PROJECT REPORT

On

PAPERPRO – QUESTION GENERATOR

Submitted in partial fulfilment for the award of the degree

of

Master of Computer Applications

By

ABITHAMOL C F (MLM23MCA-2003)

Under the Guidance of

KUKKUMOL THOMAS
(Assistant Professor, Department of Computer Applications)

DEPARTMENT OF COMPUTER APPLICATIONS

MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR

(Affiliated to APJ Abdul Kalam Technological University)

APRIL 2025

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 education,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, lifelong learning and research, thereby making meaningful contributions to 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

APRIL 2025

CERTIFICATE

*This is to certify that the Project titled "**PAPERPRO – QUESTION GENERATOR**" is the bonafide record of the work done by **ABITHAMOL C F (MLM23MCA-2003)** of **Master of Computer Applications** towards the partial fulfilment of the requirement for the award of the **DEGREE OF MASTER OF COMPUTER APPLICATIONS by APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY**,*
during the academic year 2023-25.

Project Guide
Ms. Kukkumol Thomas
Assistant Professor
Dept. Of Computer Applications

Head of the Department
Ms. Divya S.B.
Associate Professor
Dept. Of Computer Applications

Project Coordinator
Ms. Banu Sumayya S
Assistant Professor
Dept. Of Computer Applications

External Examiner

ACKNOWLEDGEMENT

I am greatly indebted to the authorities of Mangalam College of Engineering for providing the necessary facilities to successfully complete my Project on the topic “AI Receptionist”.

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 her invaluable help and support which helped me a lot in successfully completing this Project work.

I express my gratitude to my Internal Guide, **Ms. Kukkumol Thomas**, Assistant professor, Department of Computer Applications for the suggestions and encouragement which helped in the successful completion of our Project.

Furthermore, I would like to acknowledge with much appreciation the crucial role of the faculties especially Project coordinator, **Ms. Banu Sumayya S** Assistant professor, Department of Computer Applications, who gave the permission to use all the required equipment and the necessary resources to complete the presentation & report.

Finally, I would like to express my heartfelt thanks to my parents who were very supportive both financially and mentally and for their encouragement to achieve my goal.

ABITHAMOL C F
(MLM23MCA-2003)

ABSTRACT

The Automatic Question Paper Generation System is an AI-driven platform that automates the process of question paper creation using Natural Language Processing (NLP) and Machine Learning (ML) techniques. The system allows teachers to upload study materials, from which questions are generated automatically using intelligent algorithms. It supports multiple user roles, including Admin, HOD, Teacher, and Student, ensuring secure access and role-based functionalities. Students can attempt tests in both MCQ and subjective formats, with automated evaluation for objective questions and manual review for subjective responses. By reducing manual effort, ensuring question relevance, and improving assessment efficiency, this system enhances the educational evaluation process and demonstrates the potential.

CONTENTS

SLNO	TITLE	PAGE NO
1	**INTRODUCTION**	**1**
1.1	BACKGROUND	1
1.2	INTRODUCTION	1
1.3	PROBLEM STATEMENT	2
1.4	MOTIVATION	2
1.5	SCOPE	3
2	**LITERATURE REVIEW**	**4**
3	**PROPOSED SYSTEM**	**11**
4	**METHODOLOGY**	**13**
5	**SYSTEM ARCHITECTURE**	**15**
6	**MODULES**	**19**
7	**DIAGRAMS**	**21**
7.1	DFD	21
7.1.1	LEVEL 0 DFD	21
7.1.2	LEVEL 1 DFD - ADMIN	22
7.1.3	LEVEL 1 DFD - HOD	22
7.1.4	LEVEL 1 DFD- TEACHER	23
7.2	**ACTIVITY DIAGRAM**	24
7.3	**CLASS DIAGRAM**	25
7.4	**USE CASE DIAGRAM**	**26**
8	**TESTING**	**28**
9	**ADVANTAGES AND DISADVANTAGES**	**29**
10	**RESULTS AND CONCLUSIONS**	**30**
11	**APPENDICES**	**33**
12	**REFERENCES**	**82**

LIST OF FIGURES

FIG NO	FIG NAME	PAGE NO
.7.1.1	LEVEL 0 DFD	21
7.1.2	LEVEL 1 DFD-ADMIN	22
7.1.3	LEVEL 1 DFD-HOD	22
7.1.4	LEVEL 1 DFD-TEACHER	23
7.3	CLASS DIAGRAM	25
.7.4	USE CASE DIAGRAM	26

LIST OF ABBRIEVIATIONS

ABBREVIATION	FULL FORM
ASR	Automatic Speech Recognition
HMM	Hidden Markov Model
GMM	Gaussian Mixture Model
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory
NLP	Natural Language Processing

CHAPTER-1
INTRODUCTION

1.1 BACKGROUND

The PaperPro – Question Generator project is an innovative system which is designed to streamline the process of generating exam questions by utilizing advanced technologies like Natural Language Processing (NLP) and Machine Learning. It provides a user-friendly interface for educators for uploading their lecture notes pdf, which is then used to serve as the foundation for intelligent question generation. This system uses a T5-base model to generate context-aware questions, it provides both multiple-choice and subjective questions, by extracting relevant entities and distractors. Furthermore, a BERT (Bidirectional Encoder Representations from Transformers)-based evaluator to assesses the quality of generated questions, ensuring high relevance and coherence.

Apart from question generation, the system also facilitates online exam creation and administration, enabling remote assessments. It incorporates the predictive analytics to analyze students' exam performance, identifying those at risk of academic difficulties and allowing timely intervention. This project aims to reduce manual effort in exam creation, improve the question quality, and provide valuable insights for enhancing the teaching and learning experience.

1.2 INTRODUCTION

The PaperPro – Question Generator is a valuable system designed to revolutionize the traditional exam preparation process by automating the generation of high-quality examination questions. By using Natural Language Processing (NLP) and Machine Learning (ML) techniques, the system extracts information from the uploaded educational materials to create contextually relevant subjective and objective questions. By integrating an intelligent T5-based Question Generator, the platform ensures that generated questions align with the given content, maintaining relevance and accuracy. Additionally, a BERT-based Question Evaluator ranks and filters questions based on quality, ensuring that only the most valuable questions are selected for assessments.

Beyond question generation, the system also supports online exam administration, allowing educators to efficiently create and manage exams remotely. With built-in predictive analytics, the platform analyzes students' exam performance to identify those at risk and provide timely academic interventions. This automation not only reduces the manual effort required for exam

preparation but also enhances the overall learning experience by offering personalized and well-structured assessments. The project aims to modernize education by making exam creation more efficient, data-driven, and adaptive to individual learning needs.

1.3 PROBLEM STATEMENT

The traditional process of question paper generation is highly time-consuming, labor-intensive, and prone to biases or inconsistencies. Educators often spend significant amounts of time crafting examination questions manually, ensuring alignment with curriculum standards and maintaining question difficulty balance. This manual effort can lead to redundancies, lack of variety, and errors in question selection, which may impact the effectiveness of assessments. Additionally, maintaining a diverse set of high-quality questions requires continuous effort, which can be challenging for institutions with large student populations and frequent assessments.

Another major challenge is ensuring fairness and quality in question selection. Manually generated question papers may unintentionally include repetitive, vague, or ambiguous questions that fail to effectively assess students' knowledge. Furthermore, the difficulty level of questions may not always be well-distributed, leading to either an overly simple or excessively challenging examination. This inconsistency can impact students' academic performance and fail to provide an accurate measure of their understanding. Security concerns, such as question leaks and unauthorized access, also pose risks to the integrity of manually created exams.

To address these challenges, an Automatic Question Paper Generator is needed to streamline the process using Machine Learning and Natural Language Processing (NLP). By automating question generation, filtering, and evaluation, the system ensures high-quality, diverse, and contextually relevant questions that align with the provided educational content. Additionally, the system can analyze student performance data to enhance assessment strategies, making examinations more effective, scalable, and fair. This solution significantly reduces the manual workload for educators while improving the reliability and security of assessments.

1.4 MOTIVATION

The motivation behind developing an PaperPro - Question Generator stems from the need to modernize and streamline the traditional examination process. In the current educational landscape, instructors and exam committees spend a considerable amount of time crafting

question papers that align with syllabi and learning objectives. This manual process is often tedious, repetitive, and susceptible to human errors, making it inefficient for institutions that conduct frequent assessments. By automating question generation, educators can focus more on teaching and student engagement, rather than spending excessive time on exam preparation.

Another key motivation is the lack of diversity and fairness in manually created question papers. Traditional methods may inadvertently introduce biases, where certain topics are overrepresented while others are overlooked. Additionally, manually selecting questions of varying difficulty levels is challenging, often leading to exams that are either too easy or too difficult. An automated system that leverages Natural Language Processing (NLP) and Machine Learning (ML) can ensure a balanced distribution of question difficulty, improving the overall fairness and effectiveness of assessments.

The project is also motivated by the increasing demand for scalable and secure online assessment solutions. With the rise of digital learning platforms and remote education, institutions require automated tools to generate exams efficiently and securely. Manually created question papers are vulnerable to security risks, such as leakage or unauthorized distribution. By implementing an AI-driven question paper generator, institutions can enhance security measures while ensuring that unique and high-quality questions are generated for each exam session.

Lastly, integrating predictive analytics into the system provides an additional advantage by identifying students at risk based on their performance trends. This allows educators to take proactive steps in personalized learning interventions, ultimately improving student success rates. By automating the exam creation process and providing valuable insights into student learning, this project contributes to a more efficient, data-driven, and student-centric educational system.

1.5 SCOPE

The scope of the Automatic Question Paper Generator is to simplify and improve the process of creating question papers using advanced technology. This system is designed for schools, colleges, universities, and online learning platforms to help educators generate high-quality exam questions quickly and efficiently. It ensures that questions are well-structured, relevant to the syllabus, and balanced in difficulty level.

The system uses Natural Language Processing (NLP) and Machine Learning (ML) to generate different types of questions, such as multiple-choice, fill-in-the-blanks, and descriptive

questions, based on uploaded study materials. Additionally, it evaluates the quality of generated questions, ensuring that only the best ones are selected for the final exam. Educators can also customize the type and difficulty of questions as per their requirements.

Beyond question generation, the system also supports online exam creation and administration, making it useful for remote learning and digital education platforms. It enhances security by generating unique question sets for different students, reducing the chances of question leaks. Moreover, it helps analyze students' performance, identifying those who may need extra support and enabling educators to take timely action.

In summary, this project reduces the workload for educators, improves the quality of assessments, ensures fairness in exams, and enhances learning outcomes. It is a valuable tool for modern education, making exam preparation faster, smarter, and more effective.

CHAPTER-2

LITERATURE REVIEW

2.1 Automatic Question Generation using NLP and Deep Learning

Author(s): J. Sun et al.

Publication:IEEE Transactions on Learning Technologies, 2020

This paper explores the use of Natural Language Processing (NLP) and deep learning techniques for automatic question generation. The authors utilize the T5 transformer model, a state-of-the-art deep learning model, to generate questions from textual data. The study highlights the importance of contextual understanding, ensuring that the generated questions align with the given content while maintaining coherence and clarity. To further enhance the quality of the generated questions, a BERT-based evaluator is integrated, which ranks and filters questions to ensure relevance and accuracy.

The proposed approach significantly improves the relevance, coherence, and quality of generated questions, making it highly suitable for educational applications such as automated assessments and tutoring systems. The use of deep learning models allows the system to adapt to different types of text and generate diverse questions with minimal human intervention. However, the study identifies certain challenges, such as the requirement for large training datasets to fine-tune the model effectively. Additionally, domain-specific question generation remains a concern, as the model may not always generate highly specialized questions without additional fine-tuning.

Overall, this research contributes to the advancement of AI-driven question generation, demonstrating that NLP and deep learning models can significantly enhance automated assessments. The findings suggest that with further refinement and domain adaptation, this approach could be widely adopted in intelligent tutoring systems, e-learning platforms, and automated examination tools to improve educational efficiency and accessibility.

2.2 Dual Learning for Visual Question Generation

Author(s): Xing Xu et al.

Publication: IEEE International Conference on Multimedia and Expo (ICME), 2018.This research introduces a novel dual learning framework designed to enhance question generation from visual inputs, specifically images. The core idea of the paper is to train two models simultaneously: one model generates questions based on an image, while the other provides answers to those questions. The dual learning framework incorporates reinforcement learning, allowing the models to improve over time by self-correcting their outputs without requiring additional human annotations. This approach addresses the growing demand for AI systems capable of interacting with visual data and generating contextually appropriate questions. The generated questions have potential applications in education, interactive media, and accessibility.

The proposed methodology employs a dual learning framework where one model generates questions about an image and another model answers those questions. These models work in a feedback loop, with reinforcement learning helping to improve both the question generation and answering processes. The system is trained on existing datasets and refined through continuous feedback. This allows the framework to generate meaningful questions directly from images, bypassing the need for text-based input, and enables the models to improve based on the quality of their interactions.

The primary advantage of this system is that it improves itself through reinforcement learning, which eliminates the need for additional human-labeled data. Furthermore, it can generate meaningful questions directly from images without relying on text-based input. However, a significant limitation of this method is the need for substantial computational resources due to the reinforcement learning framework. Additionally, the system may struggle with generating questions for images that are complex or highly abstract.

2.3 Thematic Question Generation over Knowledge Bases

Author(s): Tanguy Raynaud et al.

