MINI PROJECT REPORT

On

ATS – AUTOMATED TEST EVALUATION

Submitted in partial fulfilment for the award of degree

of

Masters of Computer Applications

By

DEVANARAYANAN M

MLM23MCA-2023

Under the Guidance of

MS. BANU SUMAYYA S

(Assistant Professor, Department of Computer Applications)

DEPARTMENT OF COMPUTER APPLICATIONS MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR

(Affiliated to APJ Abdul Kalam Technological University)

NOVEMBER 2024

MANGALAM COLLEGE OF ENGINEERING

Accredited by NAAC& ISO 9001:2000 Certified Institution

DEPARTMENT OF COMPUTER APPLICATIONS

VISION

To become a centre of excellence in computer applications,competent in the global ecosystem with technical knowledge,innovation with a sense of social commitment.

MISSION

- To serve with state of the art ducation,foster advanced research and cultivate innovation in the field of computer applications.
- To prepare learners with knowledge skills and critical thinking to excel in the technological landscape and contribute positively to society.

Program Educational Objectives

- PEO I :Graduates will possess a solid foundation and in-depth understanding of computer applications and will be equipped to analyze real-world problems, design and create innovative solutions, and effectively manage and maintain these solutions in their professional careers.
- PEO II: Graduates will acquire technological advancements through continued education,s lifelong learning and research, thereby making meaningful contributionto the field of computing.
- PEO III: Graduates will cultivate team spirit, leadership, communication skills, ethics, and social values, enabling them to apply their understanding of the societal impacts of computer applications effectively.

Program Specific Outcomes

- **PSO I**: Apply advanced technologies through innovations to enhance the efficiency of design development.
- **PSO II**: Apply the principles of computing to analyze, design and implement sustainable solutions for real world challenges.

MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR

DEPARTMENT OF COMPUTER APPLICATIONS

NOVEMBER 2024

CERTIFICATE

This is to certify that the Project titled ***"ATS – Automated Test Evaluation"*** *is the bonafide record of the work done by* ***DEVANARAYANAN M (MLM23MCA-2023****) of Masters of Computer Applications towards the partial fulfilment of the requirement for the award of the* ***DEGREE OF MASTERS OF COMPUTER APPLICATIONS by APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY****, during the academic year 2023-24.*

Internal Examiner **External Examiner**

Project Guide
Ms. Banu Sumayya S
Assistant Professor
DCA

Head of the Department
Ms. Divya S.B
Associate Professor
DCA

ACKNOWLEDGEMENT

I am greatly in debted to the authorities of Mangalam College of Engineering for providing the necessary facilities to successfully complete my Project on the topic “ATS – AUTOMATED TEST EVALUATION”.

I express my sincere thanks to **Dr. Vinodh P Vijayan**, Principal, Mangalam College of Engineering for providing the facilities to complete my Project successfully.

I thank and express my solicit gratitude to **Ms. Divya S.B. ,**HOD, Department of Computer Applications, Mangalam College of Engineering for there invaluable help and support which helped me a lot in successfully completing this Project work.

I express my gratitude to my Project Co-ordinator, **Ms. Banu Sumayya S**, Assistant professor, Department of Computer Applications for the suggestions and encouragement which helped in the successful completion of our Project.

I express my gratitude to my Internal Guide, **Ms. Banu Sumayya S**, Assistant professor, Department of Computer Applications for the suggestions and encouragement which helped in the successful completion of our Project. Finally,I would like to express my heartfelt thanks to my parents who were very supportive mentally and for their encouragement to achieve my goal.

DEVANARAYANAN M
(MLM23MCA-2023)

ABSTRACT

This project presents an online platform designed to automatically evaluate descriptive answers submitted in online assessments. The system assesses answer responses submitted by students and awards marks based on the expected answer provided by faculty members for each question.

The automated test scrutiny is achieved using Natural Language Processing (NLP), including techniques such as tokenization and cosine similarity, implemented in Python. The front- end of the system is developed using HTML, CSS, PHP and other web technologies. The system also utilizes a MySQL database for data storage and management.

Faculty members can create questions and specify the expected answers. These ques- tions can be used to construct tests. Students can take these tests and submit their answers. The system aims to streamline the assessment process, reducing the burden on educators and ensuring consistent and fair evaluations for students.

Key features of the system include question management, automatic answer evaluation and test creation. It offers an efficient solution to the challenges of evaluating descriptive answers in online assessments. Additionally, it provides a platform for managing a repository of questions for educational purposes and generating custom question papers.

TABLE OF CONTENTS

List of Figures

LIST OF ABBREVIATIONS

ABBREVIATION		FULL FORM
DFD	-	Data Flow Diagram
UI	-	User Interface
PHP	-	Hypertext Preprocessor
HTML	-	Hyper Text Markup Language
CSS	-	Cascading Style Sheets
OS	-	Operating System
RAM	-	Random Access Memory
NLP	-	Natural Language Processing

CHAPTER 1

INTRODUCTION

1.1 Background

In the digital age, online assessments have become an integral part of educational systems, facilitating flexible and accessible learning environments. However, the evaluation of descriptive answers presents a unique challenge for educators. Traditional grading methods can be time-consuming and subjective, often leading to inconsistencies in marking. To address these issues, our project introduces an innovative online platform that automates the evaluation of descriptive answers submitted in online assessments. This system leverages advanced Natural Language Processing (NLP) techniques, including tokenization and cosine similarity, to assess student responses against faculty-provided expected answers. By employing these methods, the platform not only enhances the accuracy of evaluations but also significantly reduces the workload for educators, allowing them to focus more on teaching and mentoring. Key features of the platform include an intuitive question management system, automatic answer evaluation, and the ability to generate custom question papers. By creating a centralized repository of questions, educators can streamline the test creation process, ensuring that assessments are both relevant and varied. This platform not only aims to enhance the fairness and consistency of evaluations but also addresses the broader challenges of managing online assessments effectively. Ultimately, this project represents a significant step towards modernizing educational assessments, providing an efficient and scalable solution that benefits both educators and students in the evolving landscape of digital education.

1.2 Introduction

As educational institutions increasingly transition to online learning environments, the assessment of student performance has become a critical focus. Traditional methods of evaluating descriptive answers can be labor-intensive and often introduce inconsistencies that undermine the reliability of grades. To tackle these challenges, our project proposes an automated online platform designed to evaluate descriptive answers submitted during assessments effectively. This innovative system harnesses the power of Natural Language Processing (NLP) to provide objective and consistent

evaluations. By employing techniques such as tokenization and cosine similarity, the platform can analyze student responses against predefined expected answers, offering a more streamlined approach to grading. This not only reduces the workload on educators but also enhances the overall quality of assessments, ensuring that students receive fair and accurate evaluations. The technical backbone of the platform is robust, with a front-end developed using HTML, CSS, and PHP, complemented by a MySQL database for efficient data management. Faculty members can easily create and manage questions, define expected answers, and construct custom tests tailored to their curriculum. The system also maintains a repository of questions, allowing for efficient test creation and management. This project addresses the pressing need for efficient assessment tools in online education. By automating the evaluation of descriptive answers, the platform not only alleviates the grading burden on educators but also fosters a more equitable assessment environment for students. Through this innovative solution, we aim to contribute to the advancement of educational practices in an increasingly digital world.

1.3 **Problem Statement**

The rapid shift to online education has highlighted significant challenges in the assessment of student performance, particularly concerning descriptive answers in exams and assignments. Traditional grading methods are often time-consuming, subjective, and prone to inconsistencies, leading to potential biases and discrepancies in student evaluations. Evaluating thousands plus words essays manually is time-consuming. Educators frequently find themselves overwhelmed by the volume of assessments, resulting in delayed feedback for students, which can hinder their learning progress. Moreover, the lack of standardized evaluation processes can lead to variations in grading criteria, making it difficult to ensure fairness and accuracy. This situation not only burdens educators but also undermines the integrity of the assessment process, impacting student motivation and trust in the educational system. To address these challenges, there is a critical need for an automated solution that can efficiently and objectively evaluate descriptive answers, streamline the grading process, and provide timely feedback. This project aims to develop an online platform that utilizes Natural Language Processing (NLP) techniques to assess student responses against predefined expected answers, thereby enhancing the consistency, reliability, and overall effectiveness of online assessments.

1.4 Motivation

The increasing adoption of online education has transformed the landscape of learning and assessment, bringing both opportunities and challenges. As educators strive to provide quality education in a digital environment, the need for efficient and reliable assessment tools has become paramount. This project is motivated by the desire to improve the grading process for descriptive answers, which is often the most complex and time-consuming aspect of assessments. With the rise of online assessments, educators face the dual challenge of managing large volumes of submissions while ensuring fair and accurate evaluations. The potential for subjective biases in grading, coupled with the demand for timely feedback, can lead to significant stress for educators and confusion for students. Our motivation stems from the belief that an automated system can alleviate this burden, fostering a more equitable assessment environment. By harnessing the capabilities of Natural Language Processing (NLP), we aim to create a solution that not only streamlines the evaluation process but also enhances the learning experience for students. Providing timely, consistent, and objective feedback is essential for fostering student growth and motivation. Ultimately, this project seeks to contribute to the ongoing evolution of educational practices, ensuring that assessments are both effective and supportive of student learning in an increasingly digital world.

1.5 Scope

The Automated test scrutiny system goes beyond traditional assessment methods, providing a versatile solution applicable to various subject areas. This inclusive approach ensures educators can use the system across diverse academic domains. The innovative question bank further expands the project's scope, empowering educators to create, organize and export questions universally. By transcending subject-specific boundaries, this feature enriches educational resources, allowing educators to align assessments seamlessly with diverse curriculum. Bringing the segment to its conclusion by outlining the vast scope of the automated test scrutiny system, providing a versatile solution applicable across diverse academic domains. The innovative question bank enriches educational resources, empowering educators to create, organize and export questions universally. As the chapter closes, the stage is set for an exploration of the system's components and its transformative impact on education and assessment.

CHAPTER 2
LITERATURE REVIEW

[1]Recent advancements and challenges of NLP-based sentiment analysis: A state-of-the-art review by Jamin Rahman Jim , Md Apon Riaz Talukder , Partha Malakar (2023)

Sentiment analysis is a method within natural language processing that evaluates and identifies the emotional tone or mood conveyed in textual data. Scrutinizing words and phrases categorizes them into positive, negative, or neutral sentiments. The significance of sentiment analysis lies in its capacity to derive valuable insights from extensive textual data, empowering businesses to grasp customer sentiments, make informed choices, and enhance their offerings. For the further advancement of sentiment analysis, gaining a deep understanding of its algorithms, applications, current performance, and challenges is imperative. Therefore, in this extensive survey, we began exploring the vast array of application domains for sentiment analysis, scrutinizing them within the context of existing research. We then delved into prevalent pre-processing techniques, datasets, and evaluation metrics to enhance comprehension. We also explored Machine Learning, Deep Learning, Large Language Models and Pre-trained models in sentiment analysis, providing insights into their advantages and drawbacks. Subsequently, we precisely reviewed the experimental results and limitations of recent state-ofthe-art articles. Finally, we discussed the diverse challenges encountered in sentiment analysis and proposed future research directions to mitigate these concerns. This extensive review provides a complete understanding of sentiment analysis, covering its models, application domains, results analysis, challenges, and research directions.

The paper "Recent Advancements and Challenges of NLP-Based Sentiment Analysis: A State-of-the-Art Review" by Zhang et al. (2021) provides a comprehensive overview of the evolution and current state of sentiment analysis within the realm of natural language processing (NLP). Sentiment analysis, the computational task of identifying and categorizing opinions expressed in text, has gained significant attention due to its applications in various fields, including marketing, social media monitoring, and customer feedback analysis.The authors begin by tracing the historical development of sentiment analysis techniques, highlighting the transition from traditional methods based on lexical and rule-based approaches to modern machine learning and deep learning methods. Early sentiment analysis relied heavily on keyword matching and

predefined lexicons, which were limited in their ability to capture the nuances of human language. However, with advancements in NLP, especially the emergence of deep learning, researchers have been able to leverage more sophisticated models such as Long Short-Term Memory (LSTM) networks and Transformer-based architectures like BERT (Bidirectional Encoder Representations from Transformers). These models have demonstrated significant improvements in accuracy and performance, largely due to their ability to understand context, handle complex sentence structures, and utilize large datasets.The review emphasizes the role of pre-trained models and transfer learning in advancing sentiment analysis. Pre-trained models allow researchers to fine-tune existing architectures on specific datasets, significantly reducing the amount of labeled data needed for training. This approach has proven effective in various applications, enabling the adaptation of models to specific domains such as finance or healthcare, where language usage can vary considerably.Despite these advancements, Zhang et al. discuss several ongoing challenges in the field. One major issue is the inherent complexity of human language, which is often context-dependent and influenced by factors such as sarcasm, idiomatic expressions, and cultural nuances. These challenges make it difficult for sentiment analysis models to achieve high accuracy across diverse datasets. Moreover, the authors highlight the importance of domain adaptation, noting that models trained on general datasets may struggle when applied to domain-specific data. This limitation underscores the need for more robust training methodologies that can enhance model generalization.

Ethical considerations also emerge as a critical topic in the review. The authors point out that biases present in training data can lead to biased outcomes in sentiment analysis applications. For instance, if a model is trained predominantly on text from one demographic group, it may not perform well when analyzing text from other groups, leading to skewed interpretations of sentiment. The authors call for more transparency in the development and deployment of sentiment analysis systems to mitigate these biases and ensure fairness in applications.Looking toward the future, the paper advocates for research that integrates multimodal data sources—such as audio and visual inputs—alongside text, to create a more holistic understanding of sentiment. Additionally, the authors suggest that enhancing contextual understanding and developing frameworks that can dynamically adapt to new language trends and expressions will be crucial in overcoming existing limitations.

In conclusion, Zhang et al. provide a thorough review of the advancements and challenges in NLP-based sentiment analysis, underscoring the progress made through deep learning techniques while acknowledging the complexities that remain. The paper serves as a valuable resource for researchers and practitioners alike, highlighting both the potential and the pitfalls of current sentiment analysis methodologies, and pointing toward future directions that could enhance the efficacy and fairness of these systems in practical applications.

