with instant evaluation and results, reducing manual effort and saving time for both educators and learners. The system is designed to streamline the quiz process and promote digital learning in academic institutions.

# 2. Project Overview

**2.1 Problem Statement**

In many educational institutions, quizzes are still conducted using traditional paper-based methods. These processes are time-consuming, prone to errors, and difficult to manage. There is a need for a streamlined online platform where teachers can create quizzes, students can attempt them easily, and results can be generated instantly and accurately.

**2.2 Aims and Objectives**

* Create a system that is simple and easy to use for both teachers and students
* Allow teachers to create, edit, and delete quizzes
* Automatically generate results and performance reports
* Provide a secure environment to prevent cheating during quizzes

**2.3 Inputs & Outputs**

**Inputs:**

* Teachers: Create quizzes by entering questions, answer choices, and correct answers
* Students: Select answers for each question and submit the quiz

**Outputs:**

* Students: Receive instant scores and feedback
* Teachers: Get access to student performance reports

**2.4 Project Scope**

* Intended primarily for universities and educational institutions
* Can be extended in the future to include timed quizzes, leaderboards, and quiz difficulty levels
* Helps students improve their knowledge through self-assessment
* Enhances learning experiences with automation and digital tools

**2.5 Domain**

* The system falls under software application development
* It is a web-based application accessible through modern internet browsers
* Technologies used include React.js (frontend), Node.js + Express (backend), and MySQL (database)
* Compatible with Windows, macOS, Android, and iOS devices

# 3. Software Process Model

**3.1 Process Model: Waterfall Model**

The project follows the Waterfall Model, a linear and sequential software development methodology. Each phase must be completed before moving to the next.

**Communication:**

* Requirement gathering for login, quiz creation, result generation, and reporting

**Planning:**

* UI design planning (buttons, quiz layout, result display)
* Structuring quiz flow (navigation, timer, submission)

**Modeling:**

* Designing pages (login page, quiz page, result page)
* Planning data structures for users, quizzes, and results

**Construction:**

* Coding modules for login, quiz attempts, evaluation, and results
* Testing all components for bugs and accuracy

**Deployment:**

* Launching the system for use
* Collecting feedback and applying necessary updates

**3.2 Umbrella Activities**

* ***Tracking Progress:*** Monitoring task completion and milestones
* ***Handling Risks:*** Preventing cheating, managing downtime or crashes
* ***Ensuring Quality:*** Performing multiple rounds of testing
* ***Reviewing Work:*** Verifying all functionality and UI elements
* ***Tracking Performance:*** Observing system response under different loads
* ***Managing Changes:*** Updating code without affecting existing features
* ***Reusing Features:*** Using reusable templates and components for login and quiz modules
* ***Preparing Documents:*** Writing clear user manuals and technical documentation

# 4. System Requirements

**4.1 Functional Requirements**

* Users must be able to register and log in
* Teachers should be able to create, update, and delete quizzes
* Students must be able to attempt quizzes and view results
* The system should automatically grade quizzes and generate performance reports

**4.2 Non-Functional Requirements**

* The platform should support multiple concurrent users
* Only registered users should be able to access quizzes
* The UI should be clean, responsive, and easy to navigate

**4.3 Hardware & Software Requirements**

* Frontend: React.js
* Backend: Node.js with Express
* Database: MySQL
* Hardware: Any device with 2GB+ RAM and a modern browser (Chrome, Firefox, Edge, etc.)
* Operating System: Windows, macOS, Linux, Android, or iOS

# 5. Wireframing of Project

Wireframes were created using **Canva** to plan the layout and user interface of the system before development. They include screens for login, registration, quiz creation, quiz attempt, and result display, serving as a design guide for the frontend.

**Wireframe Link:**  
<https://www.canva.com/design/DAGotrQ5VpQ/Gbaf-z3cMM7vTm40Ehrwog/edit?utm_content=DAGotrQ5VpQ&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton>

# 6. Conclusion

The *Campus Quizzer* system addresses the growing need for digital learning tools in educational institutions. It provides a streamlined, efficient way for teachers to manage quizzes and for students to participate in assessments anytime and anywhere. The system reduces manual workload, prevents cheating, and delivers instant feedback, making it a valuable asset for both educators and learners. Future versions of the project can incorporate more advanced features such as adaptive testing, quiz analytics, timed exams, and mobile app integration, enhancing usability and reach.

# 7. References

1. Sommerville, I. (2016). *Software Engineering* (10th ed.). Pearson Education.
2. Pressman, R. S., & Maxim, B. R. (2014). *Software Engineering: A Practitioner’s Approach* (8th ed.). McGraw-Hill Education.
3. Agarwal, B., Tayal, S. P., & Gupta, M. (2010). *Software Engineering & Testing*. Jones & Bartlett Learning.
4. Jalote, P. (2005). *An Integrated Approach to Software Engineering* (3rd ed.). Springer.