Publication: IEEE/WIC/ACM International Conference on Web Intelligence (WI), 2018.

This paper introduces an automated system for generating multiple-choice questions (MCQs) from structured knowledge bases, such as knowledge graphs. The authors employ a template-based approach to extract topic-relevant information from the knowledge base and generate questions by applying predefined templates. This system ensures that the generated MCQs remain relevant to a given topic while containing well-structured distractors (incorrect answers) to enhance the difficulty level of the questions. The automation of this process enables the creation of high-quality educational content without the need for human intervention.

The authors highlight the dynamic nature of their approach, which allows for generating MCQs that are contextually relevant to any topic within the knowledge base. By using a structured knowledge graph, the system ensures that questions reflect accurate relationships and details, enhancing the learning experience. This automated process significantly reduces the time and effort involved in creating assessments, making it an efficient solution for educational systems. However, the reliance on predefined templates may limit the variety and complexity of the generated questions for more specialized topics.

The methodology of the proposed system involves a template-based approach for generating MCQs from structured knowledge bases. The first step involves identifying topic boundaries within the knowledge base to ensure that the questions are contextually relevant. Predefined templates are then applied to generate questions corresponding to the identified topics. The system also ensures that distractors are well-structured, making the MCQs more challenging. This dynamic approach allows for the automatic generation of MCQs for any topic, enhancing its scalability and efficiency for educational purposes.

A major advantage of the proposed system is its ability to generate high-quality MCQs with well-structured distractors. Additionally, it can dynamically generate questions from existing knowledge bases, making it a scalable solution for educational applications. However, the approach has some limitations, such as its reliance on predefined question templates, which restrict the diversity of the generated questions. Furthermore, the performance of the system is contingent upon the quality and completeness of the underlying knowledge base.

2.4 Deep Residual Learning for Image Recognition.

Author(s): Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun.

Publications: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

This groundbreaking paper introduces Residual Networks (ResNet), a deep learning architecture that addresses the challenge of training very deep neural networks. As the depth of a neural network increases, it often suffers from performance degradation, primarily due to the vanishing gradient problem. To solve this, the authors propose the concept of residual learning, where the network learns residual mappings instead of direct mappings, making it easier to optimize. This innovation allows the network to skip certain layers through shortcut connections, enabling it to maintain high performance even as the network depth increases. The ResNet architecture has significantly impacted image recognition, allowing for the effective training of much deeper and more complex models, achieving state-of-the-art performance in image classification tasks.

The introduction of residual learning in ResNet has revolutionized the design of deep neural networks by overcoming the challenges associated with training very deep models. This architecture allows for the effective training of networks with hundreds or even thousands of layers, which significantly improves accuracy in image classification tasks. By addressing the vanishing gradient problem, ResNet has set a new benchmark in the field of computer vision, influencing subsequent advancements and applications in deep learning.

The core methodology behind ResNet is residual learning, which involves introducing shortcut connections that bypass one or more layers of the network. This technique prevents the vanishing gradient problem and allows the network to be trained more effectively, even as its depth increases. By stacking many residual blocks, deep networks can now be trained with high accuracy, making it possible to build models with hundreds or even thousands of layers. The authors demonstrate that this approach leads to breakthrough performance in image classification, making ResNet a critical advancement in deep learning for computer vision tasks.

The ResNet architecture enables the creation of ultra-deep networks that would otherwise be difficult or impossible to train. Its ability to maintain performance while scaling up the depth of the network has led to its widespread adoption in the field of computer vision. ResNet has set a new standard for deep learning architectures, influencing both academic research and industrial applications in image recognition.

2.5 ImageNet Large Scale Visual Recognition Challenge.

Author(s): Olga Russakovsky et al.

Publication: Springer International Journal of Computer Vision (IJCV), 2015.

This paper provides a comprehensive analysis of the ImageNet dataset, a key resource for image recognition tasks, and introduces the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a prominent annual competition for evaluating machine learning models on image classification tasks. The ImageNet dataset consists of millions of labeled images across 1,000 categories, making it a critical benchmark in the field of deep learning and computer vision. The paper discusses the significance of the ILSVRC in pushing the boundaries of image recognition performance and highlights how it has fostered the development of several high-performing deep learning architectures, including ResNet, VGG, and Inception. These advances have significantly influenced the direction of research in the field and contributed to the evolution of computer vision models. The dataset and competition have played an essential role in shaping the landscape of modern image recognition by providing a diverse and expansive set of data for model training and evaluation.

The ImageNet dataset and the ILSVRC competition have significantly advanced image recognition research by providing a vast, labeled dataset and establishing benchmarks that have driven the development of cutting-edge deep learning architectures.

The methodology of the ILSVRC involves testing deep neural networks trained on the ImageNet dataset to evaluate their performance on tasks like object classification and localization. This competition has been crucial in advancing deep learning by providing both a high-quality dataset and a rigorous framework for assessing models. Over time, various architectures, such as ResNet, VGG, and Inception, have been developed and tested using this dataset, continuously improving the accuracy and efficiency of image recognition models. The challenge has provided invaluable insights into the strengths and weaknesses of various deep learning approaches, spurring continued innovation in the field.

Despite its contributions, the ImageNet dataset has some inherent limitations. It contains biases that may affect the real-world applicability of models trained on it, and certain categories within the dataset are underrepresented or lack sufficient data, which can lead to reduced performance in those specific areas. However, the impact of ImageNet and the ILSVRC on deep learning and computer vision remains profound, serving as a foundational resource for both researchers and practitioners in the field.

2.6 Universal Dependency Parsing from Scratch.

Authors: Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning.

Publication: Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies.

This paper introduces a comprehensive neural pipeline system developed by Stanford for the CoNLL 2018 Universal Dependencies (UD) Shared Task. The system is designed to process raw text across multiple stages, including tokenization, sentence segmentation, part-of-speech (POS) tagging, lemmatization, and dependency parsing. By using a biaffine classifier that jointly predicts both POS and morphological features, the system improves the consistency and accuracy of these predictions. Additionally, the system outperformed submissions on low-resource treebank categories, making it particularly robust in multilingual and underrepresented language settings.

The methodology behind the system is based on an end-to-end approach where raw text is transformed into structured linguistic information at various stages. This includes a novel model for dependency parsing that effectively addresses word order by extending the parser to model linearization. By doing so, the system improves syntactic analysis for different languages, handling their diverse linguistic structures. The pipeline ensures that all stages, from tokenization to dependency parsing, are processed seamlessly and efficiently, leading to improved performance and reduced need for manual intervention.

One of the key strengths of the system is its ability to handle rare word forms and sequences, achieved by incorporating a lemmatizer with an edit classifier. This feature ensures that the system remains robust even in cases where unusual word forms appear in the input text. The pipeline's ability to deal with these challenges was a major factor in its success on low-resource treebank categories, where data scarcity usually hampers performance. This made the system a particularly powerful tool for processing multilingual corpora with a variety of syntactic complexities.

While the system demonstrated significant performance, an initial bug in the early stages affected its results. However, once the issue was addressed, the system would have ranked highly according to official evaluation metrics. Despite its comprehensive nature and strong performance, the pipeline is resource-intensive, requiring substantial computational power due to the multiple stages of processing involved.

CHAPTER-3

PROPOSED SYSTEM

The PaperPro – Question Genarator is an automated solution designed to streamline the process of creating question papers for educational institutions. By eliminating the manual effort involved in question paper creation, this system ensures high-quality, error-free, and diverse question papers are generated based on customizable parameters. It aims to support various question types (multiple choice, short answer, descriptive) and maintain balance across different topics and difficulty levels.

The AQPGS will be a web-based application, allowing instructors to input parameters such as subject, topic, number of questions, and difficulty levels, and automatically generate the question paper. This system will significantly reduce the time and effort traditionally spent on paper creation, providing an efficient and reliable alternative to manual methods.

Features of the System

1. **Customizable Question Paper Parameters**: Instructors can set various parameters like the number of questions, topics, difficulty level, and question type (MCQs, short answer, long answer) to tailor the paper to specific requirements.
2. **Diverse Question Formats**: Supports different question formats such as Multiple Choice Questions (MCQs), short answer questions, and descriptive/essay-type questions.
3. **Balanced Question Distribution**: Ensures a fair distribution of questions across topics and difficulty levels, maintaining the integrity and variety of the question paper.
4. **Randomized Question Selection**: The system randomly selects questions from the question bank to create unique and varied question papers every time.
5. **User-Friendly Interface**: The system features an intuitive UI, allowing users to easily input parameters and generate question papers with minimal effort.
6. **Secure and Scalable**: The system will be built on secure, scalable technologies, ensuring the solution can handle multiple users and large question databases efficiently.

7. **Output Formats**: After generating the question paper, the system will provide options to download the paper in multiple formats, such as PDF or DOC, for easy distribution and printing.
8. **Question Bank Management**: Admins can manage and update the question bank by adding, editing, or deleting questions across different subjects and categories.

Benefits of the System

1. **Time-Saving**: Automates the question paper creation process, drastically reducing the time spent by instructors and administrators in crafting papers manually.
2. **Error-Free Generation**: Minimizes human errors, ensuring that the generated question papers are error-free, consistent, and well-structured.
3. **Customization**: Offers flexibility for instructors to design question papers based on specific course requirements, ensuring that the assessments are aligned with the curriculum.
4. **Enhanced Security**: By randomizing questions and ensuring unique papers each time, the system improves exam security by reducing the risk of paper leakage or cheating.
5. **Balanced Assessment**: Guarantees that the generated question papers cover all necessary topics and have a balanced mix of easy, medium, and difficult questions.
6. **Cost-Efficient**: Reduces administrative overhead by eliminating the need for manual intervention, thus saving institutional resources and costs.
7. **Improved Quality of Assessments**: With automated question selection and balanced difficulty levels, the system ensures the quality of assessments remains consistent and fair for all students.
8. **Scalability**: The web-based nature of the system allows it to be easily scaled to accommodate different institutions, subjects, and large question banks.
9. **Convenient Output**: Provides downloadable question papers in standardized formats, making it easier for instructors to distribute the papers for exams.

CHAPTER 4

METHODOLOGY

The development of the PaperPro – Question Genarator follows a structured methodology to ensure accuracy, efficiency, and reliability in automating the question paper creation process. The project begins with a requirement analysis phase, where the functional and non-functional needs of the system are identified. Educators and administrators are consulted to understand key requirements, such as customizable question paper generation based on subjects, difficulty levels, and question types. The system also ensures secure storage of questions in a well-organized question bank.

Following this, the system design phase is carried out, where the architecture and modules of the system are planned. The system consists of a frontend user interface developed using HTML, CSS, JavaScript, and PHP, allowing instructors to input parameters for generating question papers. The backend logic, implemented using PHP and MySQL, handles question selection, distribution, and formatting. The database is structured to store questions categorized by subject, topic, difficulty level, and type (MCQs, short answer, or descriptive). A key aspect of the design is the algorithm that ensures randomized question selection while maintaining a balanced distribution of topics and difficulty levels, preventing repetition and ensuring fairness in the generated papers.

The next phase involves data collection and preprocessing, where a comprehensive question bank is built using questions from textbooks, past exams, and manually entered data. Each question is tagged with metadata to ensure proper classification, and duplicate questions are removed to maintain uniqueness and quality. After data preparation, the implementation phase begins, involving the actual development of system components. The frontend is designed for ease of use, allowing instructors to enter required parameters with minimal effort. The backend ensures that question retrieval, randomization, and paper formatting are executed efficiently. The database integration ensures smooth storage and retrieval of questions.

The question paper generation algorithm follows a systematic approach. First, instructors provide input parameters such as subject, number of questions, and difficulty level. The system then retrieves questions that match these criteria from the database. The selected questions undergo randomization and balanced distribution to ensure that difficulty levels are proportionately represented. After selection, the system validates the paper to prevent

duplication and ensure that total marks align with exam requirements. Finally, the questions are formatted and structured in a standardized output format, which can be downloaded as a PDF or DOC file for easy printing or digital distribution.

Once the system is implemented, it undergoes testing and debugging to ensure reliability. Unit testing is conducted on individual components such as the question selection module and the user interface, while integration testing verifies the smooth interaction between system components. User acceptance testing (UAT) is carried out by educators to ensure the system meets their expectations. Any errors or inconsistencies identified during testing are fixed to enhance system performance.

The final phase involves deployment and maintenance, where the system is hosted on a web server, making it accessible to educators and administrators. Continuous monitoring is performed to ensure smooth operation, and periodic updates are introduced based on user feedback to improve functionality. By following this structured methodology, the AQPGS effectively automates the question paper generation process, making it time-efficient, error-free, and highly customizable, ultimately benefiting both educators and students.

CHAPTER-5

SYSTEM ARCHITECTURE

The Automatic Question Paper Generation System (AQPGS) is designed to automate the creation of question papers based on specific criteria provided by educators. The system architecture follows a structured design to ensure efficiency, scalability, and ease of use. It consists of different layers and modules that work together to provide a seamless experience for users. The architecture is primarily divided into frontend (user interface), backend (logic processing), and database (data storage) components, ensuring smooth interaction and efficient question paper generation.

Overview of System Architecture

The system follows a three-tier architecture, which includes:

1. Presentation Layer (Frontend) – The user interface where educators input their requirements.
2. Application Layer (Backend Processing) – Handles business logic, question selection, and paper formatting.
3. Database Layer (Data Storage) – Manages the question bank and retrieves questions based on specified criteria.

