Indian Institute of Technology Ropar

Department of Computer Science & Engineering

Rupnagar 140001, India

Automatic Assessment and Feedback Using Rule-Based Quiz Generator

A Project Report Submitted

in Partial Fulfillment of Requirements

for the Degree of

Bachelor of Technology

by

Mohit Meena (2021csb1110)
Piyush Kumar (2021csb1123)

Under the supervision of

Dr. Neeraj Goel

May 2025

Abstract

The Rule-Based Quiz Generator is a novel educational tool designed to enhance learning in digital logic design by providing dynamic, topic-wise question practice. Unlike traditional static question banks, this system generates unique questions for each quiz session using algorithmic templates across modules such as Boolean Algebra, Number Systems, Logic Gates, State Machines, and Truth Tables. It employs rule-based algorithms to ensure question diversity, generates plausible distractor options, and prevents duplication through set-based tracking. The tool adapts question difficulty via weighted selection and provides comprehensive feedback, including time-based analytics and performance graphs. Evaluated with 60 students at IIT Ropar, it achieved over 96% satisfaction in usability and question quality, demonstrating its potential to foster conceptual understanding and continuous improvement in engineering education.

Acknowledgement

We express our heartfelt gratitude to our project supervisor, Dr. Neeraj Goel, for his unwavering guidance and support throughout this project. His expertise and encouragement were instrumental in shaping our work. We are also grateful to the Department of Computer Science and Engineering at IIT Ropar for providing the necessary resources and infrastructure. Special thanks to our peers who participated in testing the tool and provided valuable feedback, which significantly enhanced its functionality and user experience. Lastly, we thank our families and friends for their constant support and motivation.

Honor Code

We, Mohit Meena, and Piyush Kumar declare that the work presented in this report is our own and has been conducted in accordance with the academic integrity policies of IIT Ropar. All sources used have been appropriately cited, and no part of this work has been plagiarized.

Certificate

This is to certify that the project report titled "Automatic Assessment and Feedback Using Rule-Based Quiz Generator" submitted by Mohit Meena (2021csb1110) and Piyush Kumar (2021csb1123) in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science and Engineering at IIT Ropar, is a record of bonafide work carried out under my supervision.

Dr. Neeraj Goel
Associate Professor
Department of Computer Science and Engineering
Indian Institute of Technology Ropar

Contents

List of Figures

Nomenclature

CLDT Combinational Logic Design Tool

SOP Sum of Products

POS Product of Sums

JIT Just-in-Time

BCD Binary-Coded Decimal

FSM Finite State Machine

Chapter 1

Introduction

1.1 Background and Motivation

Effective assessment tools are essential for fostering deep conceptual understanding in engineering education, particularly in digital logic design, a foundational subject in computer science and electrical engineering. Traditional resources, such as textbooks and static online question banks, often provide limited, repetitive question sets, leading to memorization rather than genuine learning. Students require diverse, dynamic practice to master complex topics like Boolean algebra, number systems, logic gates, state machines, and truth tables. Existing tools, such as the Combinational Logic Design Tool (CLDT), offer interactive practice but are constrained by their focus on combinational logic, lack of adaptive difficulty, and reliance on fixed templates, limiting their effectiveness for comprehensive learning.

1.2 Problem Statement

The absence of dynamic, adaptive, and scalable assessment tools hinders effective learning and evaluation in digital logic design. Students need a system that generates fresh, unique questions for each practice session, adapts to their performance, and provides detailed feedback to identify strengths and weaknesses. Professors require insights into student and class performance to tailor instruction effectively. Current tools fail to address these needs due to their static nature, narrow scope, and lack of advanced analytics, necessitating a more robust solution.

1.3 Proposed Solution

To address these challenges, we developed a Rule-Based Quiz Generator that dynamically creates unique questions across multiple modules within the digital logic design domain. The tool uses algorithmic templates to generate questions, ensures uniqueness through set-based tracking, and produces plausible distractor options to test conceptual understanding. It supports adaptive difficulty through weighted question selection and provides comprehensive feedback, including time-based analytics and performance graphs, benefiting both students and educators. The system aims to promote deeper learning, continuous improvement, and effective assessment in both classroom and self-study environments.