[6] NLP-based Automatic Answer Script Evaluation by Rahman, Md & Siddiqui, Fazlul (2018)

Rahman and Siddiqui's paper NLP-based Automatic Answer Script Evaluation (2018) explores the application of Natural Language Processing (NLP) techniques to the automated grading of written examination scripts. This work addresses the growing need for efficient, accurate, and scalable methods for evaluating large volumes of answer scripts, particularly in educational settings where manual grading is time-consuming and subject to human bias. The authors present an approach that leverages NLP algorithms to automatically evaluate the content and quality of students' written answers. This method aims to analyze the semantics and structure of student responses, enabling the system to understand not just keywords but the overall meaning of an answer. Rahman and Siddiqui emphasize the importance of going beyond simple keyword matching, which is commonly used in traditional automated grading systems, and instead focus on the evaluation of meaning and concept comprehension. A key aspect of the proposed system is its ability to handle subjective answers, which are often difficult to grade accurately with automated systems. The authors utilize techniques such as tokenization, part-of-speech tagging, and semantic analysis to break down and understand the natural language used in answers. By applying these NLP methods, the system is designed to assess whether the student's response adequately covers the key points expected in the answer. Rahman and Siddiqui also highlight the challenge of evaluating grammatical and syntactical correctness, which is an important aspect of answer quality. Their system integrates these aspects into the overall grading process, providing a more holistic evaluation compared to systems that only focus on content. The authors note that NLP-based systems can be trained on sample answers to improve their accuracy over time, enabling the model to better understand the nuances of various types of responses. The study demonstrates the potential of automated systems to reduce grading time and improve consistency in evaluations. By eliminating human errors and biases in the grading process, NLP-based evaluation can contribute

to more objective assessments. However, Rahman and Siddiqui acknowledge that fully automating the evaluation of complex, subjective answers remains a challenging task. They propose further research to improve the system's ability to handle diverse question types and answer formats. In conclusion, Rahman and Siddiqui's (2018) research contributes to the growing body of literature on the use of NLP in education, particularly in the area of automated assessment. Their system represents an innovative step forward in the development of tools for automatic grading, particularly in the evaluation of subjective and complex answers, though further refinement and testing are needed to improve its accuracy and adaptability to different contexts.

[7] An Intelligent System for Evaluation of Descriptive Answers" by Bagaria, Vinal, Badve, Mohit, Beldar, Manasi, and Ghane, Sunil (2020)

Bagaria et al.'s paper An Intelligent System for Evaluation of Descriptive Answers (2020) addresses the issue of automating the evaluation of descriptive answers in educational assessments. The authors propose an intelligent system that utilizes advanced computational techniques to analyze and score long-form, descriptive student responses, which are typically challenging to grade using traditional automated systems. The paper highlights the limitations of conventional automated grading systems, which often struggle to assess descriptive answers that require understanding of content, coherence, and depth of explanation. Bagaria et al. suggest that their intelligent system bridges this gap by employing Natural Language Processing (NLP) techniques and machine learning algorithms to evaluate descriptive answers more effectively. A key feature of the proposed system is its ability to analyze both the content and structure of answers. The system is designed to match student responses against a model answer, focusing on important keywords, phrases, and concepts. However, instead of relying solely on keyword matching, the system incorporates semantic analysis, allowing it to assess the meaning and relevance of the student's answer in relation to the expected response. This helps in grading answers that may be worded differently but convey the same meaning. The authors discuss the implementation of machine learning models that are trained on datasets of evaluated answers. These models learn to recognize patterns in responses that are associated with higher or lower scores. By continuously learning from new data, the system can improve its grading accuracy over time, making it adaptable to different subjects and question types. Another notable aspect of Bagaria et al.'s intelligent system is its ability to handle grammatical and syntactical variations. The system

evaluates the language quality of the answers, including sentence structure, grammar, and coherence, which are important factors in assessing descriptive answers. This feature helps ensure that the grading is more comprehensive, taking into account both content accuracy and language proficiency. The authors conducted experiments to evaluate the system's performance and found that it achieved a high degree of accuracy compared to human graders. They conclude that the intelligent system offers a promising solution for automating the grading of descriptive answers, reducing the workload on teachers while maintaining consistent and objective evaluations. However, Bagaria et al. also acknowledge certain limitations in their system, such as the challenge of grading highly subjective or creative responses. They suggest that further improvements could be made by refining the NLP models and incorporating more advanced semantic analysis techniques to better handle diverse types of descriptive answers. In conclusion, Bagaria et al.'s (2020) work contributes significantly to the field of automated grading, particularly for descriptive answers. Their intelligent system leverages NLP and machine learning to offer a more sophisticated approach to grading long-form responses, balancing the need for content accuracy, semantic understanding, and language quality. While there are areas for future research and improvement, the study provides a strong foundation for developing more effective automated assessment tools in educational settings.

[8]MF Bashir, Hamza Arshad ,*Subjective Answers Evaluation Using Machine Learning and Natural Language Processing* (2021)

The paper by Bashir et al. (2021) addresses the challenge of evaluating subjective answers, which is traditionally done manually, making it time-consuming and prone to inconsistencies. The study explores the use of machine learning (ML) and natural language processing (NLP) as potential tools to automate this evaluation process, ensuring more efficient and consistent grading. Subjective answers are inherently difficult to grade due to their unstructured nature and variability, and traditional methods often introduce human biases, leading to inconsistent outcomes. The authors propose the use of supervised and unsupervised machine learning models that can be trained to evaluate subjective answers based on content rather than relying solely on keywords. NLP techniques, such as semantic analysis and text similarity measures, are highlighted as essential components in understanding and assessing the meaning and context of the responses. These methods enable the system to evaluate answers syntactically and semantically, resulting in

more accurate evaluations compared to traditional keyword-matching approaches. One of the key challenges discussed in the paper is the need for large datasets of subjective answers to train the models effectively. Despite this difficulty, the experimental results show that the proposed approach, combining machine learning and NLP, achieves a high level of accuracy in evaluating subjective responses. The authors emphasize that this system outperforms traditional evaluation methods and has the potential to reduce educators' workload by automating the grading process, ensuring consistency and fairness. In conclusion, Bashir et al. (2021) suggest that the integration of machine learning and NLP in subjective answer evaluation could significantly improve the educational grading process. Although further research and dataset development are needed, their study demonstrates the promising potential of these technologies in revolutionizing the assessment of subjective responses. "Subjective Answers Evaluation Using Machine Learning and Natural Language Processing" by MF Bashir and Hamza Arshad (2021) explores innovative approaches to automating the evaluation of subjective answers in educational assessments. The authors discuss the challenges associated with subjective answer evaluation, including biases and inconsistencies that can arise in traditional grading methods. They emphasize the need for reliable and scalable solutions in educational settings. The book provides an overview of various natural language processing (NLP) techniques used in the evaluation process. This includes tokenization, part-of-speech tagging, sentiment analysis, and semantic analysis, which help in understanding the context and meaning of student responses. Bashir and Arshad explore different machine learning models, such as decision trees, support vector machines, and neural networks, that can be trained to recognize patterns in descriptive answers. A significant focus is placed on data preprocessing, including text normalization, stemming, and the creation of training datasets. The importance of clean and well-structured data for effective model training is highlighted. The book outlines various evaluation metrics used to assess the performance of machine learning models, such as accuracy, precision, recall, and F1 score. These metrics are crucial for understanding how well the system performs in real-world applications. Real-world applications of the proposed system are explored, showcasing how educational institutions can implement these technologies to enhance the grading process.. Overall, the book serves as a comprehensive guide for educators, researchers, and developers interested in leveraging technology to improve assessment methodologies. It outlines theoretical foundations while providing practical insights into implementing automated grading systems, making it a valuable resource in the field of educational technology.

[9] Systems Analysis and Design" by Gary B. Shelly and Harry J. Rosenblatt (2009)Gary B.

Shelly and Harry J. Rosenblatt's book Systems Analysis and Design (2009) is widely regarded as an essential text in the field of information systems development. The book offers a comprehensive examination of the systems development life cycle (SDLC), providing both foundational and advanced insights into the structured and iterative processes used to create effective information systems. A key strength of the book is its balanced coverage of traditional, structured methodologies as well as newer, agile approaches. This makes it relevant for students and professionals alike, as the industry continues to evolve in response to changing technology and project management practices.

The authors highlight those structured methodologies, like the SDLC, are critical for ensuring that projects follow a clear path from initiation to completion. At the same time, they recognize that agile methodologies are becoming increasingly important due to their flexibility and adaptability to change. Shelly and Rosenblatt also emphasize the importance of involving users throughout the systems development process. They argue that systems analysts must engage with users to gather accurate requirements and ensure the final system aligns with organizational goals. The book outlines a variety of techniques for gathering requirements, including interviews, surveys, and direct observation, ensuring that analysts can effectively capture the needs of stakeholders. Moreover, the authors focus on the importance of good design practices, particularly in creating user interfaces and system architecture that are both functional and intuitive. They advocate for a design process that considers both technical requirements and user experience, ensuring that systems not only meet the technical needs of an organization but are also easy for users to interact with.

The book also provides valuable insights into the post-implementation phase, stressing that system maintenance and support are critical for the long-term success of any information system. Shelly and Rosenblatt argue that without proper maintenance and support, even well-designed systems can fail to deliver sustained value to an organization..In conclusion, Systems Analysis and Design by Gary B. Shelly and Harry J. Rosenblatt (2009) remains a significant resource for understanding the principles and practices of systems development. Its comprehensive treatment of both structured and agile methodologies, along with a focus on user involvement, design, and maintenance, makes it a relevant and valuable text for both academic and professional audiences.

[5] Software Engineering" by Roger S. Pressman (1994)

Roger S. Pressman's Software Engineering: A Practitioner's Approach (1994) is one of the foundational texts in the field of software engineering. It provides a comprehensive exploration of the software development process, emphasizing both theoretical concepts and practical applications. This book has been widely used in academia and industry as a reference guide for understanding the principles of developing software systems effectively and efficiently.. The book delves into the key phases of the software development life cycle (SDLC), including requirements analysis, design, coding, testing, and maintenance. Pressman advocates for the importance of each phase, arguing that skipping or underestimating any part of the process can lead to project failure. Pressman introduces several models for software development, such as the Waterfall model, Incremental model, and Spiral model. These models provide frameworks for managing the complexity of software projects, offering different approaches based on the project's needs and constraints. The Waterfall model is noted for its linear progression through each phase, while the Incremental and Spiral models allow for iterative development and refinement. A significant contribution of Pressman's book is its focus on software quality. He emphasizes the importance of validation and verification techniques, which ensure that software systems meet user requirements and function correctly. Pressman advocates for rigorous testing methodologies, such as unit testing, integration testing, and system testing, as essential components of the software engineering process. Additionally, the book explores key software engineering concepts such as software metrics, project management, and risk management. Pressman stresses the need for quantifiable measures of software performance, which can help in tracking project progress and making informed decisions.

Pressman's work also addresses the growing importance of software maintenance, recognizing that software systems require ongoing updates and support after deployment. He categorizes maintenance into corrective, adaptive, and perfective, demonstrating the need for a long-term approach to software management. In conclusion, Roger S. Pressman's Software Engineering (1994) remains a key reference in the field, providing both a theoretical framework and practical tools for software development. Its structured approach to the SDLC, emphasis on software quality, and focus on risk management and maintenance make it an indispensable resource for students, educators, and professionals involved in software engineering projects.

CHAPTER 3

PROPOSED SYSTEM

The proposed system is an online platform designed to automate the evaluation of descriptive answers submitted in online assessments. This innovative solution aims to streamline the grading process for educators while providing a fair and consistent experience for students.

3.1 Key Features

1. **User-Friendly Interface**: The platform will have an intuitive front-end developed using HTML, CSS, and PHP, making it easy for both faculty and students to navigate. Faculty members will be able to create and manage assessments effortlessly, while students will find the testing process straightforward.
2. **Automated Answer Evaluation**: At the heart of the system is the use of Natural Language Processing (NLP) techniques. The platform will analyze student responses by comparing them to expected answers provided by faculty. Techniques such as tokenization and cosine similarity will enable the system to evaluate the relevance and accuracy of each response, significantly reducing grading time.
3. **Question Management**: Faculty will have the ability to create, edit, and organize a variety of questions. They can specify expected answers for each question, ensuring that assessments align with their teaching goals. This feature will also allow for the construction of custom tests tailored to specific topics or skill levels.
4. **Secure Data Management**: The platform will utilize a MySQL database to securely store and manage all user data, questions, and assessment results. This ensures that information is easily retrievable and safeguarded against unauthorized access.
5. **Feedback Mechanism**: After assessments, students will receive immediate feedback on their performance. This prompt response will help them identify areas for improvement and reinforce their learning.

3.2 Workflow

1. **Question Creation**: Faculty log into the platform and create questions, specifying the expected answers.

2. **Test Creation**: Faculty can assemble these questions into custom tests and assign them to students.
3. **Assessment Submission**: Students take the assessments online and submit their descriptive answers.
4. **Automated Evaluation**: Once submissions are received, the system automatically evaluates each response, comparing it to the expected answers.
5. **Results and Feedback**: Students receive their scores and detailed feedback, allowing them to understand their strengths and weaknesses.

3.3 Benefits

The proposed system aims to reduce the grading workload for educators, ensuring they can focus more on teaching and mentoring students. By providing timely and consistent evaluations, the platform enhances the overall educational experience, fostering a fair and supportive learning environment. It ensures consistency and objectivity in the grading process, minimizing biases and inconsistencies that can occur with manual assessments. The platform is scalable, capable of handling a large number of assessments simultaneously, making it suitable for institutions of varying sizes. Students receive immediate feedback on their answers, promoting a timely learning experience and enabling them to identify areas for improvement. This system enhances learning outcomes by providing specific feedback and suggestions, encouraging deeper understanding and mastery of subjects. Educators can create customizable assessments using a repository of questions, aligning tests with specific learning objectives. Comprehensive analytics offer valuable insights into student performance, helping educators track progress and identify common areas of struggle. The system supports diverse learning styles, accommodating various response styles for a more inclusive assessment approach. By reducing the administrative burden of grading, resources can be optimized, allowing educators to invest time in developing instructional materials and engaging with students. Implementing advanced technologies like NLP and machine learning prepares institutions for future educational trends, enhancing their technological capabilities. Additionally, the platform facilitates the development and maintenance of a knowledge repository, promoting knowledge sharing among educators and supporting collaborative assessment design.