Components of the System Architecture

a) User Interface (Frontend Layer)

The frontend is the part of the system where users interact. It is a web-based interface designed using HTML, CSS, JavaScript, and PHP, ensuring a simple and intuitive experience. The primary functions of the frontend include:

- Allowing educators to input parameters such as subject, number of questions, difficulty level, and question type (MCQs, short answer, long answer).
- Providing options for users to preview the generated question paper before downloading.
- Allowing administrators to add, edit, or delete questions in the question bank.

- Generating the question paper in PDF or DOC format for easy printing or digital sharing.

The frontend interacts with the backend by sending user inputs, which are then processed to generate the required question paper.

Backend Processing Layer

The backend is responsible for processing user requests and generating the question paper based on the given criteria. This layer is developed using PHP and SQL and includes:

- Question Selection Algorithm – Retrieves questions from the database based on subject, topic, and difficulty level while ensuring diversity and randomness.
- Paper Formatting Module – Structures the selected questions in a well-organized format, aligning them in a question paper layout with numbering and sections.
- Randomization and Balancing – Ensures that questions are evenly distributed based on difficulty levels (easy, medium, hard) and topics.
- Validation & Error Handling – Checks for duplicate questions and ensures that the total marks match the required exam format.
- Security Features – Implements authentication and role-based access control (admin vs. teacher users) to secure the database.

This layer acts as a bridge between the frontend (user inputs) and database (retrieving questions), ensuring smooth functionality.

Database Layer (Data Storage)

The database is a crucial component of the system, responsible for storing and managing a large collection of questions. It is built using MySQL and follows a structured format to store questions based on various categories, such as:

- Question ID – A unique identifier for each question.
- Subject & Topic – Categorization of questions into subjects and topics.
- Question Type – Whether the question is MCQ, short answer, or long answer.
- Difficulty Level – Questions are labeled as easy, medium, or hard for balanced distribution.

- Marks Allocation – Defines how much each question is worth.

The database allows for easy retrieval and modification of questions, ensuring that new questions can be added or updated as needed.

Working Process of the System

The workflow of the Automatic Question Paper Generation System is as follows:

1. User Input: The educator logs in and enters the subject, number of questions, difficulty level, and question type via the frontend.
2. Query Processing: The backend retrieves questions from the database based on the selected parameters.
3. Randomization & Selection: The system ensures that the selected questions are randomized and evenly distributed according to difficulty level.
4. Paper Formatting: The selected questions are arranged in a structured format with numbering and sections.
5. Output Generation: The final question paper is generated in PDF or DOC format, ready for download or printing.
6. Admin Controls: Administrators can add, edit, or delete questions from the database as needed.

This structured process ensures an efficient, automated, and error-free approach to generating question papers.

Technologies Used in System Architecture

The system is built using a combination of frontend, backend, and database technologies, ensuring smooth functionality:

- Frontend Technologies: HTML, CSS, JavaScript, PHP (for dynamic content).
- Backend Technologies: PHP (for logic processing), SQL (for data retrieval).
- Database: MySQL (for storing and managing the question bank).
- Other Tools: PDF generation libraries (TCPDF or FPDF) for exporting question papers in PDF format.

Benefits of this Architecture

The structured three-tier architecture ensures several benefits:

1. Scalability: The system can handle large question banks and multiple users simultaneously.
2. Flexibility: Questions can be updated and the system can be customized to different subjects and educational levels.
3. Time Efficiency: Automates the question paper generation process, reducing manual workload.
4. Randomization & Fairness: Ensures a balanced and unique selection of questions for every generated paper.
5. Security & Role-Based Access: Prevents unauthorized users from modifying the question bank.
6. User-Friendly Interface: Educators can easily input criteria and generate papers without technical expertise.

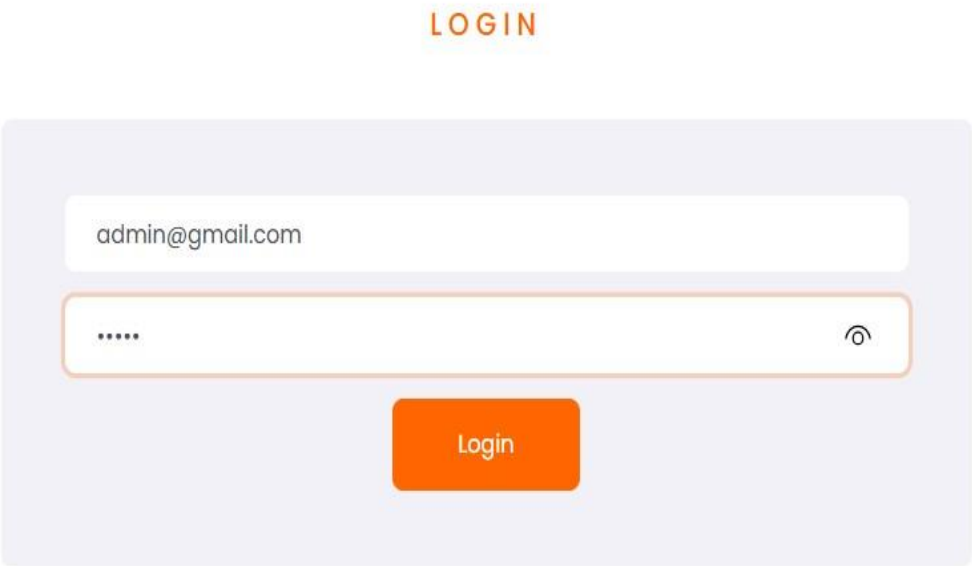
CHAPTER-6

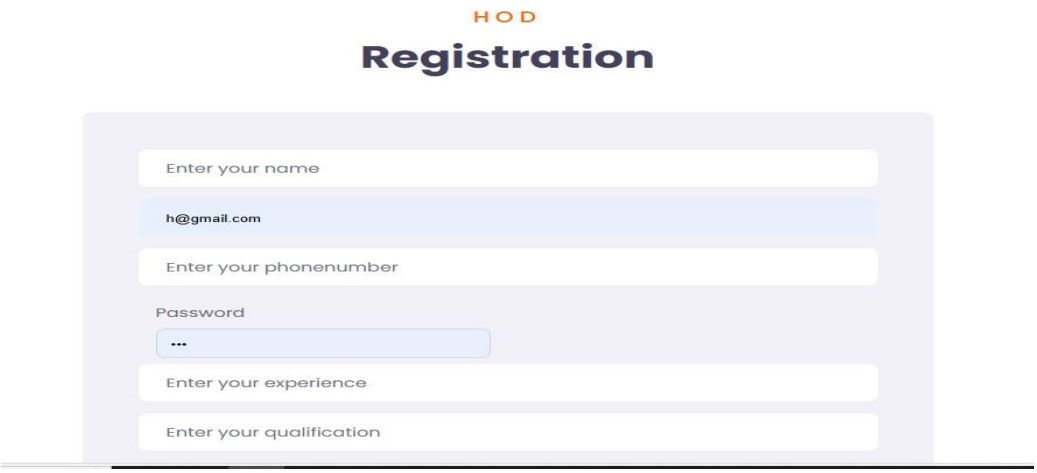
MODULES

The PaperPro – Question Generator System contains many modules for the smooth function of this system and helps it in generating questions and revolving the score card of the students who attended the tests.The following are the modules described in this project:

1. User Authentication Module:

   This system provide smooth, secure login and authentication for admin,hod,teacher and student.It ensures that onlyauthorised signatory can generate question papers. This system provides role based access control system.





2. Data Processing Module:

   This system extracts and clean study materials given by the corresponding pdf. It uses NLP tasks like tokenization, stemming and lemetisation. It helps to identify key concepts and topics using Named Entity Recognition and topic modelling.

3. Question Generation Module:

   This system converts the study material contents given in the format into Multiple Choice Questions and Subjective Questions. For generating contextually relevant questions it uses deep learning model such as T5 transformers and BERT.

4. Question Optimization and filtering Module:

   Accuracy, relevance and quality of the generated questions are evaluated here. The best quality questions are generated and selected using random algorithm. It helps to filters out grammatically error, irrelevant and duplicate questions.

5. Question Paper formatting and Randomization Module:

   It organizes the selected questions in a structured exam format. It helps in customizing the questions in different fomat such as mcq questions, short answers and long answers etc…

6. Score card generation Module:

   It generates the score card of the students and make them to download there score card. If a student attended the test of MCQ once he/she cannot be able to attend it again when they try to do it in second chance then there score card only be displayed.

CHAPTER-7

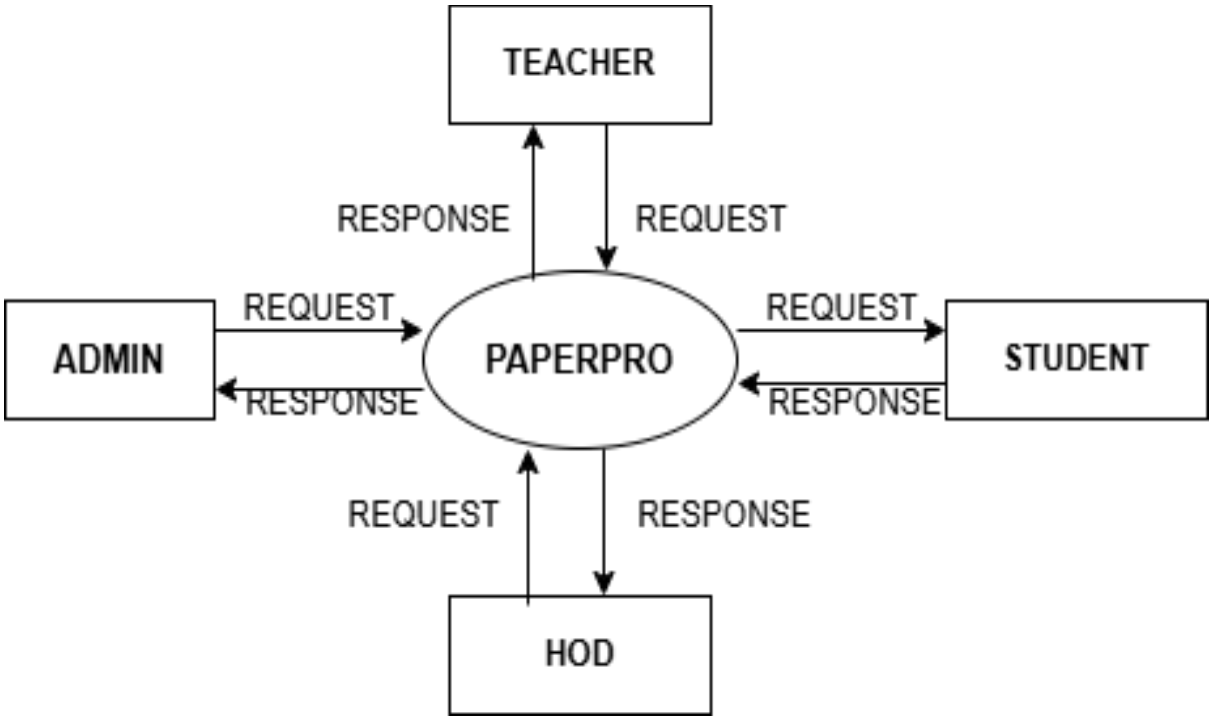
DIAGRAMS

7.1 DATA FLOW DIAGRAM

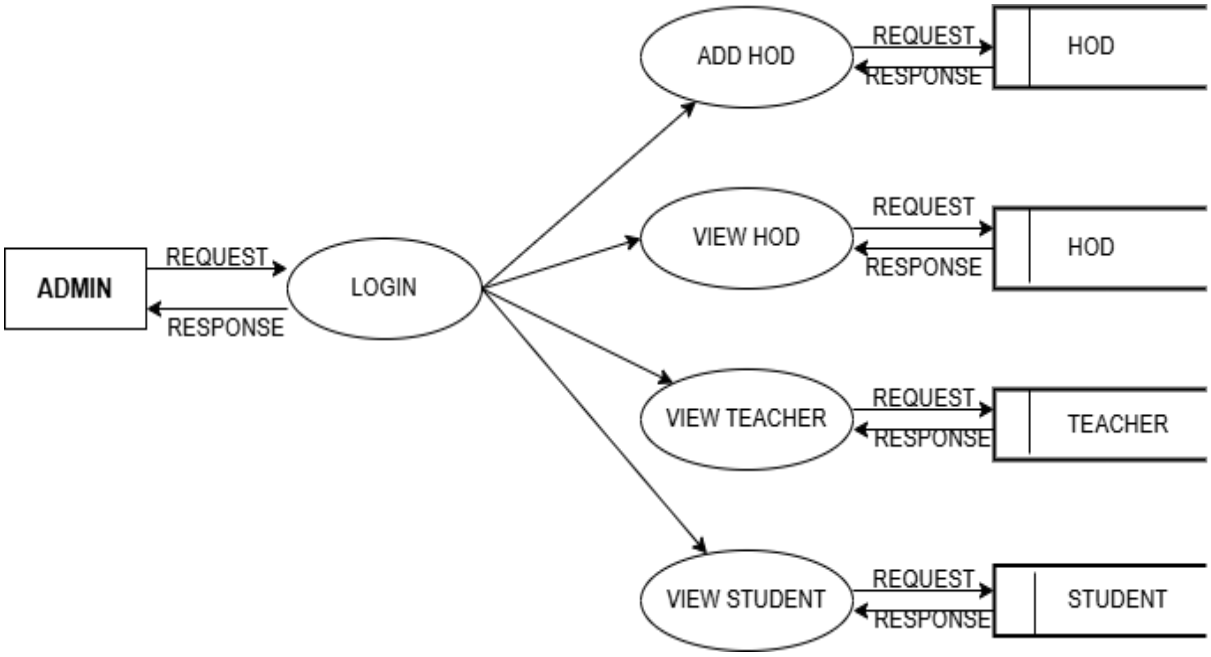
A Data Flow Diagram (DFD) is a graphical representation that shows how data flows through a system, illustrating the inputs, processes, storage, and outputs. It helps to visualize the movement of data between different components of a system.