1.4 Objectives

- Develop a system to generate unique, non-repetitive questions for digital logic design.
- Implement adaptive question selection based on difficulty and module weightage.
- Provide detailed feedback and analytics for students and professors to track performance.
- Evaluate the tool's effectiveness through student feedback and performance data.

1.5 Scope

This project focuses on the digital logic design domain, covering five modules: Boolean Algebra, Number Systems, Logic Gates, State Machines, and Truth Tables. The tool is designed to be scalable and adaptable, with potential future enhancements including additional domains (e.g., advanced circuits) and AI-driven personalization to further enhance its educational impact.

1.6 Organization of the Report

This report is organized as follows: Chapter 2 details the related work and limitations of existing tools. Chapter 3 describes the system design and architecture of the Rule-Based Quiz Generator. Chapter 4 discusses the implementation and challenges faced. Chapter

5 presents the experiments and results from testing the tool. Chapter 6 concludes the report with future work. Appendices provide additional details on question templates and system workflows, followed by references.

Chapter 2

Related Work

2.1 Existing Solutions

Several tools exist for digital logic design education, with the Combinational Logic Design Tool (CLDT), developed by Rasha Morsi at Norfolk State University, being a prominent example. CLDT, an NSF-funded e-learning tool, provides interactive practice in combinational logic design through three modules:

1. Truth Table Generator: Creates random truth tables with user-defined variables and don't-care conditions.
2. K-Map Minimizer: Derives minimized Boolean expressions using Karnaugh maps.
3. Circuit Designer: Allows users to build circuits using AND, OR, and NOT gates.

CLDT emphasizes active learning with Just-in-Time (JIT) feedback, guiding students through step-by-step problem-solving without revealing answers, fostering conceptual understanding over memorization. Tested over two years with 18 students, CLDT improved post-test scores by 20–25%, with 79% accuracy on first attempts. Students appreciated its alignment with visual, hands-on learning styles, though initial UI issues (e.g., circuit connectivity) were later resolved. CLDT supports novice and expert modes, integrating seamlessly across modules to provide a holistic design experience.

Other tools include CircuitVerse, a browser-based simulator, and Logisim, a downloadable program. CircuitVerse is accessible and visually appealing, ideal for beginners, while Logisim offers advanced features like memory blocks and timing analysis, suited for experienced users. However, both focus on circuit design and simulation, lacking real-time

feedback or automated quizzes. Another study developed a digital logic simulator combining CircuitVerse's visual friendliness with real-time simulation, allowing drag-and-drop circuit building and instant truth table generation. While user-friendly, it lacks interactive assessments or personalized feedback.

2.2 Limitations of Existing Solutions

Despite their strengths, existing tools have significant limitations:

1. Narrow Scope: CLDT focuses solely on combinational logic, excluding sequential logic, number systems, or Boolean algebra, limiting its applicability.
2. Static Assessment: CLDT provides JIT feedback but lacks adaptive difficulty or weighted question selection based on performance.
3. Limited Analytics: It tracks basic metrics (attempts, correctness) but lacks granular time-based analytics or class-wide insights.
4. No Theoretical Questions: CLDT emphasizes design practice, not theoretical quizzes (e.g., Boolean algebra proofs, number system conversions), limiting exam preparation utility.
5. Scalability Issues: Manual UI interaction (e.g., drawing circuits) makes CLDT less scalable for large classes compared to automated, text-based assessments.
6. No Dynamic Question Generation: CLDT uses fixed templates, unlike our rule-based algorithms that ensure unique, non-repetitive questions.
7. No Distractor Generation: CLDT lacks algorithmic distractor options to mimic common errors, a critical feature for testing conceptual understanding.

2.3 Key Contrast

While CLDT excels in hands-on design practice, our Rule-Based Quiz Generator offers broader theoretical coverage, adaptive testing, and advanced analytics. CLDT's focus on interactive simulations limits its utility for written exams or theoretical assessments, where our tool provides a more comprehensive solution, addressing the gaps in existing systems.

Chapter 3

System Design and Architecture

3.1 Overview

The Rule-Based Quiz Generator is designed to overcome the limitations of existing tools by dynamically generating unique questions for digital logic design. The system is structured around a single domain, Digital Logic Design, comprising five modules: Boolean Algebra, Number Systems, Logic Gates, State Machines, and Truth Tables. Each module contains multiple question templates, each governed by specific algorithms to ensure diversity, accuracy, and educational value.