CHAPTER 4

METHODOLOGY

The methodology for the ATS project focuses on building a reliable and efficient automated grading system for descriptive answers in online assessments. It employs advanced Natural Language Processing (NLP) techniques to ensure objective and consistent evaluations. The methodology can be divided into several key stages:

4.1 System Requirements Analysis

To develop the Automated Test Evaluation (ATS) system, an extensive requirements analysis was performed, targeting the core functionalities needed for automated descriptive answer grading. Through this process, both functional and non-functional requirements were identified by engaging stakeholders, including faculty and students. Functional requirements included the need for features like question management, answer evaluation, and score calculation, while non-functional requirements focused on usability, accuracy, and system responsiveness. This analysis ensured that the ATS system aligns with educational standards, meets user needs, and delivers fair and consistent evaluations.

4.2 System Design and Architecture

The architecture of the ATS system consists of a front-end interface built with HTML, CSS, and PHP to facilitate user interaction. For backend data management, MySQL is used to store user data, questions, and results. The core functionality is driven by Python, which powers the NLP-based automated answer evaluation. The system architecture is modular, divided into key components: the faculty module, question bank, student module, preprocessing module, and scoring module. Each module has a specific role and interacts seamlessly with others, providing a streamlined and cohesive solution for automated descriptive answer evaluation. This modular structure also allows for easy future upgrades, such as incorporating advanced AI features.

4.3 Development and Implementation

In the development phase, various NLP techniques were integrated to enable the automated grading process. Key modules were developed to handle tasks, from text preprocessing to answer similarity analysis.

Preprocessing Module: This module prepares student answers by transforming raw text into a standardized format, crucial for accurate comparison. Preprocessing techniques include:

Tokenization: Splits text into individual words or "tokens," making it easier to process. For instance, the sentence “Automated grading is efficient” would be split into tokens: ["Automated", "grading", "is", "efficient"].

Stop-word Removal: Common words like "is," "the," and "of" are filtered out, as they don’t contribute to the unique meaning of a response.

Stemming: Reduces words to their root forms; for example, “grading” becomes “grade.” This ensures that different forms of a word don’t affect similarity calculations.

Pattern Mining Module: After preprocessing, key patterns in student responses are extracted. Using techniques like part-of-speech tagging and phrase recognition, the module identifies essential elements, such as nouns and verbs, that capture the core meaning of each answer.

Similarity Comparison Module: This module uses cosine similarity to evaluate the semantic closeness between a student’s answer and the expected answer. Cosine similarity is a metric that quantifies how similar two text vectors are, with values ranging from -1 to 1. A higher cosine similarity score indicates a closer match between the two answers.

Cosine Similarity Calculation

In ATS, student and model answers are transformed into vector representations, where each dimension corresponds to a unique term (word) from both texts. The cosine similarity equation is applied to these vectors to measure the degree of similarity:

Cosine Similarity = $A.B / ||A|| \times ||B||$

-A.B denotes the dot product of vectors A and B, calculated as the sum of the products of corresponding terms.

- ||A|| and ||B|| are the magnitudes (or norms) of vectors A and B respectively.

The cosine similarity score helps the ATS system assign a grade by assessing the overlap between student and model answers. For instance, if two answers have a cosine similarity of 0.9, they are highly similar, indicating that the student’s answer closely matches the expected response.

Scoring Module: Based on the cosine similarity results, the scoring module assigns scores by setting a threshold for similarity. If the similarity score meets or exceeds the threshold, the answer is awarded points; otherwise, it is partially scored or marked for review. This module also considers linguistic diversity, recognizing different phrasings of correct answers.

4.4 Testing and Validation

To ensure the accuracy and reliability of the ATS system, multiple levels of testing were performed. **Unit Testing** involved testing each NLP component individually, such as tokenization and cosine similarity calculation, to verify their accuracy. **Integration Testing** was conducted to confirm that the various modules worked harmoniously together; for instance, the integration between the faculty module and the question bank was checked to ensure smooth test creation and answer evaluation. Finally, **User Acceptance Testing (UAT)** involved actual users who validated the system's ease of use and performance, and the feedback gathered during this phase ensured that the system met users' expectations and educational standards.

4.5 Deployment and Maintenance

The ATS system was deployed on a secure web server, ensuring accessibility across various devices and platforms for both students and faculty. This deployment includes robust security measures to protect sensitive data and maintain user privacy.Future maintenance will focus on scaling the platform to accommodate a growing number of users and institutions, addressing increasing demand. Additionally, the integration of advanced NLP techniques, such as transformer-based models like BERT, is planned to enhance grading accuracy. These models will improve the system’s ability to understand context and semantics, leading to more nuanced evaluations of student responses.Planned enhancements will also support additional question types, such as multiple-choice, fill-in-the-blank, and essay formats, allowing for greater flexibility in assessments. This adaptability will cater to diverse educational needs and enable educators to create a wider variety of tests.User feedback will remain integral to the development process, guiding regular updates to ensure the system evolves in response to faculty and student needs. Overall, the ATS system is positioned for continuous improvement, aiming to set a new standard for automated evaluation in educational environments and to better support diverse learning contexts.

CHAPTER 5
SYSTEM ARCHITECTURE

5.1 System Specification

The outlined hardware and software specifications form the foundation of system's development and implementation. These prerequisites ensure optimal performance, enabling seamless operation and efficient utilization of the proposed innovative functionalities. The specifications set the stage for a robust and versatile system poised to transform traditional question management and assessment paradigms.

5.2 Specification for Development

This section outlines the necessary hardware and software requirements for the project

5.2.1 Hardware Specification

CPU	intel Core i3 or higher
RAM	4 GB or higher
Hard Disk	512 GB or higher

5.2.2Software Specification

OS	Windows 10
Front end tool	HTML,CSS
Back end tool	PHP, JavaScript, Python
Database	MySQL
Web Browser	Any Browser
IDE	Visual Studio Code

5.3 Software Tools

The selection of diverse software tools like Python, HTML, PHP, MySQL, Bitbucket and Visual Studio Code underscores my system's multifaceted approach. Leveraging these tools enables to develop a robust, versatile and user centric platform catering to varied needs in question

management and descriptive answer assessment. Their synergy empowers seamless integration and enhances my system's adaptability and efficiency.

5.3.1 Python

Python is a computer programming language often used to build websites and software, automate tasks and conduct data analysis. Python is developed by Guido van Rossum in 1989. Python is a general-purpose language, meaning it can be used to create a variety of different programs. Python has simple syntax similar to the English language and the syntax allows developers to write programs with fewer lines of code. Since it is open-source, there are many libraries available that make developers' jobs easy, ultimately resulting in high productivity.

5.3.2 HTML

HTML stands for Hypertext Markup Language, was invented by Tim Berners-Lee. It is a simple text formatting language used to create hypertext documents. HTML, the foundational language of the web, structures content through simple tags and attributes. With HTML5 advancements, it introduces new elements for enhanced semantics and multimedia integration, facilitating dynamic and visually engaging websites when combined with CSS and JavaScript.

5.3.3 PHP

PHP is a widely-used server-side scripting language designed for web development, seamlessly embedding within HTML. It's known for simplicity, flexibility and versatility, supporting various databases and enabling dynamic web applications.

5.3.4 MySQL

MySQL, an open-source relational database management system, plays a crucial role in organizing data for dynamic web applications. Owned by Oracle, MySQL follows the client-server model, supporting SQL for efficient database management, transactions and high-performance indexing

5.3.5 GitHub

GitHub is a web-based interface that uses Git, the open source version control software that lets multiple people make separate changes to web pages at the same time. As Carpenter notes, because it allows for real-time collaboration, GitHub encourages teams to work together to build and edit their site content.

5.3.6 Visual Studio Code

Visual Studio Code (VS Code), a Microsoft-developed source-code editor, is known for its versatility and cross-platform compatibility (Windows, macOS, Linux). As a free and open-source tool with intelligent code editing, Git integration and task automation it's widely favored by developers for various projects.

5.3 Client Environment

The online assessment evaluation platform is designed to be user-friendly and accessible across various devices. For optimal performance and user experience, the following client specifications are recommended:

Operating System

- **Windows**: Windows 10 and above
- **macOS**: macOS Mojave (10.14) and above
- **Linux**: Latest stable distributions (e.g., Ubuntu 20.04 or Fedora 34)

Web Browser

The platform is compatible with modern web browsers to ensure smooth functionality:

- **Google Chrome**: Latest version
- **Mozilla Firefox**: Latest version
- **Microsoft Edge**: Latest version
- **Safari**: Latest version on macOS.

These specifications ensure that the system can effectively handle the demands of automated answer evaluation and provide a consistent, personalized learning experience for all users.

CHAPTER 6

MODULES

A module is a collection of source files and build settings that allows us to divide our project into discrete units of functionality. The project can have one or many modules and one module may depend on another module. In this project, there are the Faculty and Student modules.

6.1 Faculty Module

The faculty module is designed to empower instructors and educators with comprehensive tools for effective question management and assessment. Faculty members using this module can efficiently manage question repositories, organizing and categorizing questions for various subjects. The module allows faculty to create tests by selecting questions from the repository. In addition to test creation, instructors can also view and analyze students' results, gaining insights into individual and overall performance.

6.2 Question Bank Module

The question bank module serves as a comprehensive repository for organizing and managing educational questions. It facilitates the categorization of questions based on semesters and subjects, providing a structured storage system. This module enables the generation of PDF question papers tailored to specific semesters and subjects.

6.3 Student Module

The student module is tailored to provide students with a user-friendly interface for test taking and result tracking. Within this module, students can access and attempt tests assigned by faculty members. The module supports features time-bound assessments. After completing a test, students can instantly view their results.

6.4 Preprocessing Module

The input text document is changed into a normalized form using preprocessing. There are four main activities performed in this stage:

- Segmentation
- Tokenization

• Removing Stop words

• Word Stemming

Sentence segmentation is boundary detection and separating source text into a sentence. Tokenization is separating the input document into individual words. Next, removing stop words, stop words are the words which appear frequently in the document but provide less meaning in identifying the important content of the document, such as 'a', 'an', 'the', etc. The last step for preprocessing is word stemming; word stemming is the process of removing prefixes and suffixes of each word.

6.5 Pattern Mining Module

Tokens are generated after preprocessing. The sequential pattern is generated from the tokens of each sentence. The count for each obtained pattern is also determined. Most frequently occurred patterns are obtained based on the count.

6.6 Similarity Comparison Module

Given two sentences, the measurement determines how similar the meaning of two sentences is. The higher the score, the more similar the meaning of the two sentences. Cosine-based similarity approach is used to calculate the similarity between the input answer and the model description. To compute cosine similarity between two sentences s1 and s2, sentences are turned into terms/words, words are transformed in vectors. Each word in texts defines a dimension in euclidean space and the frequency of each word corresponds to the value in the dimension.

$$\text{Cosine Similarity} = \mathbf{A} \cdot \mathbf{B} \;/\; \|\mathbf{A}\| * \|\mathbf{B}\|$$

Where:

$A \cdot B$ represents the dot product of vectors A and B

$\|A\|$ and $\|B\|$ denote the magnitudes (or norms) of vectors A and B respectively.

6.7 Scoring Module

The given answers are allotted with a score according to this novel strategy. Identifying both commonalities and similarity between the students' answers scores are assigned.

CHAPTER 7

DIAGRAMS

The design phase is an essential part of any system development process, where creative and structured methodologies are employed to define the architecture and workflows of the system. A well-executed design ensures the system's effectiveness, efficiency, and ability to meet its intended objectives. The following diagrams illustrate the structural and functional aspects of the system, including data flow, user interactions, and system components. Each diagram offers a detailed view of how different modules interact with one another, contributing to the overall functionality of the platform.

7.1 DFD

A Data Flow Diagram (DFD) is a visual representation of how data moves through a system. It illustrates the flow of information between processes, data stores, and external entities. DFDs are commonly used in systems analysis and design to help stakeholders understand how data is processed and transformed within an information system.