In the PaperPro – Question Generation System, the DFD Level 0 mainly consists of four modules such as admin, teacher, hod and student. Here the admin add HOD and manages HOD, teacher and students.HOD add and manage teacher and assigns subject to teachers and view note and feedback from students.Teacher manages students and upload notes using ML algorithm and conduct tests. Student then attend the test in both MCQ and Subjective questions, once the student attend the test then they sre able to view and download there score card. In DFD level 1 the functions of admin, hod, teacher and student are explained detailedly.

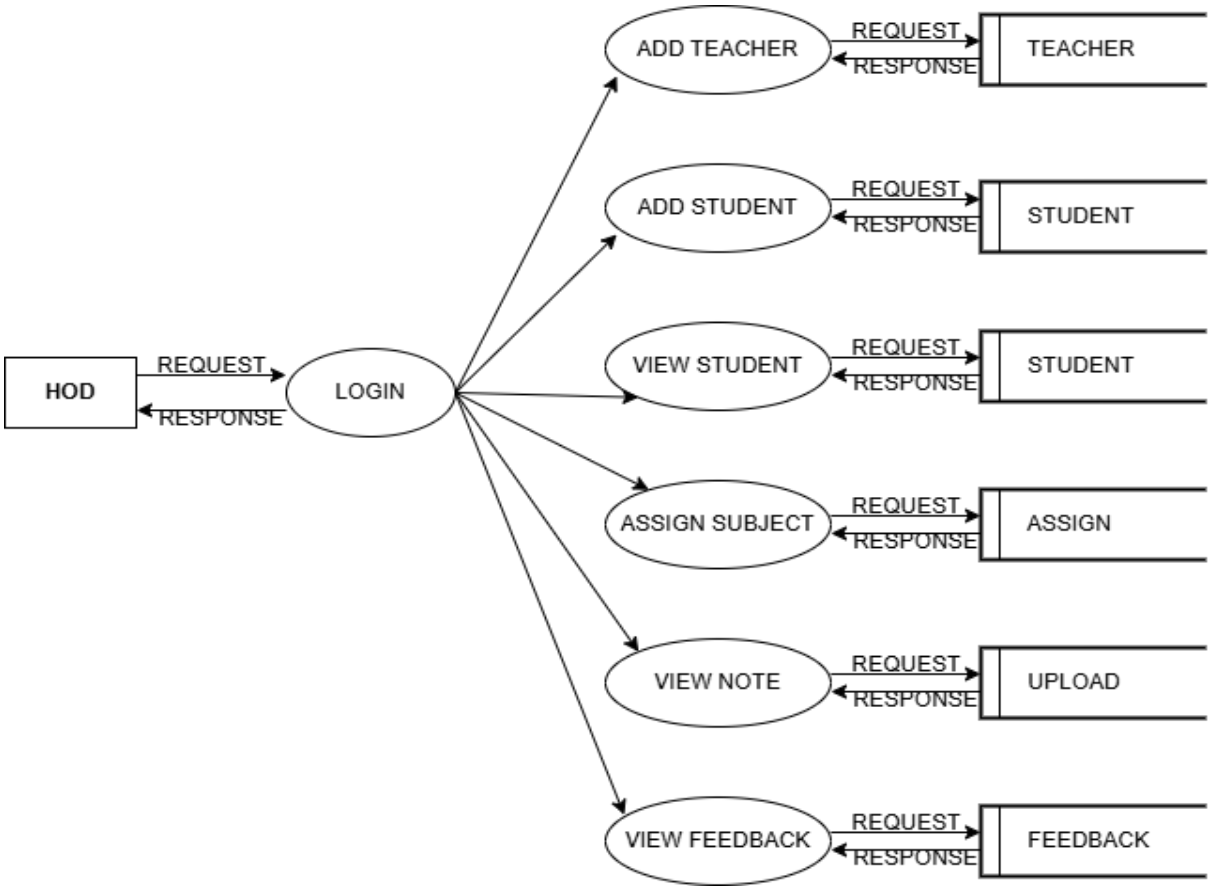
7.1.1 **Level 0 DFD**



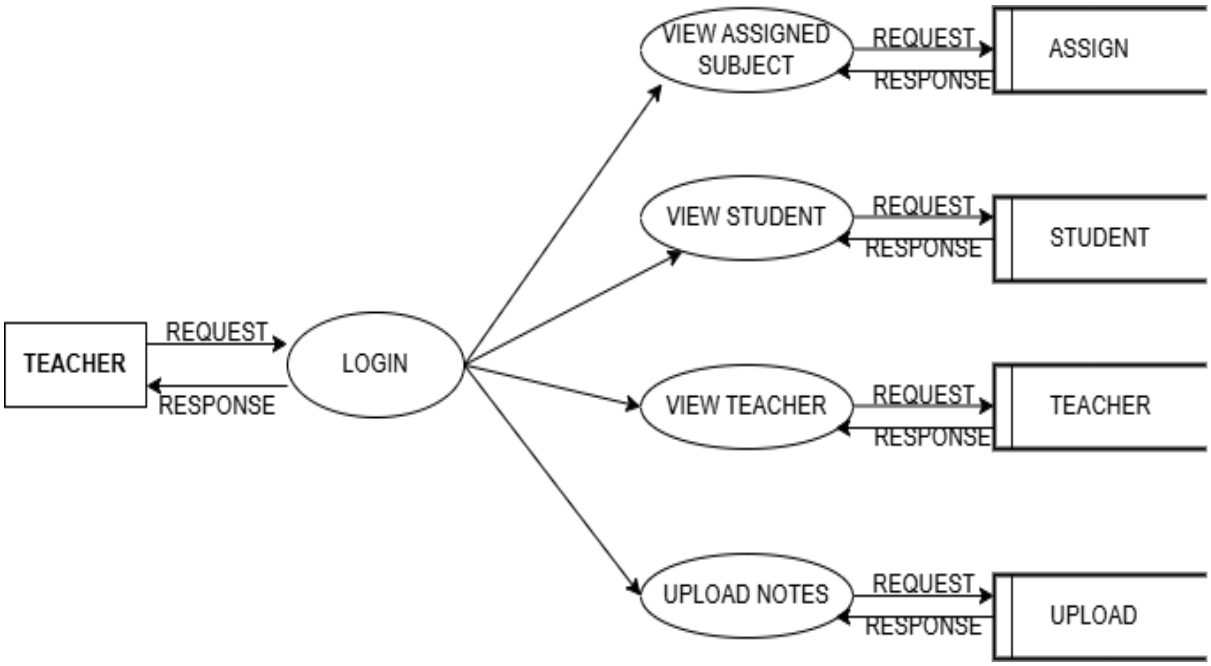
7.1.2 **Level 1 DFD – Admin**



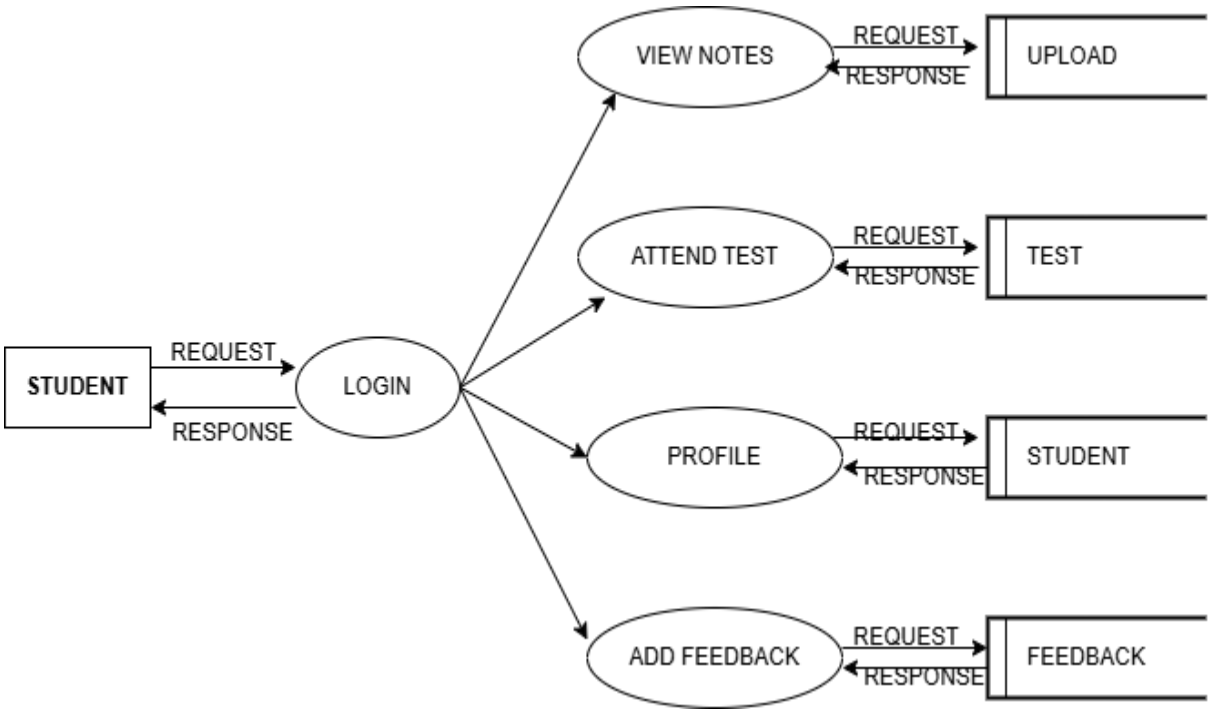
7.1.3 **Level 1 DFD – HOD**



7.1.4 **Level 1 DFD – Teacher**



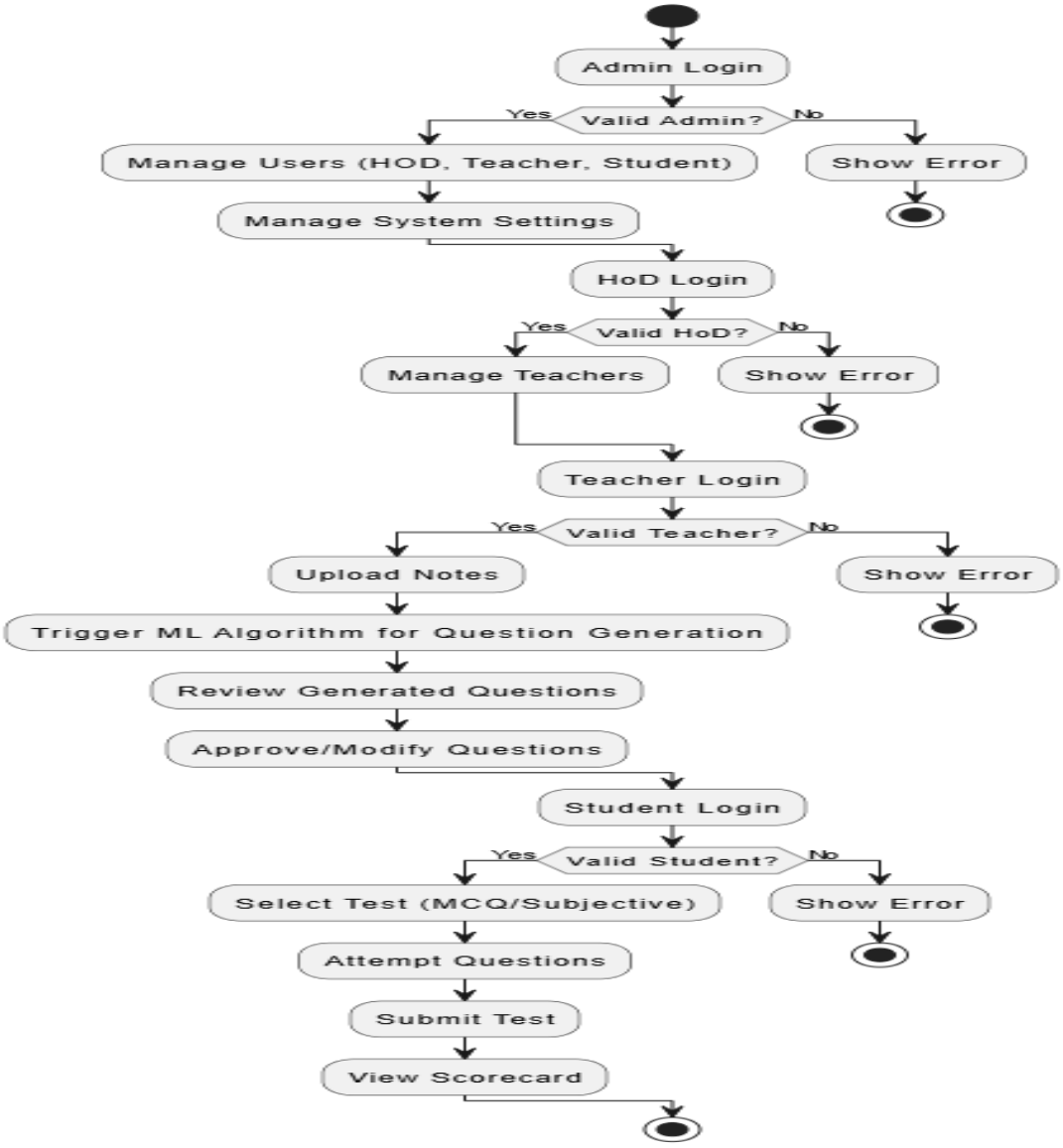
7.1.5 Level 1 DFD – Student



7.2 ACTIVITY DIAGRAM

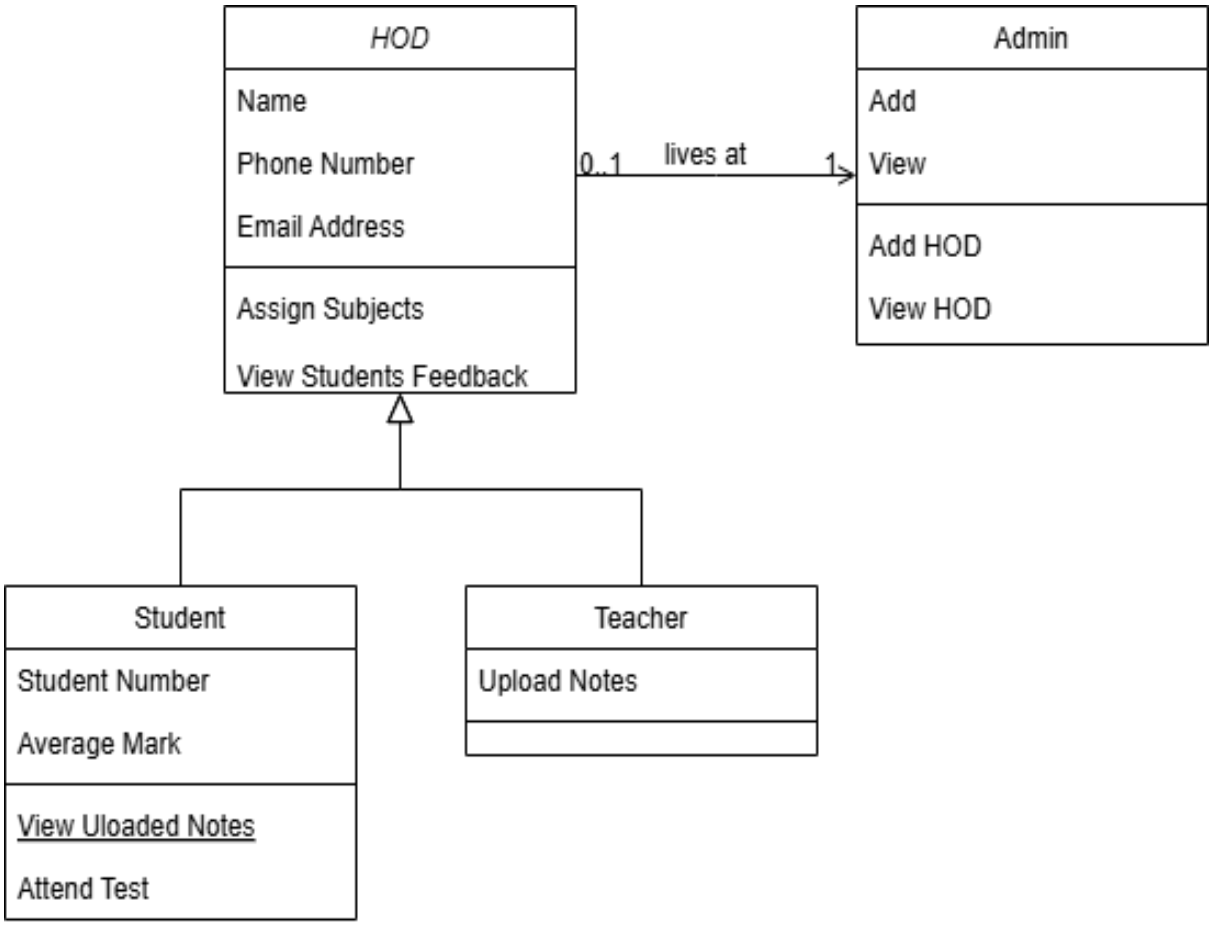
An Activity Diagram is a tool used to represent the flow of control in a system. It mainly focuses on dynamic aspects rather than implementation specifics. It models visually both sequential and concurrent activities. It shows the overall performance of a system.

In this system the Admin module starts with login authentication and followed by managing the entire users of the system and security checks are also done. The HOD module also starts with login authentication and manages teacher module and then resond to the feedback module given by the students. Teacher module firstly authenticates login and followed by it upload the pdf notes and generates question using ML based algorithm. Student module authenticates and followed by it they attend test both in MCQ and subjective levels, once they attend the test they are able to view there score card for there particular test done.



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.



The class diagram of PaperPro – Question Generator consist of mainly admin class, hod class, teacher class and student class. The Admin class manages user accounts. HOD manages teacher accounts. Teacher class manages student accounts and upload notes and generates questions using machine learning algorithms. The test mains the functions of question generation such as selecting the type of test that the student is going to attend whether it is MCQ or subjective questions.

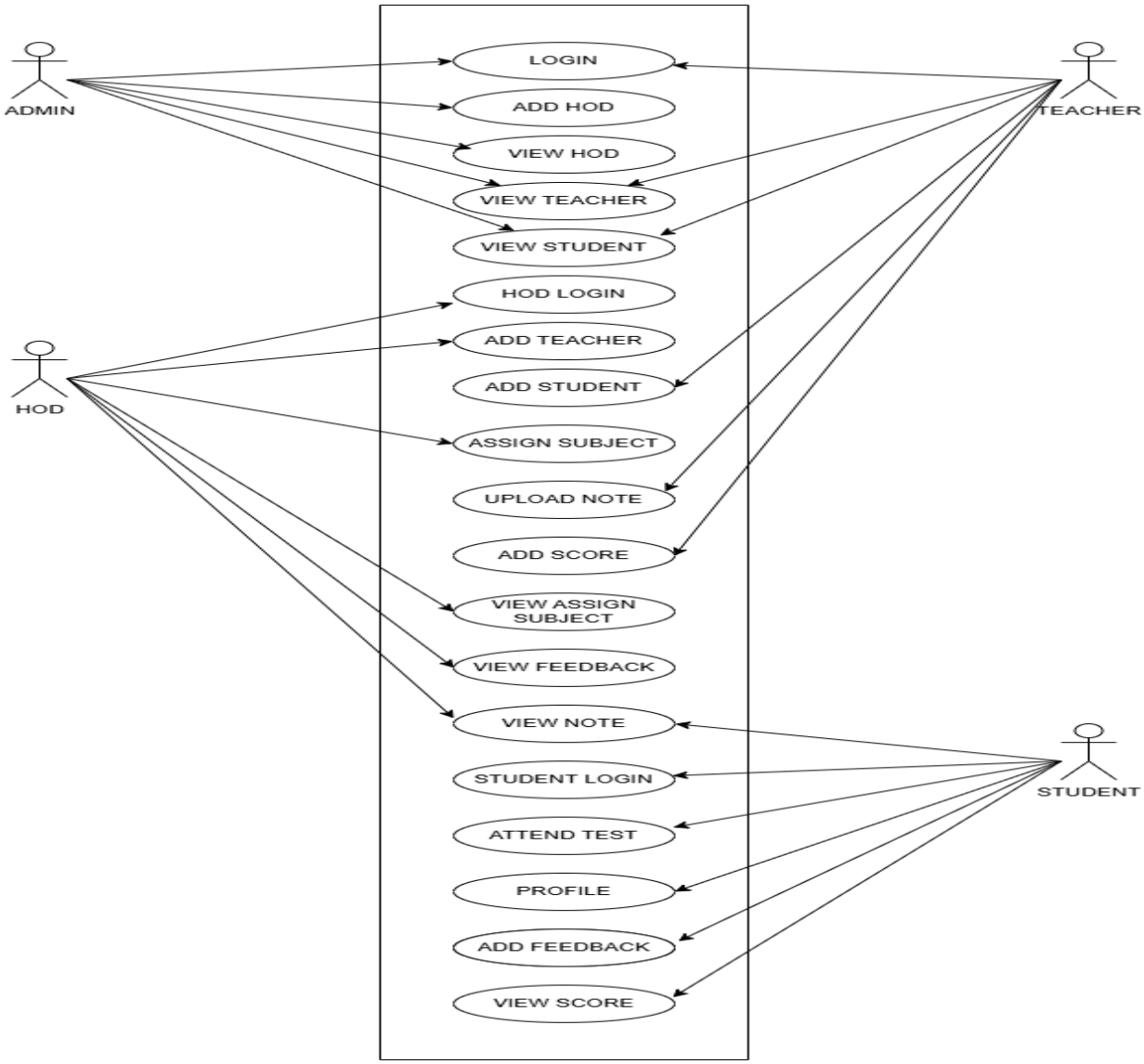
7.4 USECASE DIAGRAM

A Use Case Diagram is a visual representation that depicts the interactions between users (actors) and the system, illustrating the various functions or services (use cases) that the system provides. It helps to understand the system's functionality and the roles of different actors.

Components of Usecase diagram

A usecase diagram mainly consist of components such as actors, usecases and relationships.

- Actors: They represent the users interacting with the system.
- Usecases: It is denoted by oval symbol, they are the actions or processes done by the system.