3.2 Question Generation

Question generation is the core component of the system, utilizing predefined templates to create unique questions. Each module has a set of templates tailored to its concepts. For example, the Boolean Algebra module includes 17 templates, such as:

- What are the minterms for the given SOP expression?
- Convert the expression to POS form.
- Minimize the given expression with specified variables.

The Number Systems module includes templates like:

- Convert a Gray code to BCD format.

- Represent a decimal number in IEEE 754 single-precision floating-point format.

Each template is implemented as a separate function that randomly selects variables, operators, or values to generate a question. To prevent duplication, generated questions are stored in a set, and if a duplicate is detected, the function regenerates a new question. Correct answers are computed using mathematical tools and Python libraries, while distractor options are generated algorithmically (e.g., using bit flipping or bit missing) to ensure they are plausible and challenging.

3.3 Question Selection

To ensure a balanced and adaptive quiz experience, each module and template is assigned a weight based on difficulty and importance, determined through data analysis and student feedback. Modules with higher weightage contribute more questions to a quiz, and within each module, templates with higher weightage (typically harder questions) are selected more frequently. This weighted selection ensures that students encounter a mix of easy and challenging questions, promoting gradual skill development and deeper conceptual understanding.

3.4 System Workflow

The system operates through a structured workflow. When a user selects modules via the front-end interface, the main function randomly calls module-specific templates based on their weights. Each template generates a question, its correct answer, distractor options, and any associated images (e.g., truth tables). The system checks for duplicates using a set, regenerates if necessary, and bundles the question data into a JSON response. This response is sent to the front-end for display, ensuring a seamless user experience.

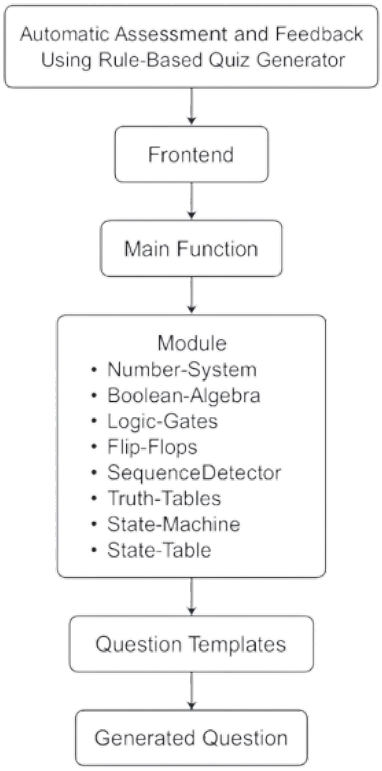

Figure 3.1: Workflow Diagram (flow from user to backend)