7.1.1 DFD (Level 0)

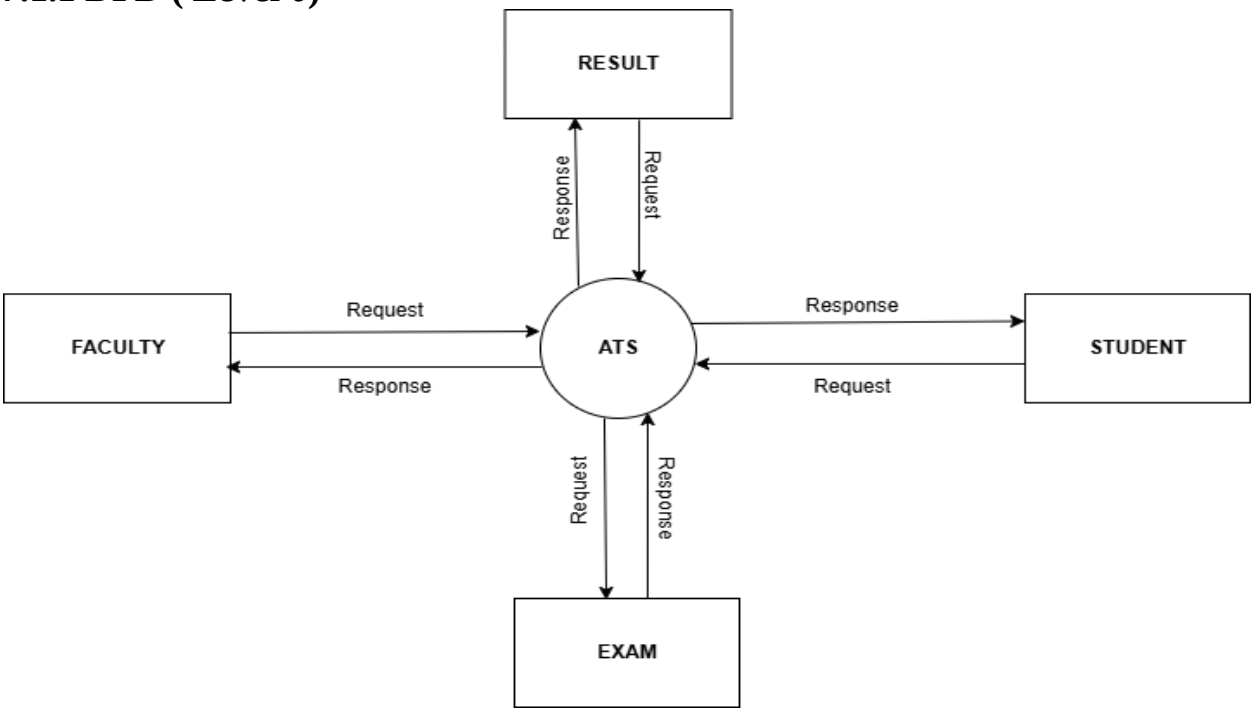

Fig. 7.1.1 DFD Level 0

7.1.2 DFD LEVEL 1 (FACULTY)

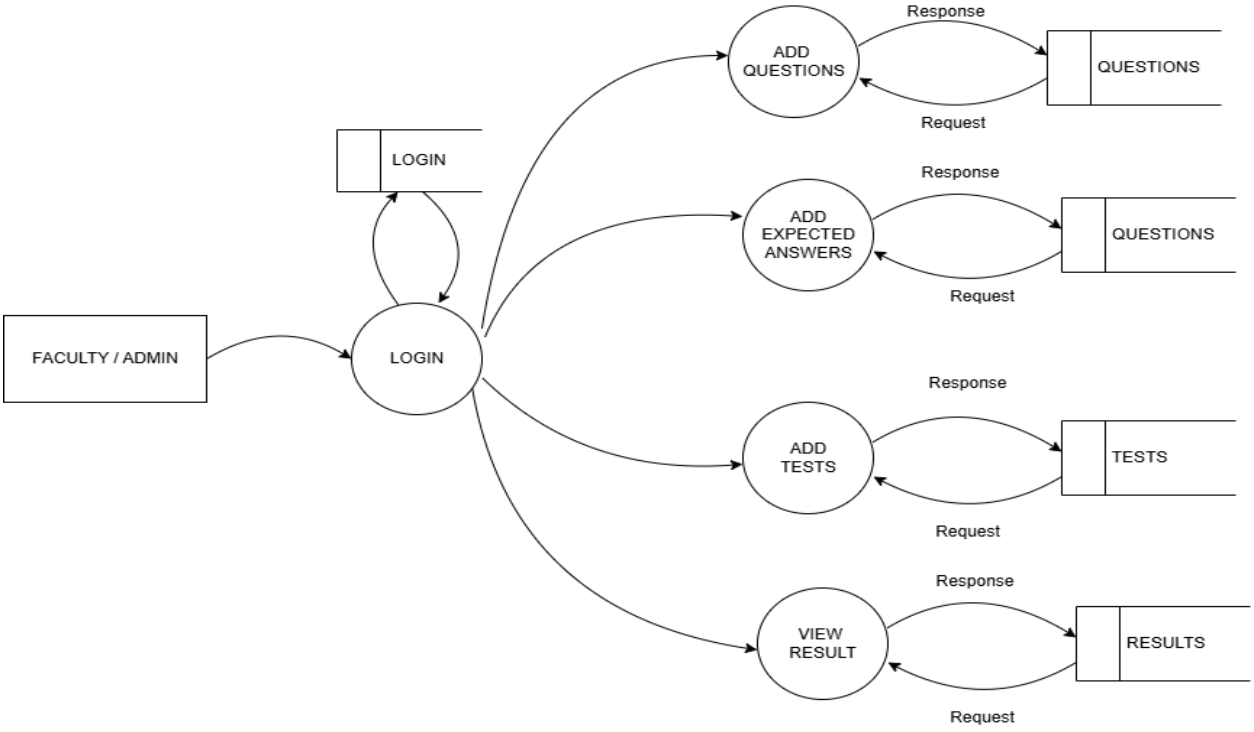

Fig. 7.1.2 DFD Level 1

7.1.3 DFD LEVEL 1 (STUDENT)

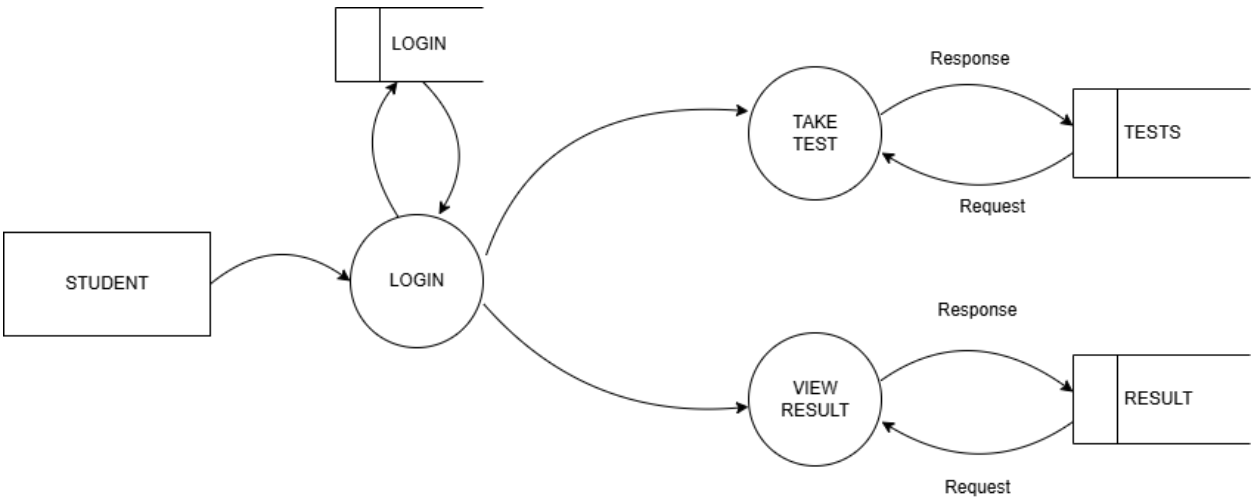

Fig. 7.1.3 DFD Level 2

7.2 ACTIVITY DIAGRAM

The UML Activity Diagram is a graphical tool used to represent the flow of control in a system, focusing on the dynamic aspects rather than implementation specifics. It visually models both sequential and concurrent activities, showcasing how tasks are executed in a particular workflow. The diagram effectively represents the transitions between different actions or states within a system, highlighting conditions, decisions, and branching points.

Key components of the activity diagram include:

- **Actions**: Individual tasks or operations.
- **Decision nodes**: Points where the flow can diverge based on certain conditions.
- **Forks and Joins**: These allow for parallel or concurrent flows to be modeled, showing how activities can split into multiple paths or merge back together.
- **Start and End**: Denotes the initiation and termination of the workflow.

The activity diagram for your online platform provides a detailed visualization of the workflow for automatically evaluating descriptive answers, emphasizing the interactions between faculty members and students. Faculty members play a crucial role by creating questions and specifying expected answers, which are stored in the system's database for future use. Once the questions are prepared, they can generate custom tests that are accessible to students. Students engage with the platform by taking these tests and submitting their answers through an intuitive user interface. Upon submission, the system leverages advanced Natural Language Processing (NLP) techniques to evaluate the responses. This evaluation involves comparing the submitted answers against the expected answers and generating constructive feedback based on the results. This feedback is instrumental in helping students understand their performance and identify areas for improvement. The diagram may also include decision points that determine whether a submission meets expected criteria or requires further clarification, thereby influencing the feedback provided. Overall, the activity diagram captures the flow of operations within the system, highlighting key user interactions and system processes.

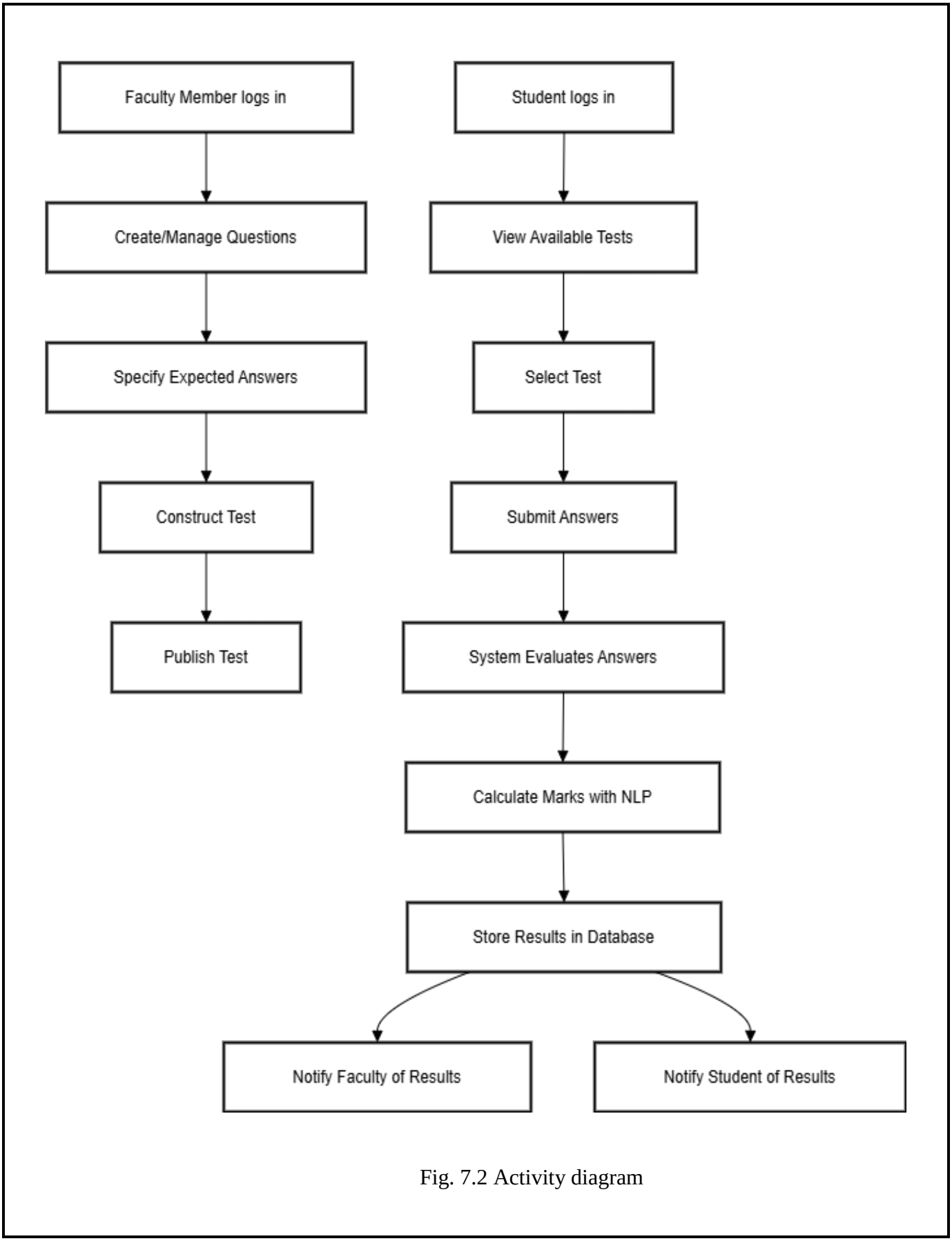

Fig. 7.2 Activity diagram

7.2.1 Components of an Activity Diagram

In an activity diagram, several components work together to represent the control flow and system behavior effectively. These components are essential in modeling the dynamic aspect of the system:

a) Activities

An **activity** represents a unit of behavior or a set of actions. Each activity is a collection of nodes connected by edges, indicating the flow of control. Activities often represent high-level processes that consist of smaller, discrete actions or tasks performed sequentially or concurrently.

b) Activity Partition (Swim Lane)

The **swim lane** is a mechanism for grouping related activities into distinct columns or rows within the diagram. It helps to indicate the ownership or responsibility of each activity by an actor, department, or system. Swim lanes can be either vertical or horizontal and provide modularity and clarity, especially when modeling complex workflows.

c) Forks

Fork nodes facilitate the concurrent execution of multiple activities. A fork has one inward edge and multiple outward edges, effectively splitting the flow into parallel processes. This is useful for representing decisions where multiple paths can be taken simultaneously.

d) Join Nodes

A **join node** merges multiple concurrent flows back into a single flow. It is the opposite of a fork and performs a logical AND operation on all incoming edges, ensuring that all concurrent processes are completed before the flow continues along the single outgoing edge. Join nodes help synchronize parallel activities in the system.

7.3 CLASS DIAGRAM

The **Class Diagram** is a static representation of the system, illustrating the types of objects, their properties, methods, and relationships within the application. A class diagram captures the blueprint of a system by describing its structure in terms of classes and their interactions. This diagram not only aids in visualizing and documenting system architecture but also plays a critical role in building executable code by outlining the system's foundational elements.

A **class** is composed of its objects, and classes can inherit attributes and behaviors from other classes. The diagram encapsulates various elements such as attributes, functions, and relationships that provide an overview of how the system behaves. These components are placed into separate compartments for clarity, allowing developers to easily navigate and interact with the structure.

Since class diagrams represent classes, interfaces, associations, collaborations, and constraints, they are categorized as structural diagrams.

7.3.1 Components of a Class Diagram

The class diagram consists of three primary sections:

- **Upper Section**: This section contains the name of the class. A class represents a blueprint for similar objects that share common relationships, attributes, operations, and behaviors. The class name is always positioned at the top.
- **Middle Section**: This section contains the attributes (variables) of the class, which describe the qualities or properties of the class. Attributes represent the data stored in the class, and each class can have multiple attributes with unique types.
- **Lower Section**: The lower section includes the methods or operations that define the behavior of the class. Each method is typically listed in a single line. The methods show how a class interacts with the data it contains and how it provides functionality.