- Relationships: They show the relationship between the actors and each usecases.
- System Boundaries: They are used to denote the scope of the system generated.



In our system the Admin actor is responsible for managing users, configuring the system, and looking the overall function of the system. HOD manages the teachers and ensure proper working of the system. Teacher interacts with the system by uploading the notes and by reviewing the generated questions.

The student actor is then involved in the test taking activities.They login and download the notes and then select a test type such as MCQ or Subjective questions, attend the test and view the score card.

CHAPTER-8
TESTING

Introduction

System testing is a crucial part in PaperPro – Question Generator System, by ensuring that all the modules have met the expectation of the customer. The testing process validates whether the testing phase met all the functional and non-functional requirements. Various testing phases such as unit testing, integration testing, functional testing and performance testing are done to evaluate the systems scalability and efficiency in working.

Testing Methodologies:

- Unit Testing: Each module is tested individually to verify their functionalities, such as user authentication, question generation and test evaluation to aknowledge that the work done is correctly.
- Integration Testing: It ensures the overall interaction between modules such as the ML algorithm generating questions automatically when the teacher upload notes.
- Functional Testing: It verifies the overall feature of a system are expected or not such as test-taking, score generation and answer evaluation are done properly or not.
- Performance Testing: It checks how the system can handle multiple users at a time and ensuring proper time management.
- Security Testing: It ensures that the proper user authentication is provided to each user, it means that the datas given by the user cannot be manipulated or lost.

Expected Outcomes:

Proper system testing ensures that the students can attend there test without any failure or issues, the teachers are able to upload there notes simultaneously, and the ML system for question generation works efficiently. It also ensures data integrity by preventing unauthorised access of users to the system. The system will be stable, user friendly and fully functional for the real world scenerios.

CHAPTER-9
ADVANTAGES AND DISADVANTAGES

Advantages:

- Time Management: It reduces the time taken by teachers to generate the question paper for examination manually.
- Accuracy: The ML based algorithm helps to generate the questions only from the uploaded notes, it reduces this type of manual errors.
- Test Formation Flexibility: It allows the students to attend different test patterns such as MCQ and Subjective questions.
- Automated Evaluation: This sytem generates or evaluates the answers provided by the students automatically and provide the scores.
- User Role Management: This system mainly provides four users such as admin, hod, teacher and student and they ensure the proper security authentication.