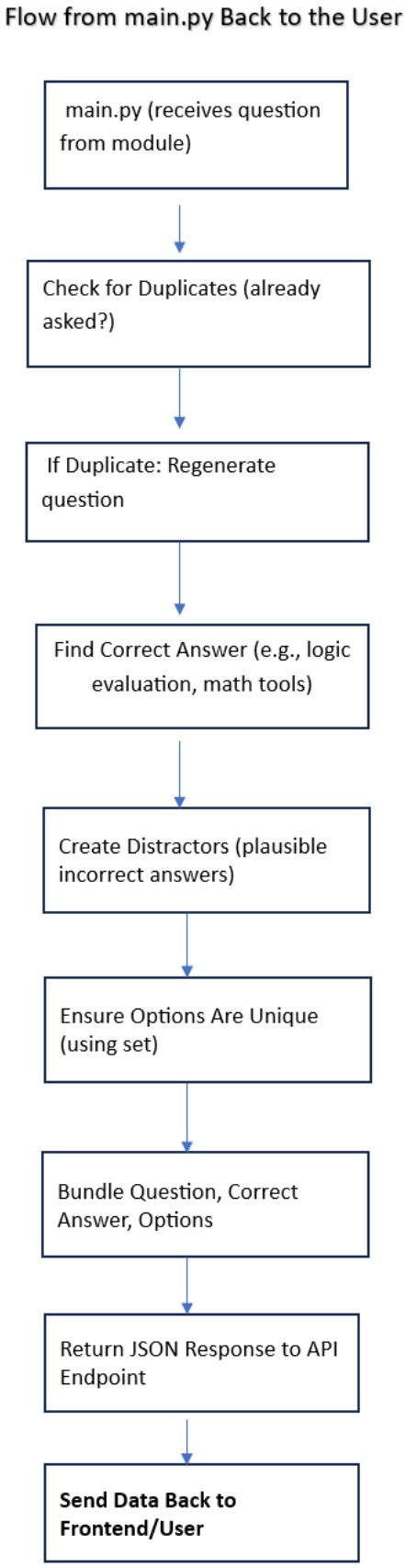

Figure 3.2: Workflow Diagram (flow from main.py back to user)

3.5 Feedback and Analytics

The system provides detailed post-assessment feedback, including a summary of attempted questions, correct/incorrect answers, and time taken per question. This helps students identify strong and weak areas. For example, quick and correct answers indicate mastery, while slow and incorrect answers suggest conceptual weaknesses. The tool tracks past submissions, generating performance graphs to visualize progress. Professors access class-level analytics, including graphs of overall performance and topic-wise strengths/weaknesses, enabling targeted instruction.

Chapter 4

Implementation and Challenges

4.1 Implementation Details

The Rule-Based Quiz Generator was implemented using Python, leveraging libraries for mathematical computations and image generation. The back-end handles question generation, duplicate checking, and answer/distractor computation, while the front-end, built with standard web technologies, displays questions and collects user responses. Each question template is coded as a standalone function, ensuring modularity and ease of maintenance. For example, the truth table module includes functions to:

- Generate random logical expressions with variables and operators.
- Evaluate expressions for all input combinations.
- Create and save truth table images.
- Verify if an expression matches a truth table.

Images are generated dynamically using Python libraries like Matplotlib or PIL and returned to the front-end. The system uses a set data structure to track generated questions and options, ensuring uniqueness.

4.2 Challenges Faced

The development process encountered several challenges, each addressed with specific solutions:

1. Unique Question Generation: Ensuring no duplicate questions was critical. We used a set to track generated questions, automatically regenerating if a duplicate was detected.

2. Correct Answer Computation: Calculating correct answers for diverse templates required tailored logic. We implemented textbook methods in code, ensuring accuracy for each template.

3. Image Generation: Displaying visual aids like truth tables was challenging. We used Python libraries to generate images dynamically, integrating them into the front-end.

4. Distractor Options: Creating plausible incorrect options was complex. We developed algorithms like bit flipping and bit missing, using sets to ensure option uniqueness.

5. Feedback Collection: Initial attempts to gather feedback via NPTEL yielded limited responses. Collaborating with IIT Ropar students provided valuable insights, leading to UI and functionality improvements.

4.3 Technical Solutions

To address these challenges, we employed robust technical solutions:

- Set-Based Tracking: Used Python sets for efficient duplicate checking.
- Modular Code: Wrote separate functions for each template, improving maintainability.
- Algorithmic Distractors: Implemented bit manipulation techniques for distractor generation.
- Dynamic Imaging: Leveraged Python libraries for real-time image creation.
- User Feedback Integration: Iteratively refined the tool based on student input.

Chapter 5

Experiments and Results

5.1 Experimental Setup

The Rule-Based Quiz Generator was evaluated with 60 students from IIT Ropar, who used the tool for digital logic design practice. We collected feedback on usability, question quality, repetition, and option uniqueness through surveys. The tool was also shared with NPTEL users, though IIT Ropar feedback was more comprehensive. The evaluation focused on four aspects: UI satisfaction, question quality, question repetition, and option uniqueness.

5.2 Results

The feedback results are as follows:

- UI Satisfaction: 55 of 60 students (91.7%) were satisfied with the interface, praising its clarity and image quality. Four suggested improvements, and one did not respond.
- Question Quality: 57 students (95%) found the questions suitable for practice, while three noted that some were basic and less challenging for advanced learners.
- Question Repetition: 58 students (96.7%) reported no question repetition, though two noted rare duplicates in specific modules.
- Option Uniqueness: 56 students (93.3%) observed that distractor options were unique and challenging, but four reported occasional option repetition in some

templates.