7.3.2 Relationships in Class Diagrams

In UML, relationships between classes are essential for understanding how different parts of the system interact. There are three main types of relationships in a class diagram:

- **Dependency**: A dependency is a weak relationship between two or more classes where changes in one class can cause changes in the dependent class. It typically represents a situation where one class uses or depends on the functionality of another.
- **Generalization**: This relationship represents inheritance between a parent class (superclass) and a child class (subclass). The subclass inherits attributes and behaviors

from the superclass. Generalization shows an "is-a" relationship, meaning the child class is a more specific form of the parent class.

Association: Association is a static or physical connection between two or more classes or objects. It describes how objects are related to each other and often represents real-world relationships. Multiplicity indicators in an association show how many objects are involved in the relationship.

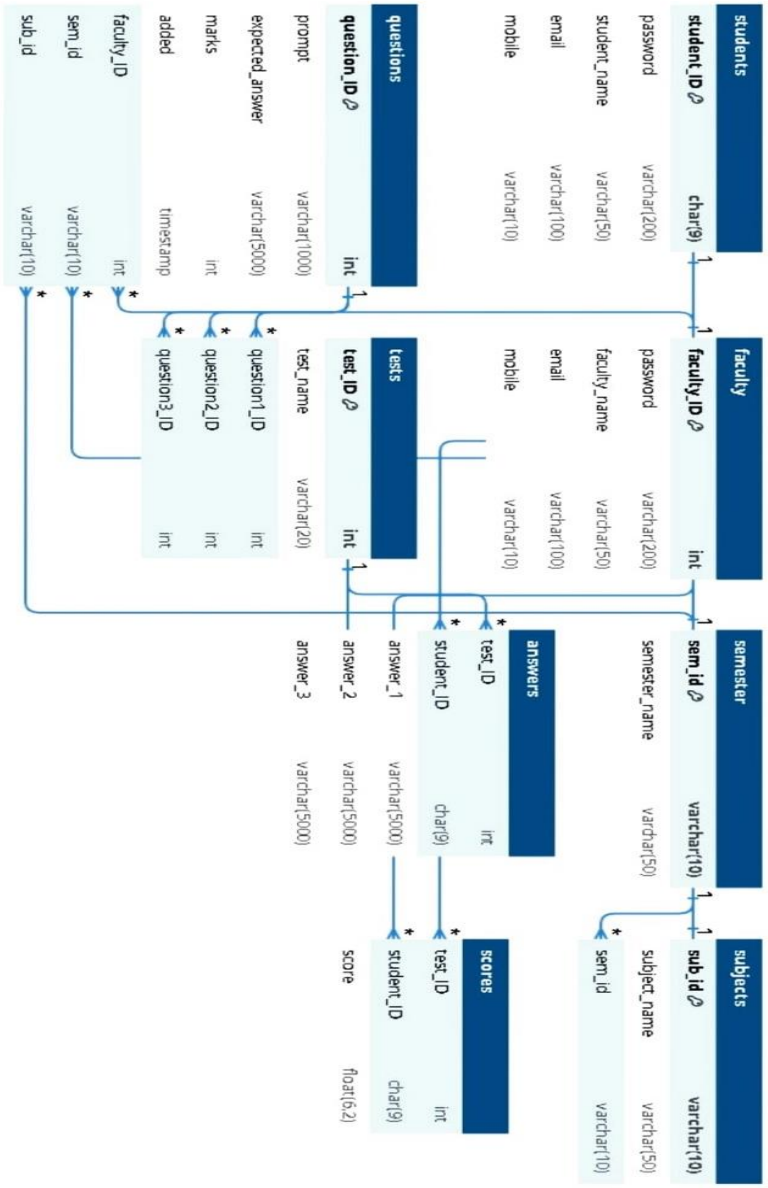

Fig 7.3

7.4 USE CASE DIAGRAM

A **Use Case Diagram** is a visual representation of the interactions between the components of a system. It is used in system analysis to define, describe, and structure system requirements by illustrating how users (actors) interact with the system. This method helps clarify the roles, actions, and the flow of interactions within and around the system. Use case diagrams are a core part of UML (Unified Modeling Language) and are widely used to model real-world objects and systems, either in development or already in operation. They offer a high-level view of system functionality and user interaction. The use case for your online platform focuses on automating the evaluation of descriptive answers in educational assessments. Faculty members create questions and specify expected answers, which are stored in a centralized database. Students then submit their answers through an intuitive interface. The system employs Natural Language Processing (NLP) techniques to analyze these responses, comparing them against the expected answers to generate scores. Immediate feedback is provided to students, highlighting strengths and areas for improvement. This approach reduces the grading workload for educators while ensuring consistent and objective evaluations, ultimately enhancing the educational experience and promoting student engagement.

7.4.1 Components of a Use Case Diagram

1. **System Boundary**:

   The system boundary defines the scope of the system and sets it apart from the external world. Everything inside the boundary is part of the system, while everything outside represents external actors and elements that interact with the system.

2. **Actors**:

   Actors represent individuals, external systems, or entities that interact with the system. These actors are defined according to their roles, which can be either human users or other systems that interface with the system being modeled.

3. **Use Cases**:

   A use case defines a specific interaction between the actor and the system to achieve a particular goal. Each use case represents a function or behavior of the system from

the user's perspective. The use cases help to identify and explain the functionalities the system provides.

4. **Relationships**:

Relationships in use case diagrams represent the associations between actors and use cases, and between use cases themselves. These relationships describe the interactions and dependencies, such as which actors are responsible for which tasks and how different use cases work together. The relationships are typically represented using lines connecting actors to the relevant use cases or between use cases to show dependencies or extensions.

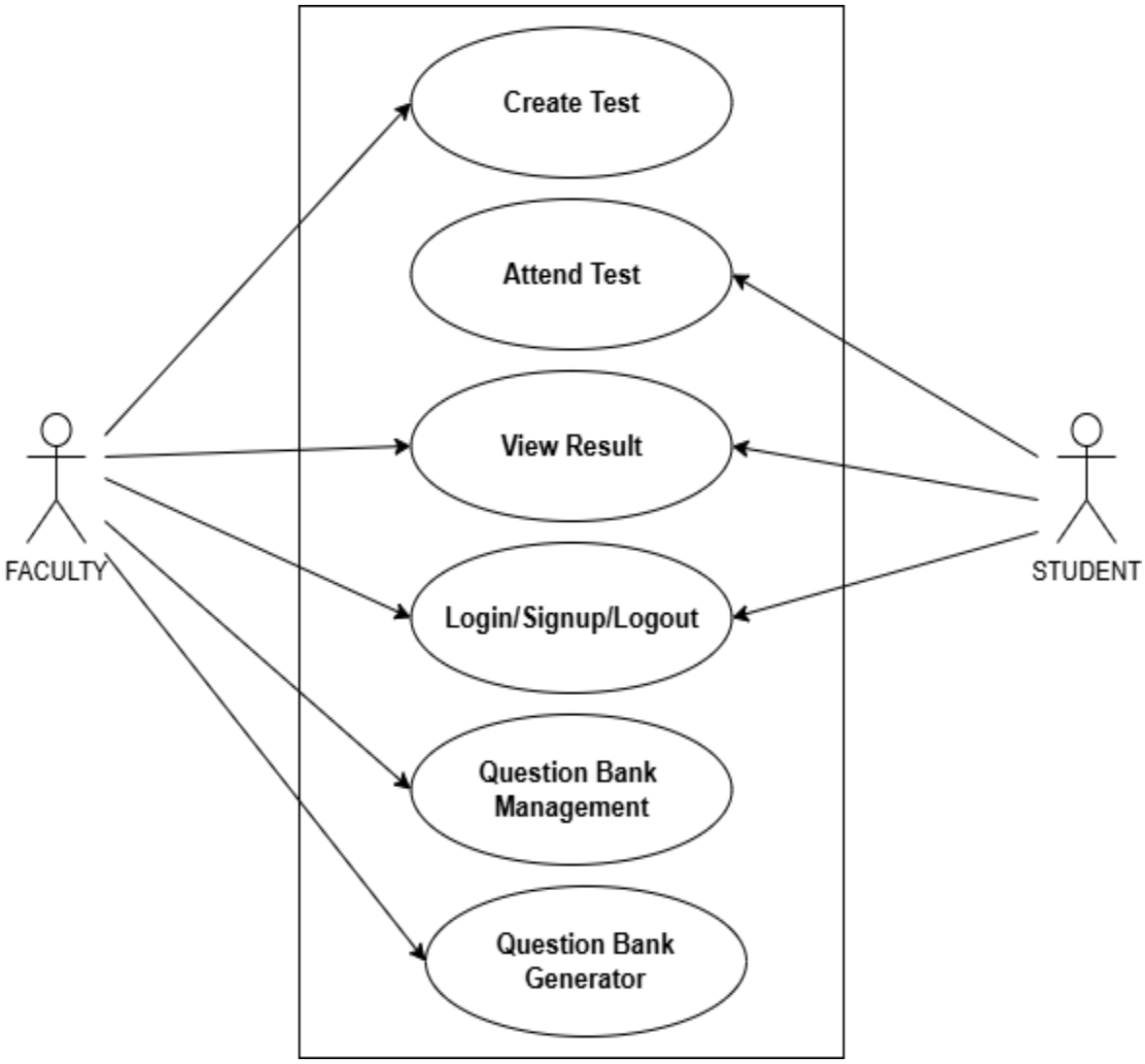

Fig 7.4

CHAPTER 8
TESTING

8.1 Introduction

Software testing is a crucial process used to determine whether the software operates according to its specified requirements. It serves to verify and validate that the system meets both the functional and performance expectations. The primary goal of testing is to ensure that the software functions as intended, detecting any defects or issues before deployment. There are three key approaches to testing:

- **Testing for correctness**: Ensures the software produces the correct output for given inputs.
- **Testing for implementation efficiency**: Assesses whether the software implementation is optimized and efficient.
- **Testing for computational complexity**: Ensures the software handles complex tasks within acceptable time and resource constraints.

8.1.1 Test Plan

A test plan outlines the series of steps taken during various testing phases. It acts as a guide for developers and testers to ensure that all components of the system are tested comprehensively. In this project, the testing process involves both developers and independent testers to mitigate bias. The following levels of testing are applied:

- **Unit Testing**
- **Integration Testing**
- **Validation (System) Testing**
- **Output Testing (User Acceptance Testing)**
- **Automation Testing**

8.2 Unit Testing

Unit testing is a software testing method where individual units or components of a software application are tested in isolation. The primary purpose is to validate that each unit of the software

code performs as expected. Unit testing is typically carried out during the development (coding phase) of an application by the developers themselves. Unit tests isolate a specific section of code and verify its correctness. A unit in this context may refer to an individual function, method, procedure, module, or object.

Test Case: Verify that the system allows faculty to add a new question to the repository.

Test Steps:

• Faculty logs in and navigates to the question repository.

• Adds a new question with relevant details (subject, topic, options).

Expected Result: Question is successfully added and appears in the repository.

8.3 Integration Testing

Integration Testing is a critical phase in software testing, focused on assessing how different components or units within a system interact. It aims to identify and resolve issues that may arise when individual units are combined to form larger modules or the entire system. This phase reveals 12 inconsistencies, data flow problems, or communication errors that may occur during the integration of diverse units.

Test Case: Validate the integration between the question bank and faculty module.

Test Steps: • Faculty selects questions from the question bank to create a test.

• Test creation occurs with selected questions.

• Faculty views the test in the faculty module dashboard.

Expected Result: Selected questions seamlessly integrate into the test creation process and display correctly in the faculty's dashboard.

8.4 Output Testing (User Acceptance Testing)

User Acceptance Testing (UAT) is the final phase of software testing where intended users or stakeholders evaluate the software to ensure it meets their needs and functions as expected in

a real-world setting. During UAT, users perform tasks based on real-life scenarios, provide feedback on issues or improvements and validate that the software aligns with specified business requirements. Successful UAT results in the approval for software deployment to the production environment confirming that the system is ready for actual use.

Test Case: Assess the usability of the student module during a test attempt.

Test Steps: • Student logs in and selects a test to attempt.

• Completes the test within the specified time.

Expected Result: Student navigates the test easily, submits answers and views results upon completion.

8.5 Automation Testing

Automation testing is crucial for ensuring that the system operates correctly under various conditions without requiring manual intervention. This project utilizes automation testing tools to create scripted test cases for UI interactions and data flows. Automated testing is especially beneficial for running repetitive test cases across different environments, ensuring consistent performance and functionality.

- **UI Automation**: Automated scripts are used to verify the user interface's behavior, including the responsiveness of the **Faculty**, and **Student** modules.
- **Cross-platform Testing**: Ensures that the platform functions correctly across different devices and operating systems.

8.6 Test Cases

Name	Validations	Input	Response
Login Credentials	User enters incorrect username and password	Invalid username/password	Login Failed
User Registration	Required fields (ID, Name, Email, Contact, Password) not filled	Incomplete details	Validation error

Table 8.6

CHAPTER 9

ADVANTAGES & DISADVANTAGES

9.1 ADVANTAGES

1. **Time Efficiency**:
    - Automated evaluation significantly reduces grading time for educators, allowing them to focus more on teaching and providing personalized feedback.
2. **Consistency and Objectivity**:
    - The use of Natural Language Processing (NLP) ensures a standardized evaluation process, minimizing biases and inconsistencies that can arise from manual grading.
3. **Immediate Feedback**:
    - Students receive instant feedback on their submissions, which can enhance their learning experience and help them identify areas for improvement quickly.
4. **Scalability**:
    - The platform can handle a large volume of assessments simultaneously, making it suitable for institutions with many students and courses.
5. **Enhanced Learning Analytics**:
    - The system can provide valuable insights into student performance trends, helping educators identify common areas of difficulty and adjust their teaching strategies accordingly.
6. **Question Management**:
    - Faculty can easily create, edit, and organize a repository of questions, streamlining the test creation process and ensuring a diverse range of assessments.
7. **Customizable Assessments:**
    - Educators can tailor assessments to meet specific learning objectives or student needs, promoting a more personalized learning experience.
8. **Resource Optimization:**
    - By automating grading, educators can allocate more time to developing instructional materials and engaging with students, enhancing overall educational quality.