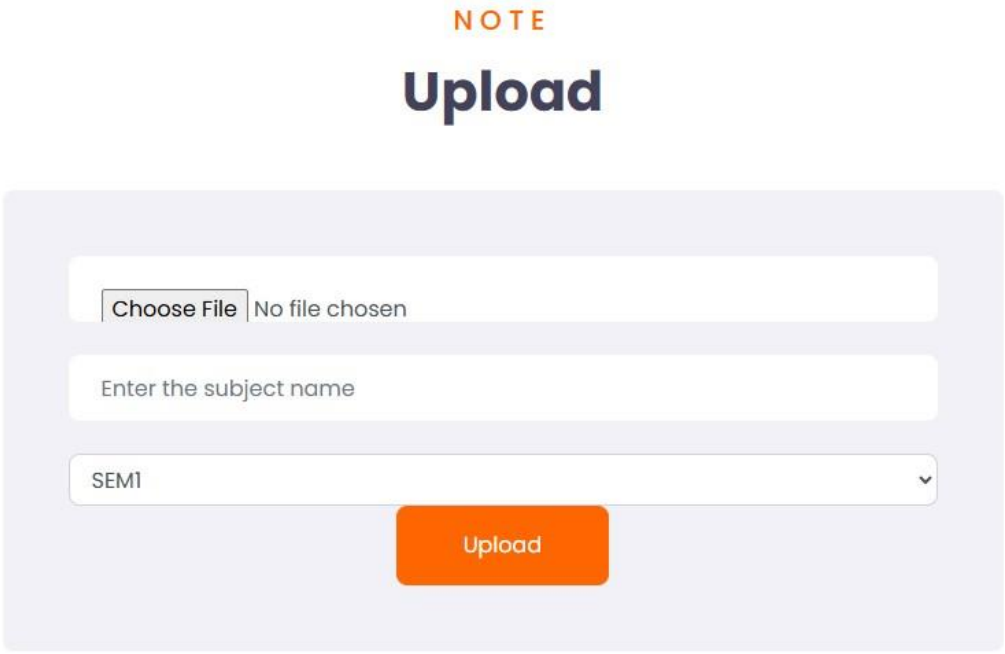
Disadvantages:

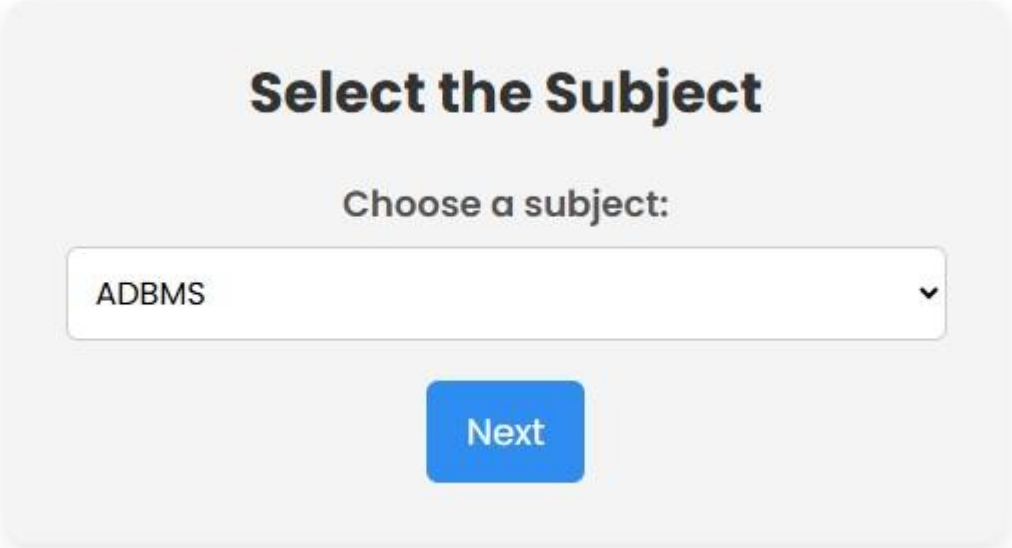
- Dependency of ML algorithm: The quality of the generated questions are fully dependent on the quality and accessibility of the ML algorithm used.
- Limited Subjective Understanding: The system cannot fully understand the words in the notes such as humans, wolds, etc this can provide some error in generated questions.
- Clear Input data: The system requires proper data, it means that if the note uploaded by the teacher is not clear the system is not able to generate meaningful questions.
- Technology expertise: The questions generated using ML based system need an expertise in the technology for integrating machine learning, database management and web development.
- Security: For storing the question paper, user credentials and score card of the student we must ensure proper security to the system.

CHAPTER 10

RESULT AND CONCLUSION

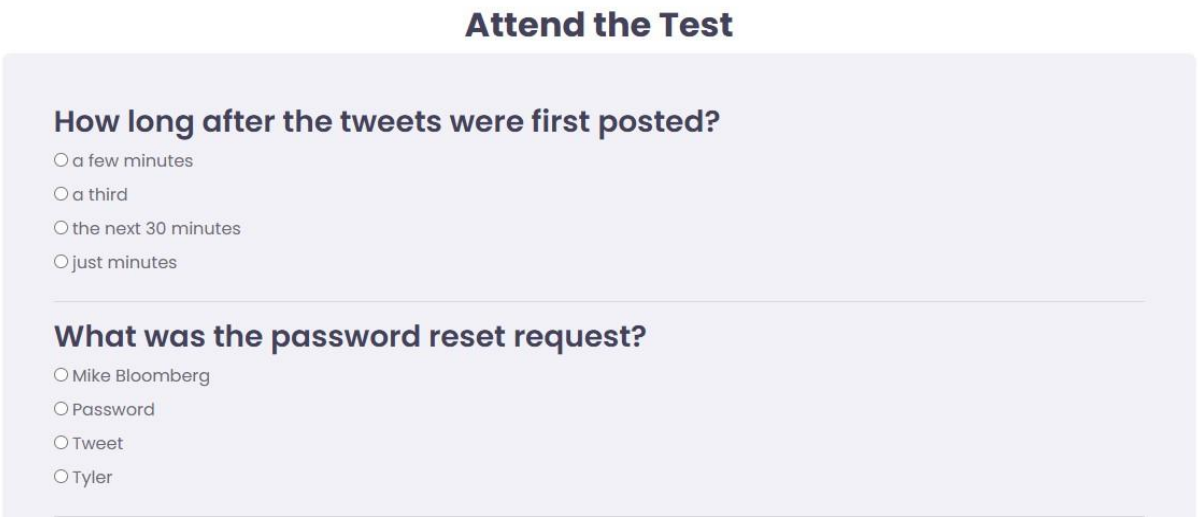
RESULTS:





Select the Question type
Normal Question
MCQ

Attend the Test
What did the campaign of Joe Biden say about the hack?
type your answer...
What is the worst hack of a major social media platform yet?
type your answer...
What is the best way to get rid of the fake accounts?
type your answer...
What other people were targeted?
type your answer...
What happened to the account?
type your answer...


Attend the Test
How long after the tweets were first posted?
a few minutes
a third
the next 30 minutes
just minutes
What was the password reset request?
Mike Bloomberg
Password
Tweet
Tyler

CONCLUSION:

The Automatic Question Paper Generation System is an innovative solution that leverages machine learning to automate the creation of test papers, improving efficiency and accuracy in academic assessments. By integrating multiple user roles—Admin, HOD, Teacher, and Student—the system ensures seamless management of users, study materials, and test evaluations. With the ability to generate both MCQ and subjective questions from uploaded notes, it significantly reduces manual workload while maintaining relevance and quality in assessments. Although challenges like ML dependency, data security, and technical complexity exist, the system's benefits in time efficiency, automation, and flexible test-taking options make it a valuable tool in modern education. This project highlights the potential of AI-driven solutions in streamlining academic processes and enhancing the learning experience.