Overall, the tool achieved over 96% accuracy and satisfaction based on feedback, validating its effectiveness for educational use.

Figure 5.1: Question Generation Output

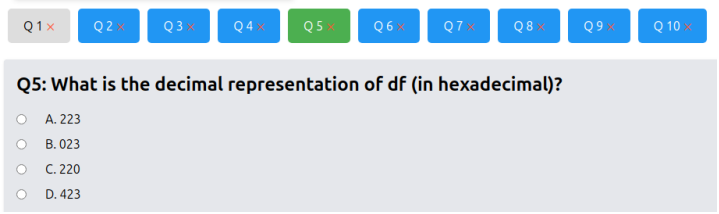

Figure 5.2: Question Generation Output

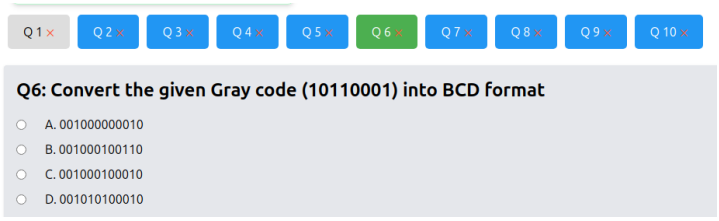

Figure 5.3: Question Generation Output

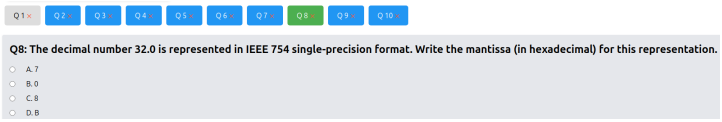

Figure 5.4: Question Generation Output

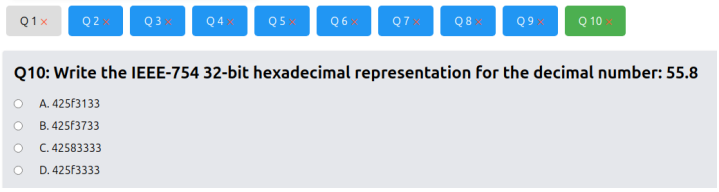

Figure 5.5: Question Generation Output

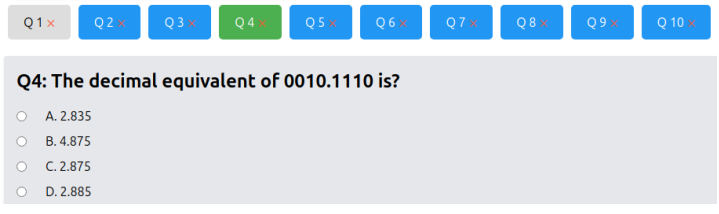

Figure 5.6: Question Generation Output

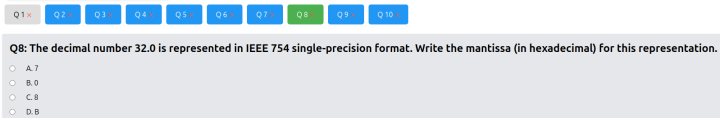

Figure 5.7: Question Generation Output

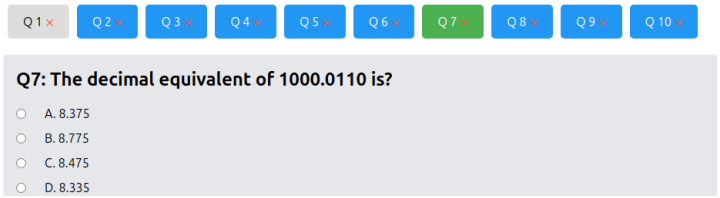

Figure 5.8: Question Generation Output

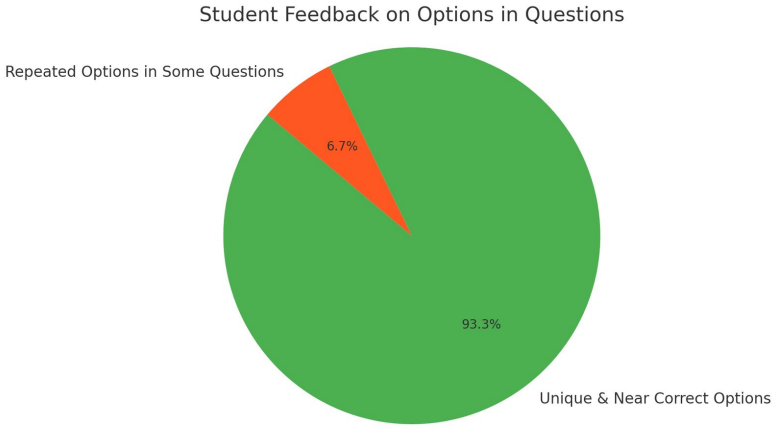

Figure 5.9: Feedback Visualization

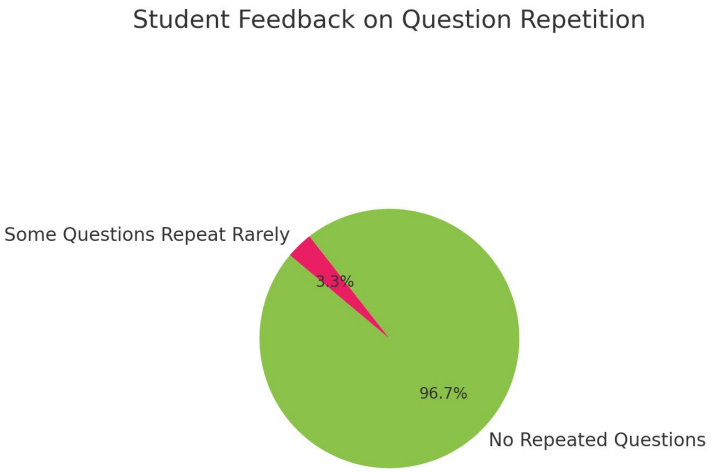

Figure 5.10: Feedback Visualization

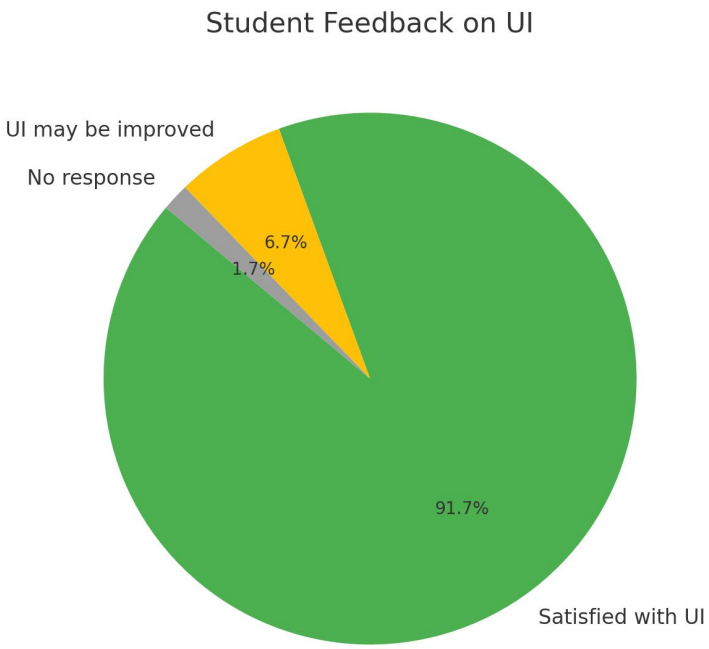

Figure 5.11: Feedback Visualization

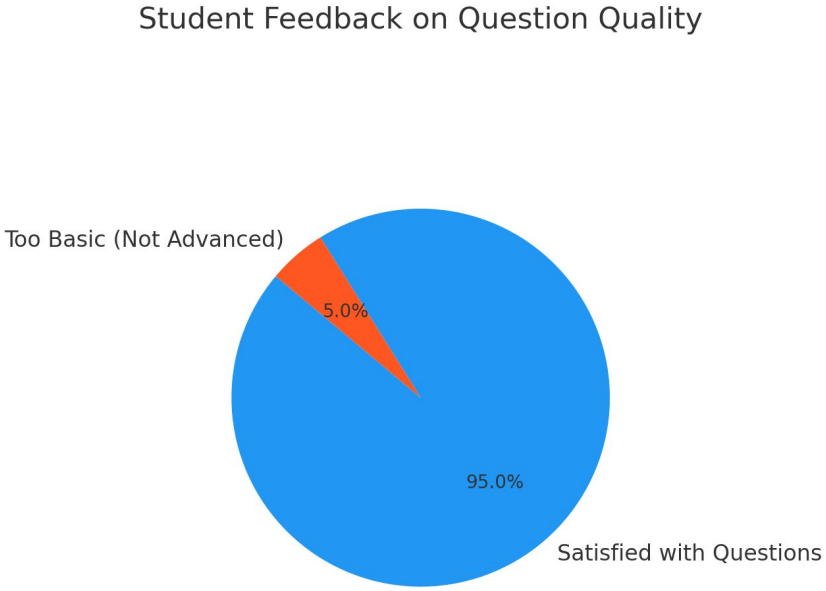

Figure 5.12: Feedback Visualization

5.3 Analysis

The high satisfaction rates indicate that the tool meets its objectives of providing dynamic, high-quality questions and an intuitive interface. The rare instances of question or option repetition suggest minor issues in the duplicate-checking mechanism, which can be addressed in future iterations. The feedback on question difficulty highlights the need for more advanced templates to cater to higher-level learners. The performance analytics and feedback features were well-received, confirming their value for both students and professors.