9.2 DISADVANTAGES

1. **NLP Limitations**:
    - NLP techniques may struggle with nuanced or complex language, leading to potential inaccuracies in evaluating responses that require deeper understanding or creativity.
2. **Dependency on Expected Answers**:
    - The system relies heavily on the clarity and specificity of expected answers provided by faculty. Vague or poorly defined answers can lead to inconsistent grading outcomes.
3. **Initial Setup Time**:
    - The implementation of the system requires an initial investment of time and resources for setup, training, and integration into existing educational frameworks.
4. **Technical Challenges**:
    - Developing and maintaining the NLP algorithms may require specialized technical expertise, which could be a barrier for some institutions.
5. **Resistance to Change**:
    - Educators and students accustomed to traditional grading methods may be resistant to adopting automated systems, requiring change management efforts to encourage acceptance.
6. **Limited Scope**:
    - The focus on descriptive answers means that the system may not address other question types, such as multiple-choice or true/false questions, potentially limiting its overall utility.
7. **Over-Reliance on Technology:**
    - There is a risk that educators may become overly reliant on automated systems for grading, potentially diminishing their engagement with students and their understanding of individual student needs. This could lead to a less personalized educational experience.

CHAPTER 10

RESULTS AND CONCLUSION

10.1 Results:

This chapter explores the outcomes of the automated test scrutiny system, presenting a paradigm shift from traditional assessment methodologies prevalent in educational institutions.

The automated test scrutiny system successfully addresses the limitations inherent in current assessment methodologies prevalent in educational institutions. By shifting the focus from over-reliance on multiple choice questions to the evaluation of theoretical knowledge through descriptive answers, the system introduces a transformative approach to student assessment. The key outcomes and results of the project are highlighted below:

10.1.1 Efficient Evaluation Process

• The system introduces efficiency into the evaluation process by automating the assessment of descriptive answers.

• Reduction in manual effort leads to a significant decrease in errors and a considerable saving of time and resources for educators.

10.1.2 Innovative Question Bank

• The implementation of the question bank feature streamlines question management for faculty.

• Educators can efficiently categorize, organize, customize assessments and generate custom question papers contributing to a more dynamic and adaptable evaluation system.

Chapter 10, ”Result and Conclusion,” demonstrates the descriptive answer evaluation system’s success in automating assessments, reducing errors and saving educator’s time. The innovative question bank feature enhances faculty efficiency, allowing for dynamic and customized assessments. Overall, the chapter emphasizes the transformative impact on student evaluation methodologies

10.2 Conclusion

This chapter encapsulates the automated test scrutiny system’s response to prevailing assessment challenges in educational institutions. Departing from the over-reliance on multiple-choice

questions, the system prioritizes evaluating student's theoretical knowledge through descriptive answers. The automated test scrutiny system addresses the limitations of current assessment trends in educational institutions, specifically the over-reliance on multiple-choice questions. Focused on evaluating student's theoretical knowledge through descriptive answers, the system aims to minimize human effort, reduce errors and save valuable time and resources. The innovative question bank feature streamlines question management, allowing educators to categorize, organize and customize assessments efficiently. The system's impact is significant, offering a transformative solution that enhances efficiency in the evaluation process. In educational institutions, exams are crucial, but existing automatic assessment systems often focus on multiple-choice questions, neglecting theoretical knowledge evaluation. This project aims to overcome these limitations by proposing a system that prioritizes assessing descriptive answers, providing a time and resource saving solution for institutes. The system evaluates students at an advanced level, assessing descriptive answers with multiple sentences. By automating the process, it saves time and resources for educational institutes. The chapter concludes the automated test scrutiny system's transformative impact on educational assessments. By addressing the limitations of multiple choice centric approaches, the system minimizes human effort, reduces errors and streamlines question management through its innovative question bank feature. The chapter highlights the system's efficiency in enhancing the overall evaluation process. This chapter encapsulates the transformative potential of the automated test scrutiny system in addressing the prevalent assessment challenges faced by educational institutions. Traditional assessment methods often rely heavily on multiple-choice questions, which, while efficient, tend to overlook the nuances of students' theoretical knowledge and critical thinking skills. This project proposes a paradigm shift, emphasizing the evaluation of descriptive answers to foster deeper understanding and engagement among students.The automated test scrutiny system effectively tackles the limitations associated with existing assessment trends. By prioritizing the assessment of descriptive answers, the system not only evaluates students' comprehension of complex concepts but also encourages them to articulate their thoughts more coherently. This approach promotes higher-order thinking skills, essential for success in both academic and real-world contexts. One of the standout features of the system is its innovative question bank, which significantly streamlines the management of assessments.

CHAPTER 11

APPENDICES

Code :

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>ATS - Automated Test Scrutiny</title>
    <link rel="stylesheet" href="style1.css">
    <link href="https://fonts.googleapis.com/css?family=Open+Sans:300,400,600&display=swap" rel="stylesheet">
</head>
<body>
    <header class="header">
        <h1>ATS - Automated Test Scrutiny</h1>
        <p>Your gateway to innovative assessments</p>
    </header>
    <div class="hero-section">
        <div class="hero-text">
            <h2>Welcome to ATS</h2>
            <p>Transforming the assessment experience for students and faculty.</p>
            <div class="button-container">
                <a class="button" href="student-login.php">Student Login</a>
```

```
            <a class="button" href="faculty-login.php">Faculty Login</a>
        </div>
    </div>
</div>
<footer>
    <p>&copy; 2024 ATS. All rights reserved.</p>
</footer>
</body>
</html>
```

evaluate.py

```
import json
import numpy as np
from collections import Counter
import math
import os
import nltk
from nltk.tag import pos_tag
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import PunktSentenceTokenizer
from nltk.corpus import wordnet, stopwords
from nltk import sent_tokenize, word_tokenize, PorterStemmer
import pandas as pd
import sys

nltk.download('punkt')
nltk.download('wordnet')
```

```
nltk.download('stopwords')
nltk.download('punkt_tab')


ps = PorterStemmer()
lemmatizer = WordNetLemmatizer()
stop_words = stopwords.words('english')
special = ['.', ',', '\'', '"', '-', '/', '*', '+', '=', '!',
          '@', '$', '%', '^', '&', '``', '\'\'', 'We', 'The', 'This']


def normalise(word):
    word = word.lower()
    word = ps.stem(word)
    return word


def get_cosine(vec1, vec2):
    intersection = set(vec1) & set(vec2.keys())
    numerator = sum([vec1[x] * vec2[x] for x in intersection])


    sum1 = sum([vec1[x]**2 for x in vec1.keys()])
    sum2 = sum([vec2[x]**2 for x in vec2.keys()])
    denominator = math.sqrt(sum1) * math.sqrt(sum2)


    if not denominator:
        return 0.0
    else:
        return numerator / denominator
```

```
def text_to_vector(text):
    words = word_tokenize(text)
    vec = []
    for word in words:
        if (word not in stop_words):
            if (word not in special):
                w = normalise(word)
                vec.append(w)
    # print Counter(vec)
    return Counter(vec)

def docu_to_vector(sent):
    vec = []
    for text in sent:
        words = word_tokenize(text)
        for word in words:
            if (word not in stop_words):
                if (word not in special):
                    w = normalise(word)
                    vec.append(w)
    # print Counter(vec)
    return Counter(vec)

def f_s_to_s(orig_answer, N, sent):
    cosine_mat = np.zeros(N+1)

    row = 0
```

```
    for text in orig_answer:
        maxi = 0
        vector1 = text_to_vector(text)

        for text1 in sent:
            vector2 = text_to_vector(text1)
            cosine = get_cosine(vector1, vector2)

            if (maxi < cosine):
                maxi = cosine

        cosine_mat[row] = maxi

        row += 1

    return cosine_mat

def get_score(orig_ans, stud_ans, max_mark):

    model_answer = sent_tokenize(orig_ans)

    test_answer = sent_tokenize(stud_ans)

    N = len(model_answer)
    mat = f_s_to_s(model_answer, N, test_answer)

    cnt = docu_to_vector(test_answer)
```

```
    cnt = cnt.most_common(10)

    if len(cnt) == 0:
        score = 0
    else:
        thematic = []
        for i in range(len(cnt)):
            thematic.append(cnt[i][0])

        tot = 0

        thematic = ",".join(str(x) for x in thematic)
        thematic = text_to_vector(thematic)

        for i in test_answer:
            i = text_to_vector(i)
            tot = tot + get_cosine(thematic, i)

        point = sum(mat)
        score = point * max_mark * 3/(4*N)
        score = round(min(score + (tot/1.5), max_mark), 2)

        return score

def evaluate():
    scores = []
    max_mark = 10
```

```
df = pd.read_csv(sys.argv[1])

# Student's answer and faculty's expected answer for question 1
stu_ans1 = df.iloc[0]["student1"]
unique_words = set(stu_ans1.split())
stu_ans1 = ' '.join(unique_words)
exp_ans1 = df.iloc[0]["expected1"]

# Student's answer and faculty's expected answer for question
stu_ans2 = df.iloc[0]["student2"]
unique_words = set(stu_ans2.split())
stu_ans2 = ' '.join(unique_words)
exp_ans2 = df.iloc[0]["expected2"]

# Student's answer and faculty's expected answer for question 1
stu_ans3 = df.iloc[0]["student3"]
unique_words = set(stu_ans3.split())
stu_ans3 = ' '.join(unique_words)
exp_ans3 = df.iloc[0]["expected3"]

# Evaluate answers and get marks scored by students
question1_marks = round(get_score(exp_ans1, stu_ans1,max_mark), 2)
question2_marks = round(get_score(exp_ans2, stu_ans2,max_mark), 2)
question3_marks = round(get_score(exp_ans3, stu_ans3,max_mark), 2)

# Total marks
total = question1_marks + question2_marks + question3_marks
```

```
    if total < 0:
        total = 0
    # Return marks scored to application

    return total
print("marrk:")
print(evaluate())
```

evaluate-test.php

```
<?php
set_time_limit(600);
include('student-session.php');

$student_ID = $login_id;
$test_ID = $_SESSION['test_ID'];
$answer1 = $_REQUEST['answer1'];
$answer2 = $_REQUEST['answer2'];
$answer3 = $_REQUEST['answer3'];

$sql1 = "INSERT INTO answers VALUES ('$test_ID', '$student_ID', '$answer1', '$answer2', '$answer3')";
$result1 = mysqli_query($db, $sql1);

$exp_ans1 = $_SESSION['question1_ea'];
$exp_ans2 = $_SESSION['question2_ea'];
$exp_ans3 = $_SESSION['question3_ea'];
```

```
$user_CSV[0] = array('student1', 'expected1', 'student2', 'expected2', 'student3', 'expected3');
$user_CSV[1] = array($answer1, $exp_ans1, $answer2, $exp_ans2, $answer3, $exp_ans3);

$fp = fopen('answers.csv', 'wb');
foreach ($user_CSV as $line) {
    fputcsv($fp, $line, ',');
}
fclose($fp);

$command = escapeshellcmd('python evalaute.py answers.csv');
echo $command;
$score = exec($command);

$stmt2 = $db->prepare("INSERT INTO scores (test_ID, student_ID, score) VALUES (?, ?, ?)");
$stmt2->bind_param("sss", $test_ID, $student_ID, $score);
$stmt2->execute();

$sql2 = "INSERT INTO scores VALUES ('$test_ID', '$student_ID', '$score')";
$result2 = mysqli_query($db, $sql2);