CHAPTER 11

APPENDICES

Views.py

```
from django.http import HttpResponse
from django.shortcuts import render
from .models import *
from django.shortcuts import redirect, get_object_or_404
# FILE UPLOAD AND VIEW
from  django.core.files.storage import FileSystemStorage
# SESSION
from django.conf import settings
from ml .questiongenerator import QuestionGenerator
import json
import ast
from django.urls import reverse
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from django.db.models import Sum
from django.contrib import messages
from django.shortcuts import render, redirect
def first(request):
    return render(request,'index.html')

def login(request):
    return render(request,'login.html')

def addlogin(request):
    email=request.POST.get('email')
    password=request.POST.get('password')
    if email=='admin@gmail.com' and password=='admin':
```

```
        request.session['login_id']=email
        return render(request,'index.html')

    elif hodtable.objects.filter(email=email,password=password).exists():
        userdetails=hodtable.objects.get(email=request.POST['email'], password=password)
        if userdetails.password == request.POST['password']:
            request.session['hid'] = userdetails.id
            request.session['hname'] = userdetails.name
            return render(request,'index.html')
    elif teacher.objects.filter(email=email,password=password).exists():
        userdetails=teacher.objects.get(email=request.POST['email'], password=password)
        if userdetails.password == request.POST['password']:
            request.session['tid'] = userdetails.id
            request.session['tname'] = userdetails.name
            return render(request,'index.html')

    elif student_table.objects.filter(email=email,password=password).exists():
        userdetails=student_table.objects.get(email=request.POST['email'], password=password)
        if userdetails.password == request.POST['password']:
            request.session['sid'] = userdetails.id
            request.session['sname'] = userdetails.name
            return render(request,'index.html')

    return render(request,'login.html')


def logout(request):
    session_keys=list(request.session.keys())
    for key in session_keys:
        del request.session[key]
```

```
    return redirect(first)


def home(request):
    return render(request,'index.html')



def tech(request):
    return render(request,'staff.html')


def addteacher(request):
    if request.method=='POST':
        email=request.POST.get('email')
        password=request.POST.get('pass1')
        name=request.POST.get('name')
        phone=request.POST.get('phone')

        experience=request.POST.get('experience')
        qualification=request.POST.get('qualification')
        department=request.POST.get('department')
        dob=request.POST.get('dob')
        myfiles=request.FILES['idproof']
        fs=FileSystemStorage()
        filepath=fs.save(myfiles.name,myfiles)
        # filepath= "Testing"
        ins=teacher(name=name, email=email, phone=phone,
                    password=password, experience=experience,
                    qualification=qualification, department=department,
                    dob=dob, idproof=filepath)
        ins.save()

    return render(request,'index.html',{'message':"Successfully Registerd"})
```

```
def hod(request):
    return render(request,'hod.html')

def addhod(request):
    if request.method=='POST':
        email=request.POST.get('email')
        password=request.POST.get("pass1")
        name=request.POST.get('name')
        phone=request.POST.get('phone')

        experience=request.POST.get('experience')
        qualification=request.POST.get('qualification')
        department=request.POST.get('department')
        myfiles=request.FILES['idproof']
        fs=FileSystemStorage()
        filepath=fs.save(myfiles.name,myfiles)
        # filepath= "Testing"
        ins=hodtable(name=name, email=email, phone=phone,
                    password=password, experience=experience,
                    qualification=qualification, department=department,
                    idproof=filepath)
        ins.save()

    return render(request,'index.html',{'message':"Successfully Registerd"})
    return render(request,'hod.html')

def student(request):
    return render(request,'student.html')
```

```
def addstudent(request):
    if request.method=='POST':
        email=request.POST.get('email')
        password=request.POST.get('pass1')
        name=request.POST.get('name')
        phone=request.POST.get('phone')
        age=request.POST.get('age')
        dob=request.POST.get('dob')
        department=request.POST.get('department')
        # image=request.FILES['image']
        # fs=FileSystemStorage()
        # file=fs.save(image.name,image)
        myfiles=request.FILES['idproof']
        fs=FileSystemStorage()
        filepath=fs.save(myfiles.name,myfiles)
        # filepath= "Testing"
        ins=student_table(name=name, email=email, phone=phone,
                    password=password,age=age,
                    dob=dob,department=department,
                    idproof=filepath)
        ins.save()
        messages.success(request,"Registration succussfully completed")
        return render(request,'student.html',{'message':"Registration succussfully completed"})


    return render(request,'student.html',{'message':"Registration succussfully completed"})



def viewstd(request):
    std=student_table.objects.all()
```

```
    return render(request,'viewstd.html',{'result':std})

def viewteacher(request):
    std=teacher.objects.all()
    return render(request,'viewteacher.html',{'result':std})
def feedback(request):
    a=teacher.objects.all()
    return render(request,'feedback.html',{'result':a})

def addfeedback(request):
    if request.method=='POST':
        sname=request.session['sname']
        feedback=request.POST.get('feedback')
        teacher_id=request.POST.get('teacher_id')



        cus=feedtable(user_id=sname,feedback=feedback,teacher_id=teacher_id)
        cus.save()
    return render(request,"index.html",{'message':'sucessfully Added'})


def viewfeedback(request):
    user=feedtable.objects.all()
    ww=teacher.objects.all()
    for i in user:
        for j in ww:
            if str(i.teacher_id) == str(j.id):
                i.teacher_id = j.name

    return render(request,'viewfeedback.html',{'result':user})
```

```
def viewstudent(request):
    fdb=student_table.objects.all()
    return render(request,'viewstudent.html',{'result':fdb})

def delete(request,id):
    user=student_table.objects.get(id=id)
    user.delete()
    return redirect(viewstudent)

def notes(request):
    return render(request,'notes.html')

# def addnotes(request):
#    if request.method=='POST':
#       a=request.POST.get('Subject_name')
#       b=request.POST.get('sem')
#       myfiles=request.FILES['Notes']
#       fs=FileSystemStorage()
#       filepath=fs.save(myfiles.name,myfiles)
#       ins=uploadnotes(Notes=myfiles,Subject_name=a,sem=b)
#       ins.save()
#    return render(request,'notes.html')
import fitz  # PyMuPDF

# def addnotes(request):
#     teacherr=teacher.objects.get(id=request.session['tid'])

#     if request.method == 'POST':
```

```
#       subject_name = request.POST.get('Subject_name')
#       semester = request.POST.get('sem')
#       myfile = request.FILES['Notes']

#       # Save file
#       fs = FileSystemStorage()
#       filepath = fs.save(myfile.name, myfile)
#       file_url = fs.path(filepath)

#       # Save uploaded notes
#       ins = uploadnotes(Notes=myfile, Subject_name=subject_name, sem=semester)
#       ins.save()

#       # Read file content
#       with open(file_url, 'r', encoding='utf-8') as file:
#           file_content = file.read()

#       # Generate Questions
#       qg = QuestionGenerator()

#       # Generate normal questions
#       normal_questions = qg.generate(file_content, num_questions=10, answer_style="sentences")

#       for question in normal_questions:
#           GeneratedQuestions.objects.create(
#               subject_name=subject_name,
#               semester=semester,
#               question_text=question['question'],
#               correct_answer=question.get('answer', ''),
#               question_type="normal",
```

```
#                teacher = teacherr
#            )

#        # Generate multiple-choice questions
#                mcq_questions = qg.generate(file_content, num_questions=10, answer_style="multiple_choice")

#        for question in mcq_questions:
#            GeneratedQuestions.objects.create(
#                subject_name=subject_name,
#                semester=semester,
#                question_text=question['question'],
#                options=question.get('options', ''),
#                correct_answer=question.get('answer', ''),
#                question_type="mcq",
#                teacher = teacherr
#            )

#    return render(request, 'notes.html')

def addnotes(request):
    teacherr = teacher.objects.get(id=request.session['tid'])

    if request.method == 'POST':
        subject_name = request.POST.get('Subject_name')
        semester = request.POST.get('sem')
        myfile = request.FILES['Notes']

        # Save file
        fs = FileSystemStorage()
        filepath = fs.save(myfile.name, myfile)
```

```
file_url = fs.path(filepath)

# Save uploaded notes
ins = uploadnotes(Notes=myfile, Subject_name=subject_name, sem=semester)
ins.save()

# **Read file content**
file_extension = myfile.name.split('.')[-1].lower()

if file_extension == "txt":
    with open(file_url, 'r', encoding='utf-8') as file:
        file_content = file.read()
elif file_extension == "pdf":
    file_content = extract_text_from_pdf(file_url)
else:
    return render(request, 'notes.html', {'error': 'Unsupported file format. Only .txt and .pdf
are allowed.'})

# **Generate Questions**
qg = QuestionGenerator()

# Generate normal questions
normal_questions = qg.generate(file_content, num_questions=10,
answer_style="sentences")

for question in normal_questions:
    GeneratedQuestions.objects.create(
        subject_name=subject_name,
        semester=semester,
        question_text=question['question'],
        correct_answer=question.get('answer', ''),
```

```
            question_type="normal",
            teacher=teacherr
        )

    # Generate multiple-choice questions
    mcq_questions = qg.generate(file_content, num_questions=10, answer_style="multiple_choice")

    for question in mcq_questions:
        GeneratedQuestions.objects.create(
            subject_name=subject_name,
            semester=semester,
            question_text=question['question'],
            options=question.get('options', ''),
            correct_answer=question.get('answer', ''),
            question_type="mcq",
            teacher=teacherr
        )

    return render(request, 'notes.html')

def extract_text_from_pdf(pdf_path):
    """Extracts text from a PDF file"""
    doc = fitz.open(pdf_path)
    text = ""
    for page in doc:
        text += page.get_text("text") + "\n"
    return text

def viewnotes(request):
    upload=uploadnotes.objects.all()
```

```
    return render(request,'viewnotes.html',{'result':upload})

def viewnote(request):
    upload=uploadnotes.objects.all()
    return render(request,'viewnote.html',{'result':upload})

def dlt(request,id):
    user=uploadnotes.objects.get(id=id)
    user.delete()
    return redirect(viewnote)

def viewhod(request):
    upload=hodtable.objects.all()
    return render(request,'viewhod.html',{'result':upload})

def remove(request,id):
    user=hodtable.objects.get(id=id)
    user.delete()
    return redirect(viewhod)


def assign(request):
    sel=teacher.objects.all()
    return render(request,'assign.html',{'result':sel})

def assignsub(request):
   if request.method=='POST':
       teacher_id=request.POST.get('teacher_id')
       a=request.POST.get('subject')
       ins=assign_table(teacher_id=teacher_id,subject_name=a,hod_id=request.session['hid'])
```

```
        ins.save()
    return render(request,'assign.html')
def view_assigned_subject(request):
    sel=assign_table.objects.all()
    user= teacher.objects.all()
    for i in sel:
        for j in user:
            if str(i.teacher_id)==str(j.id):
                i.teacher_id=j.name
    return render(request,'view_assign_subject.html',{'result':sel})
def profile(request):
    sel=student_table.objects.get(id=request.session['sid'])

    return render(request,'profile.html',{'result':sel})

def viewassign(request):
    upload=assign_table.objects.filter(teacher_id=request.session['tid'])
    return render(request,'viewassign.html',{'result':upload})

def select_subject(request):
    subjects = GeneratedQuestions.objects.values_list('subject_name', flat=True).distinct()

    if request.method == "POST":
        selected_subject = request.POST.get("subject")
        return redirect(reverse("question_type", kwargs={"subject_name": selected_subject}))  # ✅ Use reverse()

    return render(request, "select_subject.html", {"subjects": subjects})

def question_type(request, subject_name):
    return render(request, 'question_type.html', {'subject_name': subject_name})
```

```
def normal_question(request, subject_name):
    student=student_table.objects.get(id=request.session['sid'])
    if NormalTestAttempt.objects.filter(student=student, question__subject_name=subject_name).exists():
        return redirect('result')

    normal_question = GeneratedQuestions.objects.filter(question_type="normal", subject_name=subject_name)

    if request.method == "POST":
        for question in normal_question:
            answer = request.POST.get(f"question_{question.id}")
            print("pppppppppppppppppp",answer)
            correct_answer = question.correct_answer
            print("mmmmmmmmmmmmmmmmmmmmm", correct_answer)

            vectorizer = TfidfVectorizer()
            tfidf_matrix = vectorizer.fit_transform([answer, correct_answer])
            cosine_sim = cosine_similarity(tfidf_matrix[0], tfidf_matrix[1])[0][0]

            print("Cosine Similarity:", cosine_sim)

            # Define a threshold for similarity (adjust as needed)
            SIMILARITY_THRESHOLD_5 = 0.9  # 10 points
            SIMILARITY_THRESHOLD_4 = 0.7  # 7 points
            SIMILARITY_THRESHOLD_3 = 0.5  # 5 points
            SIMILARITY_THRESHOLD_2 = 0.3  # 2 points
            SIMILARITY_THRESHOLD_1 = 0.2  # 1 point
```

```
        if cosine_sim >= SIMILARITY_THRESHOLD_5:
            score = 10
            is_correct = True
        elif cosine_sim >= SIMILARITY_THRESHOLD_4:
            score = 7
            is_correct = True
        elif cosine_sim >= SIMILARITY_THRESHOLD_3:
            score = 5
            is_correct = True
        elif cosine_sim >= SIMILARITY_THRESHOLD_2:
            score = 2
            is_correct = True
        elif cosine_sim >= SIMILARITY_THRESHOLD_1:
            score = 1
            is_correct = True
        else:
            score = 0
            is_correct = False

        NormalTestAttempt.objects.create(
            student=student,
            question=question,
            answer=answer,
            correct_answer=correct_answer,
            is_correct=is_correct,
            score=score
        )
    return redirect('select_subject')

return render(request, 'normal_question.html',{'question': normal_question})
```

```
def mcq(request, subject_name):
    student=student_table.objects.get(id=request.session['sid'])
    if MCQTestAttempt.objects.filter(student=student, question__subject_name=subject_name).exists():
        return redirect('result')

    mcq_questions = GeneratedQuestions.objects.filter(question_type="mcq", subject_name=subject_name)

    for question in mcq_questions:
        question.options_list = question.get_options()

        correct_answers_list = []  # Default empty list in case of errors

        if question.correct_answer and isinstance(question.correct_answer, str):
            print("iiiiiiiiiiiiiiiiiiiiiiiiiiii", question.correct_answer)
            print(type(question.correct_answer))

            try:
                # Step 1: Convert the Python string representation into an actual Python list
                python_list = ast.literal_eval(question.correct_answer)

                # Step 2: Convert the Python list to a JSON string
                json_string = json.dumps(python_list)

                # Step 3: Convert JSON string to a valid Python object
                correct_answers_list = json.loads(json_string)

                print(correct_answers_list)  # Debugging
```

```
            print(type(correct_answers_list))

        except (json.JSONDecodeError, ValueError, SyntaxError) as e:
            print("Error: Invalid JSON format in correct_answer:", e)
            correct_answers_list = []

    # Extract only the correct answers safely
    answers = [item['answer'] for item in correct_answers_list if 'answer' in item]
    correct_answers = [item['answer'] for item in correct_answers_list if item.get('correct') ==
True]
    print("ooooooooooooo",correct_answers)

    print("lllllllllllllllllllllllll", answers)
    print(f"Question: {question.question_text}, Correct Answers: {answers}")

    # Attach correct_answers to each question object
    question.answers = answers
    print("pppppppppppppppppp",question.answers)

  if request.method == "POST":
    for question in mcq_questions:
        selected_answer = request.POST.get(f"question_{question.id}", None)

        correct_answers_list = []  # Default empty list in case of errors

        if question.correct_answer and isinstance(question.correct_answer, str):
            try:
                python_list = ast.literal_eval(question.correct_answer)
                json_string = json.dumps(python_list)
                correct_answers_list = json.loads(json_string)
            except (json.JSONDecodeError, ValueError, SyntaxError) as e:
```

```
            print("Error: Invalid JSON format in correct_answer:", e)
            correct_answers_list = []

            correct_answers_list = [item['answer'] for item in correct_answers_list if item.get('correct') == True]
        print("ooooooooooooo",correct_answers_list)
        correct_answers = " ".join(correct_answers_list)
        print("qqqqqqqqqqqqqqqqqq", correct_answers)
        print(question.question_text)
        print(selected_answer)

        if selected_answer == correct_answers:
            score = 1
            is_correct = True
        else:
            score=0
            is_correct = False

        MCQTestAttempt.objects.create(
            student=student,
            question=question,
            selected_answer=selected_answer,
            correct_answer=correct_answers,
            is_correct=is_correct,
            score=score
        )
    return redirect('select_subject')
  return render(request, 'mcq.html', {"mcq_questions": mcq_questions})


def result(request):
  student=student_table.objects.get(id=request.session['sid'])
```

```
    subjects = GeneratedQuestions.objects.values_list("subject_name", flat=True).distinct()
    results = {}

    for subject in subjects:
        # Fetch total score for normal test in the subject
        normal_scores = NormalTestAttempt.objects.filter(student=student, question__subject_name=subject).aggregate(total_score=Sum("score"))

        # Fetch total score for MCQ test in the subject
        mcq_scores = MCQTestAttempt.objects.filter(student=student, question__subject_name=subject).aggregate(total_score=Sum("score"))

        # If no attempts were made, default score to 0
        normal_total = normal_scores["total_score"] or 0
        mcq_total = mcq_scores["total_score"] or 0
        total = normal_total + mcq_total

        results[subject] = {
            "normal": normal_total,
            "mcq": mcq_total,
            "total": total,
        }
    return render(request, 'stu_result.html', {'results': results})

def student_test_result(request):
    students = student_table.objects.all()
    return render(request, 'test_result.html',{'students': students})


def teacher_result(request, id):
    student = student_table.objects.get(id=id)
```

```
    teacher_instance = get_object_or_404(teacher, id=request.session.get('tid'))
    subjects = GeneratedQuestions.objects.filter(teacher=teacher_instance).values_list("subject_name", flat=True).distinct()

    results = {}

    for subject in subjects:
        # Fetch total score for normal test in the subject
        normal_scores = NormalTestAttempt.objects.filter(student=student, question__subject_name=subject).aggregate(total_score=Sum("score"))

        # Fetch total score for MCQ test in the subject
        mcq_scores = MCQTestAttempt.objects.filter(student=student, question__subject_name=subject).aggregate(total_score=Sum("score"))

        # If no attempts were made, default score to 0
        normal_total = normal_scores["total_score"] or 0
        mcq_total = mcq_scores["total_score"] or 0
        total = normal_total + mcq_total

        results[subject] = {
            "normal": normal_total,
            "mcq": mcq_total,
            "total": total,
        }
    return render(request, 'student_result.html', {'results': results})
```

urls.py

```
"""knotebook URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
```

```
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path
from . import views
from django.conf.urls import url
from django.conf import settings
from django.conf.urls.static import static



urlpatterns = [
    #path('admin/', admin.site.urls),
    path('', views.first),
    path('login/', views.login),
    path('login/addlogin', views.addlogin),
    # path('login/logout',views.logout),
    path('logout/',views.logout, name="logout"),
    path('home',views.home),
    path('tech',views.tech),
    path('addteacher',views.addteacher),
```

```
path('addhod',views.addhod),
path('hod',views.hod),
path('student',views.student),
path('addstudent',views.addstudent),
path('viewstd',views.viewstd),
path('viewteacher',views.viewteacher),
path('feedback',views.feedback),
path('addfeedback',views.addfeedback),
path('viewfeedback',views.viewfeedback),
path('viewstudent',views.viewstudent),
path('delete/<int:id>',views.delete),
path('remove/<int:id>',views.remove),
path('notes',views.notes),
path('addnotes',views.addnotes),
path('viewnotes',views.viewnotes),
path('viewnote',views.viewnote),
path('dlt/<int:id>',views.dlt),
path('viewhod',views.viewhod),
path('profile',views.profile),
path('assign',views.assign),
path('assignsub',views.assignsub),
path('viewassign',views.viewassign),
path('view_assigned_subject/',views.view_assigned_subject),
path('select_subject/',views.select_subject, name="select_subject"),
path('question_type/<str:subject_name>/',views.question_type, name="question_type"),
path('normal_question/<str:subject_name>/',views.normal_question,
name="normal_question"),
path('mcq/<str:subject_name>/',views.mcq, name="mcq"),
path('result/',views.result, name="result"),
path('students/',views.student_test_result, name="student_test_result"),
path('student/result/<int:id>/',views.teacher_result, name="teacher_result"),
```

```
]+ static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

Index.html