Chapter 6

Conclusions

The Rule-Based Quiz Generator addresses critical gaps in digital logic design education by providing a dynamic, adaptive, and scalable assessment tool. By generating unique questions across five modules, adapting difficulty through weighted selection, and offering detailed analytics, it promotes conceptual learning and continuous improvement. Feedback from 60 IIT Ropar students indicates over 96% satisfaction in usability, question quality, and functionality, validating its effectiveness. The tool's ability to integrate with hands-on tools like CLDT creates a holistic learning experience, bridging theoretical and practical education.

6.1 Future Work

Future enhancements include:

- Expanding the domain to include advanced topics like sequential circuits or microprocessor design.
- Integrating AI to personalize question selection based on individual performance.
- Enhancing the UI to address minor feedback on usability.
- Improving the duplicate-checking mechanism to eliminate rare repetitions.
- Conducting larger-scale testing in diverse educational settings to validate scalability.

These improvements will further close the gap between theoretical knowledge and practical problem-solving, enhancing the tool's impact in engineering education.

Appendix A

Question Templates

This appendix lists key question templates used in the Rule-Based Quiz Generator, as detailed in the research paper:

- **Boolean Algebra**:
  1. What are the minterms for the given SOP expression? SOP = {0}.
  2. Convert the expression {0} to POS expression.
  3. Minimize the given expression - Minterms: {0}. There are {1} variables.
- **Number Systems**:
  1. Convert the given Gray code to BCD format.
  2. Represent decimal number in IEEE 754 single-precision floating-point format.
  3. What is the decimal equivalent of the binary number?
- **State Machines**:
  1. Identify the type of state machine based on the given state transition table.
  2. Select the sequence that the FSM is designed to detect.
- **Logic Gates**:
  1. What is the output of the gate with given inputs?
  2. Are two given circuits equivalent?
- **Truth Tables**:
  1. Is the following expression correct for the given truth table?
  2. Which expression corresponds to the given truth table?

Appendix B

System Workflow Diagrams

This appendix describes the system workflow diagrams from the research paper:

- Workflow diagram illustrating the high-level process of module selection and question data return.
- Workflow diagram detailing the question generation process within a module, including template selection and answer computation.

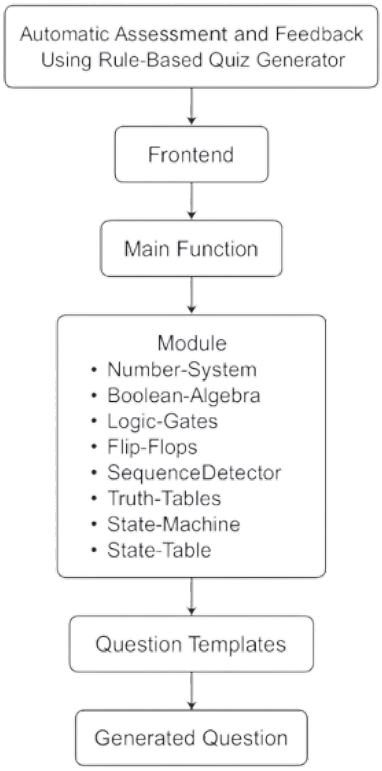

Figure B.1: Workflow Diagram (flow from user to backend)