echo '<script>alert("Your test has been evaluated.")</script>';
header("location: student-home.php");
```

CHAPTER 12

REFERENCES

[1] Jamin Rahman Jim , Md Apon Riaz Talukder , Partha Malakar , *Recent advancements and challenges of NLP-based sentiment Analysis : A state-of-the-art review* (2023)

[2]Andrew Hunt, David Thomas, *The Pragmatic Programmer: From Journeyman to Master*, Pearson India, 1st Edition (2008).

[3]Ken Schwaber, Mike Beedle, *Agile Software Development with Scrum*, Pearson (2008).

[4]Lisa Crispin, Janet Gregory, *Agile Testing: A Practical Guide for Testers and Agile Teams*, Addison Wesley Professional, 1st Edition (2008).

[5]Roger S. Pressman, *Software Engineering* (1994)

[6]Rahman and Siddiqui ,*NLP-based Automatic Answer Script Evaluation* (2018)

[7]Vinal Bhagaria, Mohit Badve, *An Intelligent System for Evaluation of Descriptive Answers* (2020)

[8]MF Bashir, Hamza Arshad ,*Subjective Answers Evaluation Using Machine Learning and Natural Language Processing* (2021)

[9]Gary B. Shelly, Harry J. Rosenblatt, *System Analysis and Design*, 2009.

CHAPTER 14

SCREENSHOTS

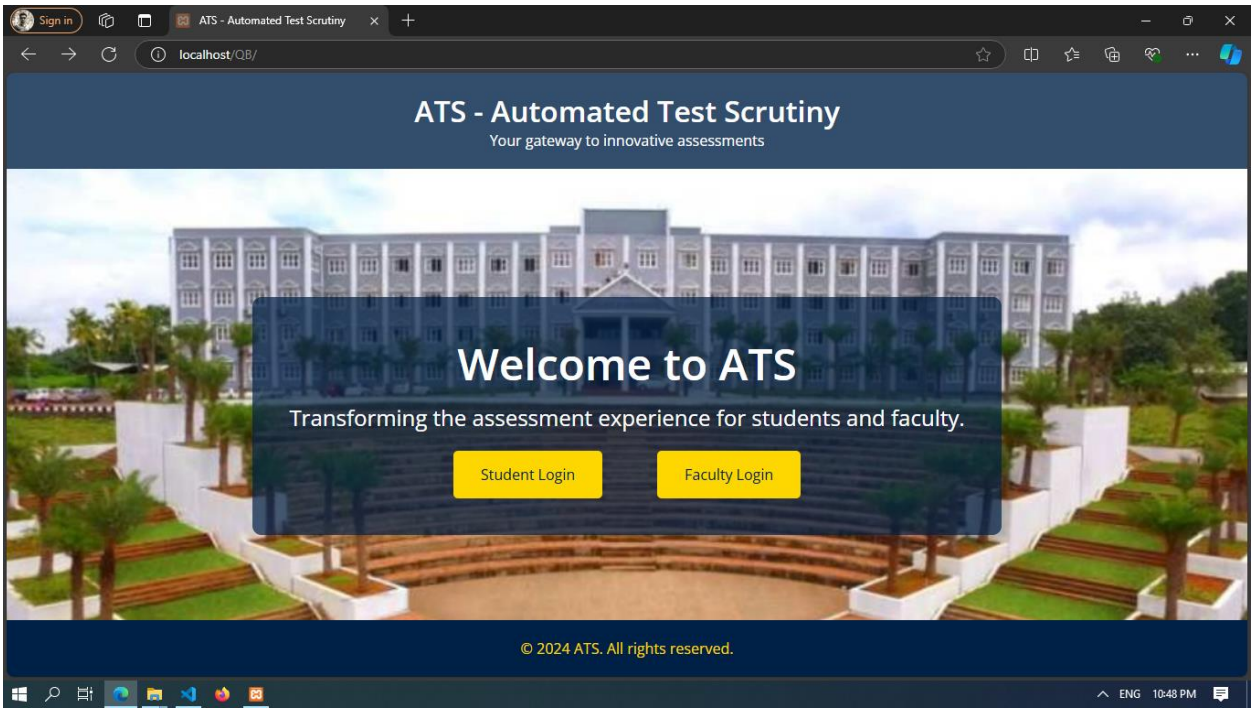

Fig 13.1 Index Page

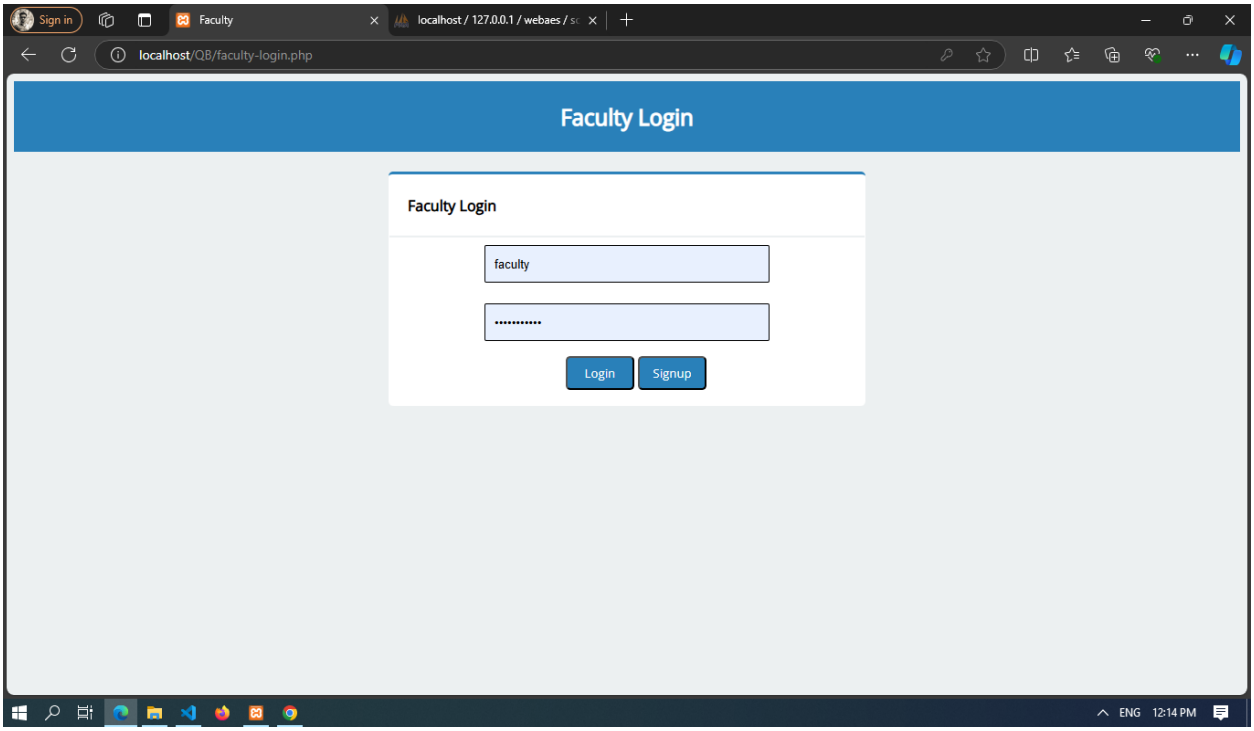

Fig 13.2 Faculty Login

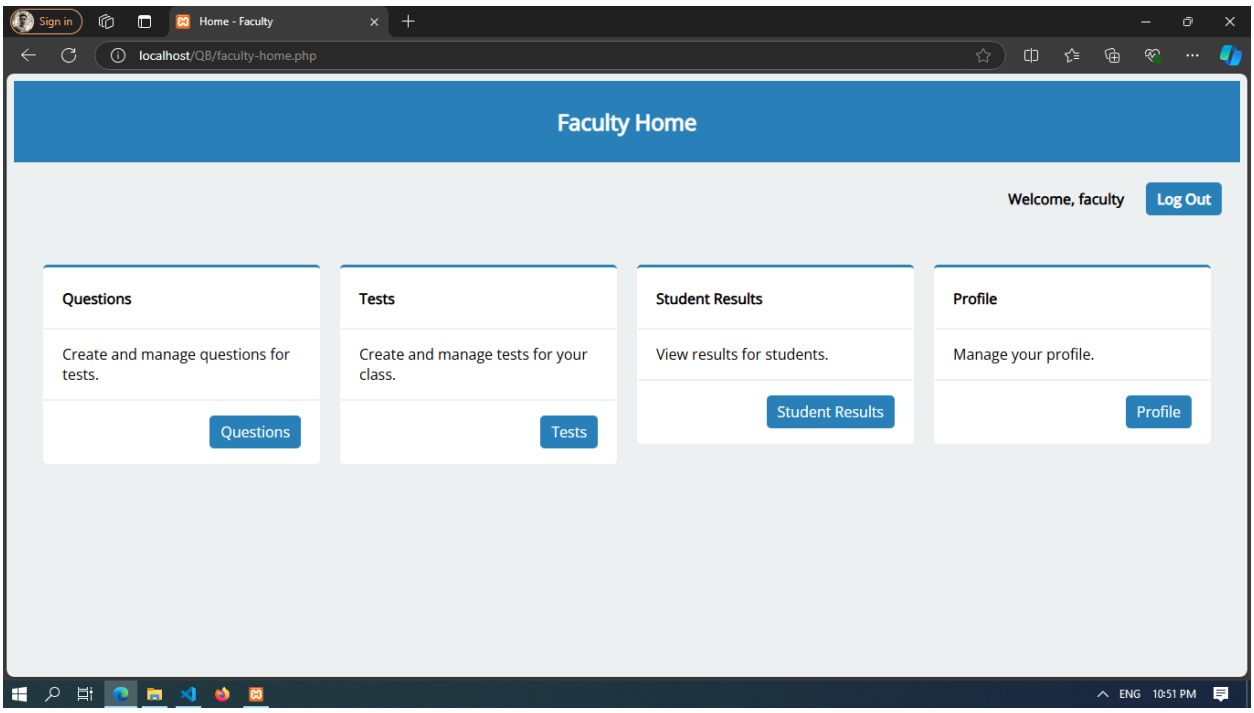

Fig 13.3 Faculty Home

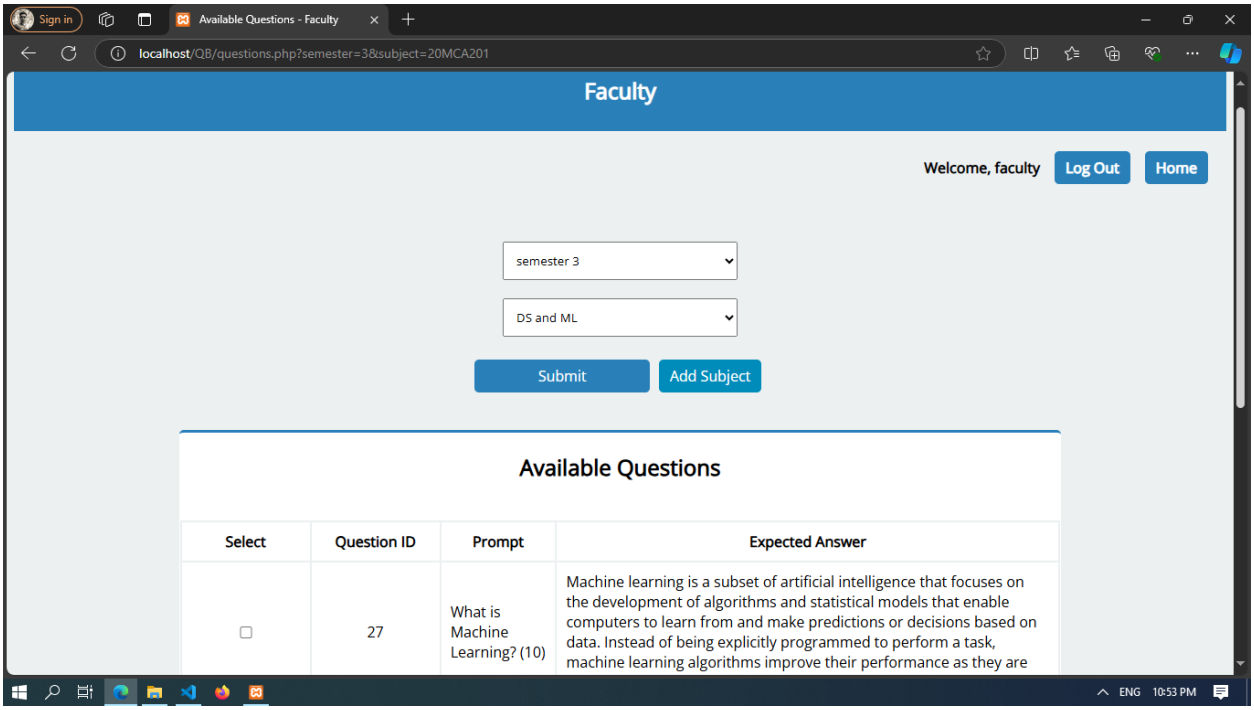

Fig 13.4 Faculty Question

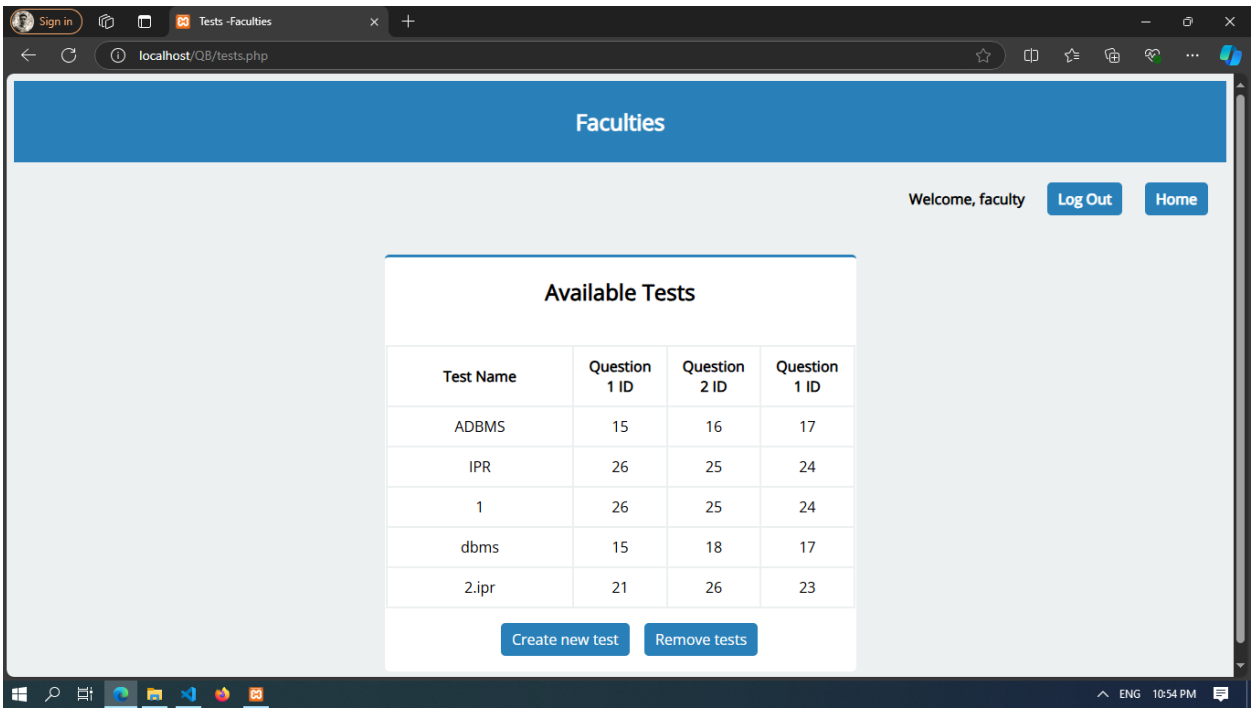

Fig 13.5 Faculty Test

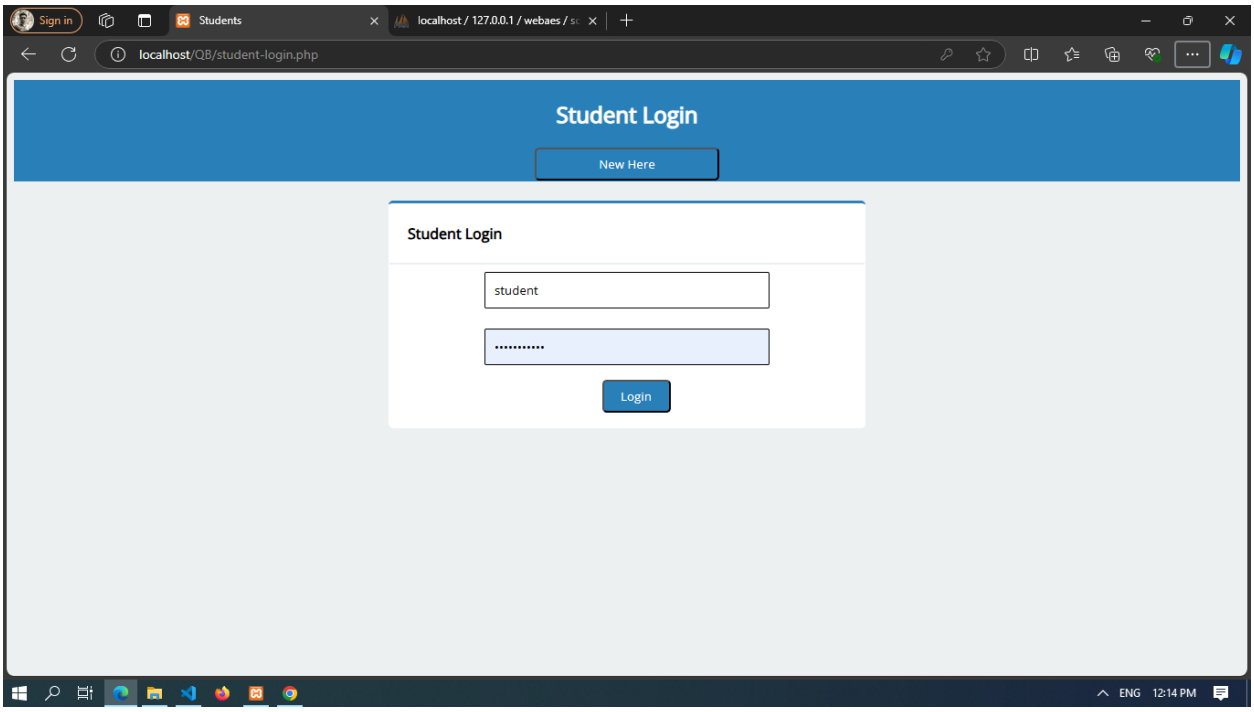

Fig 13.6 Student Login

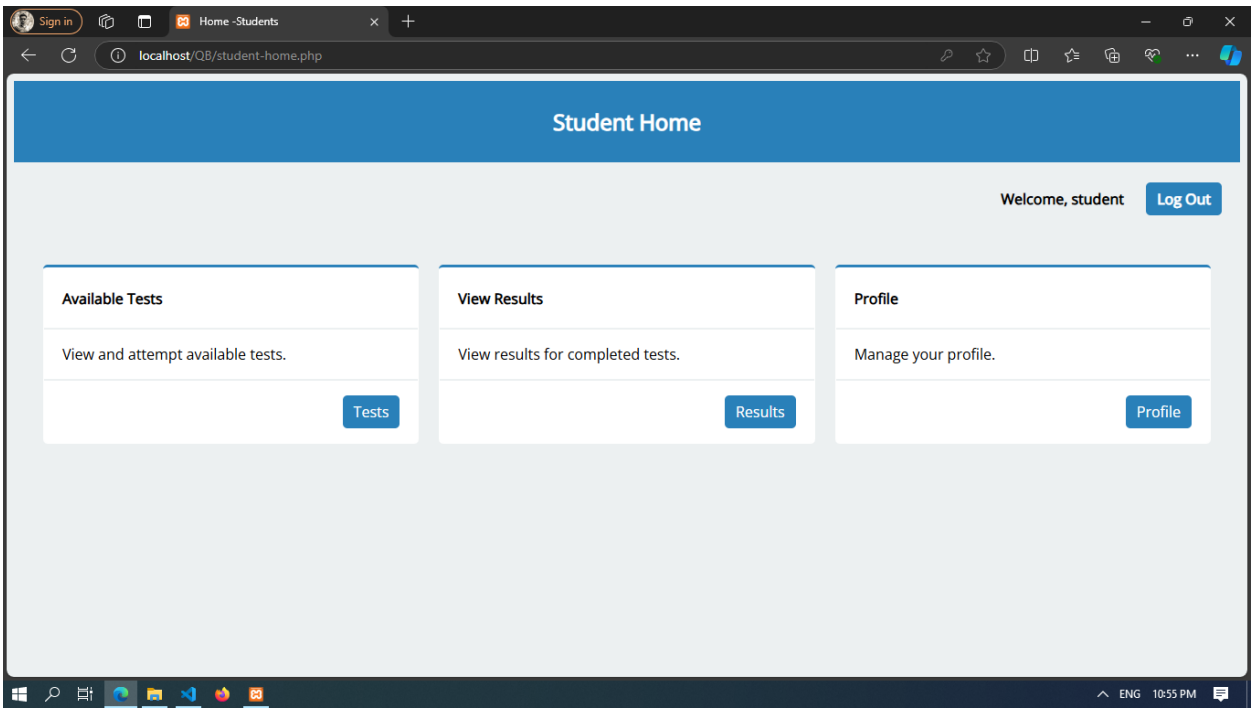

Fig 13.7 Student Home

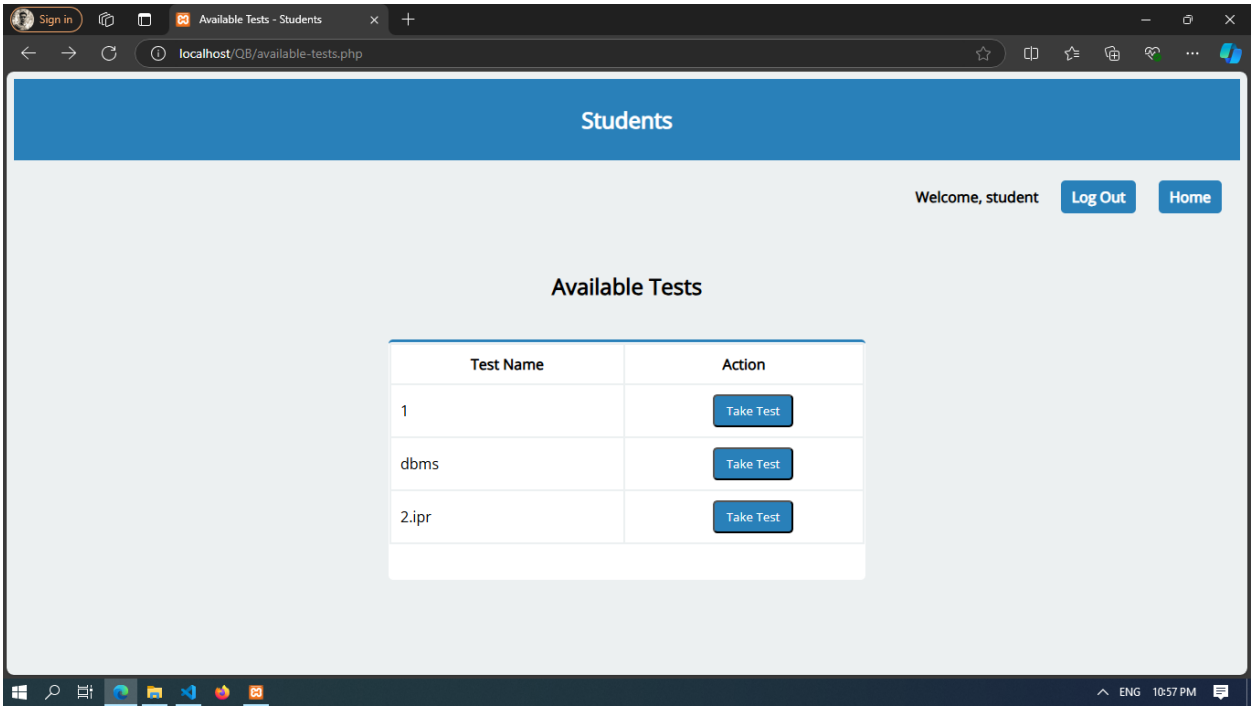

Fig 13.8 Student Tests

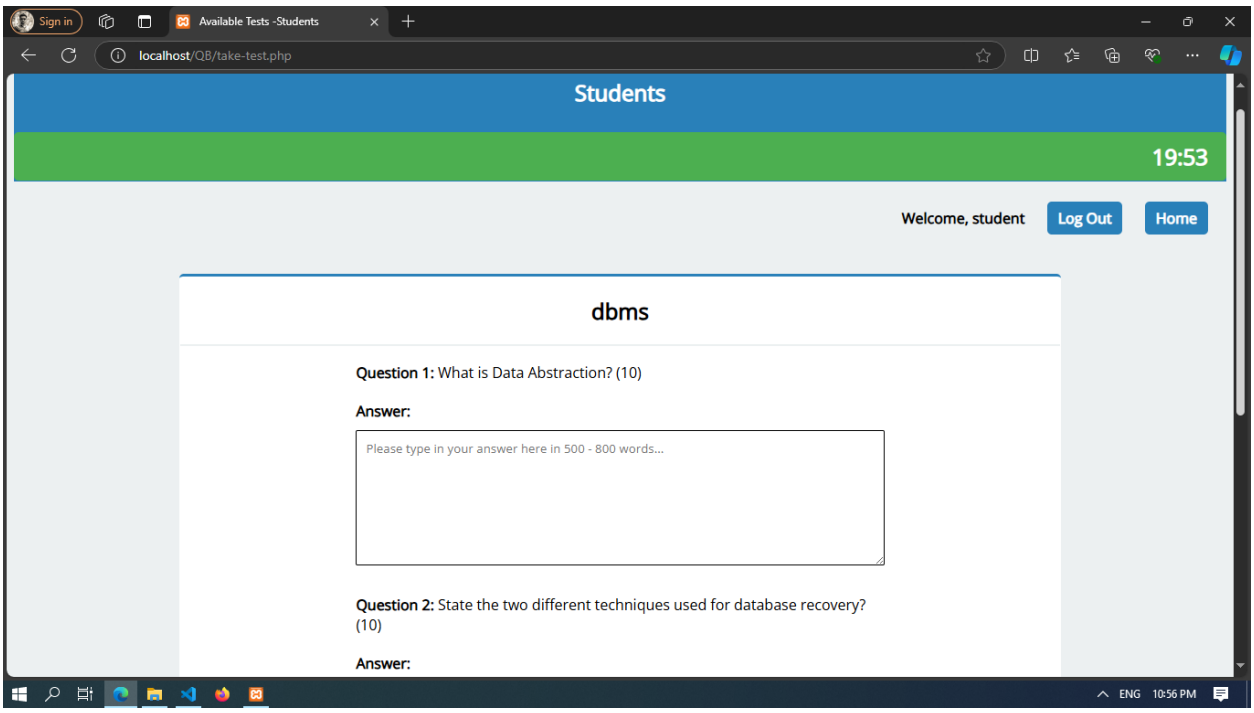

Fig 13.9 Student Test

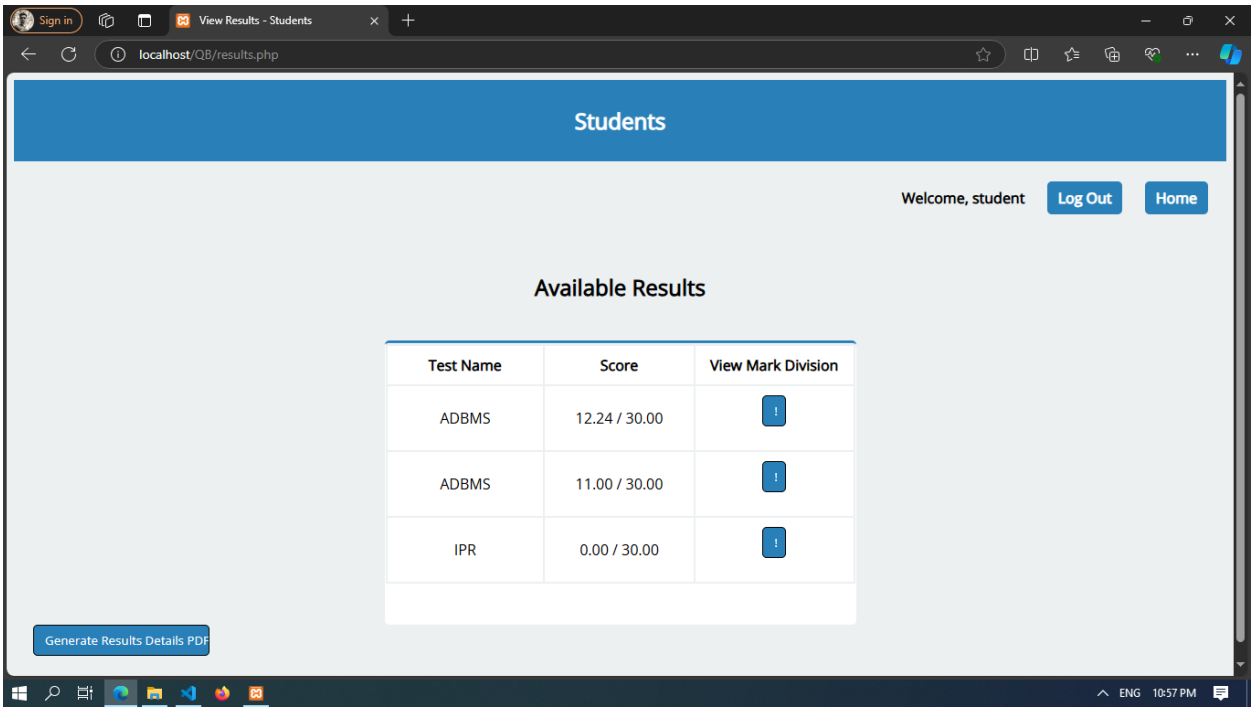

Fig 13.11 Student Result