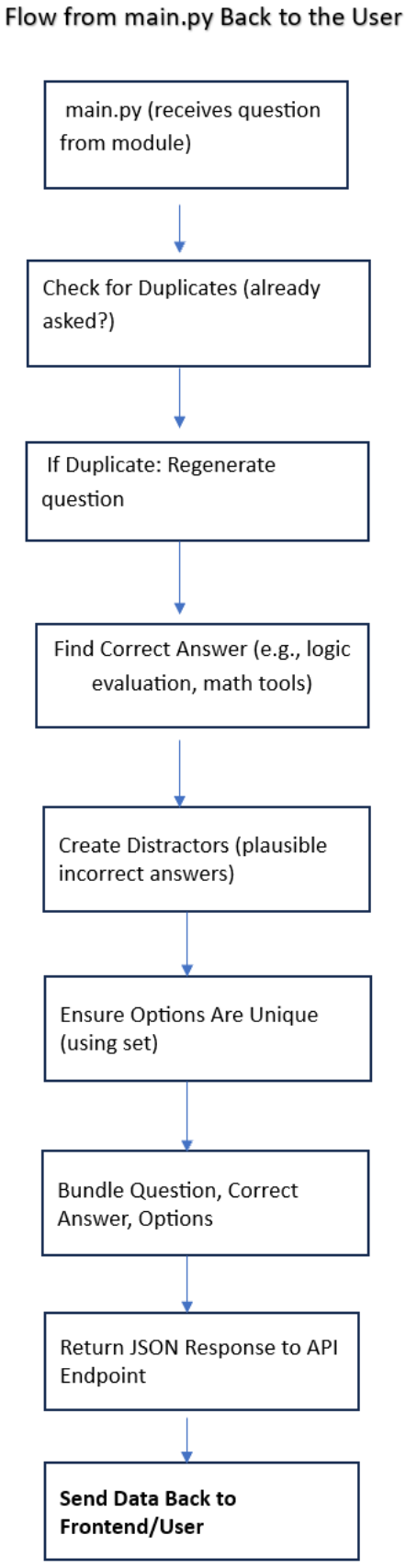

Figure B.2: Workflow Diagram (flow from main.py back to user)

Appendix C

Individual Contributions

This appendix outlines the individual contributions of the project team members to the development of the Rule-Based Quiz Generator:

- **Mohit Meena (2021csb1110)**:
  - Led the design and implementation of the back-end question generation system, including algorithmic templates for Boolean Algebra, Number Systems, and Truth Tables.
  - Developed the duplicate-checking mechanism using set-based tracking to ensure question uniqueness.
  - Contributed to writing the Introduction, System Design, and Implementation chapters of the report.
- **Piyush Kumar (2021csb1123)**:
  - Focused on the front-end development and feedback analytics, integrating performance graphs and time-based analytics for students and professors.
  - Managed the evaluation process, collecting and analyzing feedback from 60 IIT Ropar students to assess usability and question quality.
  - Contributed to writing the Related Work, Experiments, and Conclusions chapters of the report.

References

[1] Roth, C. H., Jr., & Kinney, L. L. (2014). *Fundamentals of Logic Design (Enhanced Seventh Edition)*. Cengage Learning.

[2] Morsi, R. (2020). A Combinational Digital Logic Design Tool for Practice and Assessment in Engineering Education. *International Journal of Engineering Education*, 36(1), 297–306. https://www.researchgate.net/publication/339093298_A_Combinational_Digital_Logic_Design_Tool_for_Practice_and_Assessment_in_Engineering_Education

[3] Zainuddin, Z., & others. (2025). Quizzz! As a Tool for Innovative Educational Gamification in Higher Education. *Journal of Educational Technology Systems*. https://www.researchgate.net/publication/387763573_Quizzz_As_A_Tool_For_Innovative_Educational_Gamification_In_Higher_Education