```
{% load static %}
   {% block content %}

{% include 'header.html' %}

{% if message %}
<script>
   alert('{{message}}')
</script>
{% endif %}
   <!-- Carousel Start -->
   <div class="container-fluid p-0 pb-5 mb-5">
      <div id="header-carousel" class="carousel slide carousel-fade" data-ride="carousel">
         <ol class="carousel-indicators">
            <li data-target="#header-carousel" data-slide-to="0" class="active"></li>
            <li data-target="#header-carousel" data-slide-to="1"></li>
            <li data-target="#header-carousel" data-slide-to="2"></li>
         </ol>
         <div class="carousel-inner">
            <div class="carousel-item active" style="min-height: 300px;">
                  <img class="position-relative w-100" src="{% static 'img/carousel-1.jpg' %}"
style="min-height: 300px; object-fit: cover;">
               <div class="carousel-caption d-flex align-items-center justify-content-center">
                  <div class="p-5" style="width: 100%; max-width: 900px;">
                     <h5 class="text-white text-uppercase mb-md-3">Best Online Courses</h5>
                        <h1 class="display-3 text-white mb-md-4">Best Education From Your
Home</h1>
```

```
                <a href="" class="btn btn-primary py-md-2 px-md-4 font-weight-semi-bold mt-2">Learn More</a>
                </div>
            </div>
        </div>
        <div class="carousel-item" style="min-height: 300px;">
            <img class="position-relative w-100" src="{% static 'img/carousel-2.jpg' %}" style="min-height: 300px; object-fit: cover;">
            <div class="carousel-caption d-flex align-items-center justify-content-center">
                <div class="p-5" style="width: 100%; max-width: 900px;">
                    <h5 class="text-white text-uppercase mb-md-3">Best Online Courses</h5>
                    <h1 class="display-3 text-white mb-md-4">Best Online Learning Platform</h1>
                <a href="" class="btn btn-primary py-md-2 px-md-4 font-weight-semi-bold mt-2">Learn More</a>
                </div>
            </div>
        </div>
        <div class="carousel-item" style="min-height: 300px;">
            <img class="position-relative w-100" src="{% static 'img/carousel-3.jpg' %}" style="min-height: 300px; object-fit: cover;">
            <div class="carousel-caption d-flex align-items-center justify-content-center">
                <div class="p-5" style="width: 100%; max-width: 900px;">
                    <h5 class="text-white text-uppercase mb-md-3">Best Online Courses</h5>
                    <h1 class="display-3 text-white mb-md-4">New Way To Learn From Home</h1>
                <a href="" class="btn btn-primary py-md-2 px-md-4 font-weight-semi-bold mt-2">Learn More</a>
                </div>
            </div>
        </div>
    </div>
</div>
```

```
    </div>
    <!-- Carousel End -->


    <!-- About Start -->
    <div class="container-fluid py-5">
        <div class="container py-5">
            <div class="row align-items-center">
                <div class="col-lg-5">
                    <img class="img-fluid rounded mb-4 mb-lg-0" src="{% static 'img/about.jpg' %}"
alt="">
                </div>
                <div class="col-lg-7">
                    <div class="text-left mb-4">
                            <h5 class="text-primary text-uppercase mb-3" style="letter-spacing:
5px;">About Us</h5>
                        <h1>Innovative Way To Learn</h1>
                    </div>
                    <p>Aliquyam accusam clita nonumy ipsum sit sea clita ipsum clita, ipsum dolores
amet voluptua duo dolores et sit ipsum rebum, sadipscing et erat eirmod diam kasd labore clita
est. Diam sanctus gubergren sit rebum clita amet, sea est sea vero sed et. Sadipscing labore
tempor at sit dolor clita consetetur diam. Diam ut diam tempor no et, lorem dolore invidunt no
nonumy stet ea labore, dolor justo et sit gubergren diam sed sed no ipsum. Sit tempor ut
nonumy elitr dolores justo aliquyam ipsum stet</p>
                    <a href="" class="btn btn-primary py-md-2 px-md-4 font-weight-semi-bold mt-
2">Learn More</a>
                </div>
            </div>
        </div>
    </div>
    <!-- About End -->


    <!-- Category Start -->
    <div class="container-fluid py-5">
```

```
<div class="container pt-5 pb-3">
  <div class="text-center mb-5">
        <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Subjects</h5>
    <h1>Explore Top Subjects</h1>
  </div>
  <div class="row">
    <div class="col-lg-3 col-md-6 mb-4">
      <div class="cat-item position-relative overflow-hidden rounded mb-2">
        <img class="img-fluid" src="{% static 'img/cat-1.jpg' %}" alt="">
        <a class="cat-overlay text-white text-decoration-none" href="">
          <h4 class="text-white font-weight-medium">Web Design</h4>
          <span>100 Courses</span>
        </a>
      </div>
    </div>
    <div class="col-lg-3 col-md-6 mb-4">
      <div class="cat-item position-relative overflow-hidden rounded mb-2">
        <img class="img-fluid" src="{% static 'img/cat-2.jpg' %}" alt="">
        <a class="cat-overlay text-white text-decoration-none" href="">
          <h4 class="text-white font-weight-medium">Development</h4>
          <span>100 Courses</span>
        </a>
      </div>
    </div>
    <div class="col-lg-3 col-md-6 mb-4">
      <div class="cat-item position-relative overflow-hidden rounded mb-2">
        <img class="img-fluid" src="{% static 'img/cat-3.jpg' %}" alt="">
        <a class="cat-overlay text-white text-decoration-none" href="">
          <h4 class="text-white font-weight-medium">Game Design</h4>
          <span>100 Courses</span>
```

```
            </a>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 mb-4">
        <div class="cat-item position-relative overflow-hidden rounded mb-2">
            <img class="img-fluid" src="{% static 'img/cat-4.jpg' %}" alt="">
            <a class="cat-overlay text-white text-decoration-none" href="">
                <h4 class="text-white font-weight-medium">Apps Design</h4>
                <span>100 Courses</span>
            </a>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 mb-4">
        <div class="cat-item position-relative overflow-hidden rounded mb-2">
            <img class="img-fluid" src="{% static 'img/cat-5.jpg' %}" alt="">
            <a class="cat-overlay text-white text-decoration-none" href="">
                <h4 class="text-white font-weight-medium">Marketing</h4>
                <span>100 Courses</span>
            </a>
        </div>
    </div>
    <div class="col-lg-3 col-md-6 mb-4">
        <div class="cat-item position-relative overflow-hidden rounded mb-2">
            <img class="img-fluid" src="{% static 'img/cat-6.jpg' %}" alt="">
            <a class="cat-overlay text-white text-decoration-none" href="">
                <h4 class="text-white font-weight-medium">Research</h4>
                <span>100 Courses</span>
            </a>
        </div>
    </div>
```

```
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="cat-item position-relative overflow-hidden rounded mb-2">
                    <img class="img-fluid" src="{% static 'img/cat-7.jpg' %}" alt="">
                    <a class="cat-overlay text-white text-decoration-none" href="">
                        <h4 class="text-white font-weight-medium">Content Writing</h4>
                        <span>100 Courses</span>
                    </a>
                </div>
            </div>
            <div class="col-lg-3 col-md-6 mb-4">
                <div class="cat-item position-relative overflow-hidden rounded mb-2">
                    <img class="img-fluid" src="{% static 'img/cat-8.jpg' %}" alt="">
                    <a class="cat-overlay text-white text-decoration-none" href="">
                        <h4 class="text-white font-weight-medium">SEO</h4>
                        <span>100 Courses</span>
                    </a>
                </div>
            </div>
        </div>
    </div>
</div>
<!-- Category Start -->


<!-- Courses Start -->
<div class="container-fluid py-5">
    <div class="container py-5">
        <div class="text-center mb-5">
                    <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Courses</h5>
            <h1>Our Popular Courses</h1>
        </div>
```

```
<div class="row">
  <div class="col-lg-4 col-md-6 mb-4">
    <div class="rounded overflow-hidden mb-2">
      <img class="img-fluid" src="{% static 'img/course-1.jpg' %}" alt="">
      <div class="bg-secondary p-4">
        <div class="d-flex justify-content-between mb-3">
            <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25
Students</small>
          <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h
30m</small>
        </div>
        <a class="h5" href="">Web design & development courses for beginner</a>
        <div class="border-top mt-4 pt-4">
          <div class="d-flex justify-content-between">
              <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5
<small>(250)</small></h6>
            <h5 class="m-0">$99</h5>
          </div>
        </div>
      </div>
    </div>
  </div>
  <div class="col-lg-4 col-md-6 mb-4">
    <div class="rounded overflow-hidden mb-2">
      <img class="img-fluid" src="{% static 'img/course-2.jpg' %}" alt="">
      <div class="bg-secondary p-4">
        <div class="d-flex justify-content-between mb-3">
            <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25
Students</small>
          <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h
30m</small>
        </div>
```

```
                <a class="h5" href="">Web design & development courses for beginner</a>
                <div class="border-top mt-4 pt-4">
                  <div class="d-flex justify-content-between">
                        <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5
<small>(250)</small></h6>
                      <h5 class="m-0">$99</h5>
                  </div>
                </div>
              </div>
            </div>
            <div class="col-lg-4 col-md-6 mb-4">
              <div class="rounded overflow-hidden mb-2">
                <img class="img-fluid" src="{% static 'img/course-3.jpg' %}" alt="">
                <div class="bg-secondary p-4">
                <div class="d-flex justify-content-between mb-3">
                      <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25
Students</small>
                    <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h
30m</small>
                </div>
                <a class="h5" href="">Web design & development courses for beginner</a>
                <div class="border-top mt-4 pt-4">
                  <div class="d-flex justify-content-between">
                        <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5
<small>(250)</small></h6>
                      <h5 class="m-0">$99</h5>
                  </div>
                </div>
              </div>
            </div>
```

```
<div class="col-lg-4 col-md-6 mb-4">
    <div class="rounded overflow-hidden mb-2">
        <img class="img-fluid" src="{% static 'img/course-4.jpg' %}" alt="">
        <div class="bg-secondary p-4">
            <div class="d-flex justify-content-between mb-3">
                <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25 Students</small>
                <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h 30m</small>
            </div>
            <a class="h5" href="">Web design & development courses for beginner</a>
            <div class="border-top mt-4 pt-4">
                <div class="d-flex justify-content-between">
                    <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5 <small>(250)</small></h6>
                    <h5 class="m-0">$99</h5>
                </div>
            </div>
        </div>
    </div>
</div>
<div class="col-lg-4 col-md-6 mb-4">
    <div class="rounded overflow-hidden mb-2">
        <img class="img-fluid" src="{% static 'img/course-5.jpg' %}" alt="">
        <div class="bg-secondary p-4">
            <div class="d-flex justify-content-between mb-3">
                <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25 Students</small>
                <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h 30m</small>
            </div>
            <a class="h5" href="">Web design & development courses for beginner</a>
```

```
                <div class="border-top mt-4 pt-4">
                  <div class="d-flex justify-content-between">
                        <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5
<small>(250)</small></h6>
                    <h5 class="m-0">$99</h5>
                  </div>
                </div>
              </div>
            </div>
            <div class="col-lg-4 col-md-6 mb-4">
              <div class="rounded overflow-hidden mb-2">
                <img class="img-fluid" src="{% static 'img/course-6.jpg' %}" alt="">
                <div class="bg-secondary p-4">
                <div class="d-flex justify-content-between mb-3">
                     <small class="m-0"><i class="fa fa-users text-primary mr-2"></i>25
Students</small>
                   <small class="m-0"><i class="far fa-clock text-primary mr-2"></i>01h
30m</small>
                </div>
                <a class="h5" href="">Web design & development courses for beginner</a>
                <div class="border-top mt-4 pt-4">
                  <div class="d-flex justify-content-between">
                        <h6 class="m-0"><i class="fa fa-star text-primary mr-2"></i>4.5
<small>(250)</small></h6>
                    <h5 class="m-0">$99</h5>
                  </div>
                </div>
              </div>
            </div>
          </div>
```

```
    </div>
  </div>
  <!-- Courses End -->


  <!-- Registration Start -->
  <div class="container-fluid bg-registration py-5" style="margin: 90px 0;">
    <div class="container py-5">
      <div class="row align-items-center">
        <div class="col-lg-7 mb-5 mb-lg-0">
          <div class="mb-4">
            <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Need Any Courses</h5>
              <h1 class="text-white">30% Off For New Students</h1>
          </div>
            <p class="text-white">Invidunt lorem justo sanctus clita. Erat lorem labore ea, justo dolor lorem ipsum ut sed eos,
              ipsum et dolor kasd sit ea justo. Erat justo sed sed diam. Ea et erat ut sed diam sea ipsum est
              dolor</p>
          <ul class="list-inline text-white m-0">
            <li class="py-2"><i class="fa fa-check text-primary mr-3"></i>Labore eos amet dolor amet diam</li>
            <li class="py-2"><i class="fa fa-check text-primary mr-3"></i>Etsea et sit dolor amet ipsum</li>
              <li class="py-2"><i class="fa fa-check text-primary mr-3"></i>Diam dolor diam elitripsum vero.</li>
          </ul>
        </div>
        <div class="col-lg-5">
          <div class="card border-0">
            <div class="card-header bg-light text-center p-4">
              <h1 class="m-0">Sign Up Now</h1>
```

```
                </div>
                <div class="card-body rounded-bottom bg-primary p-5">
                  <form>
                    <div class="form-group">
                      <input type="text" class="form-control border-0 p-4" placeholder="Your
name" required="required" />
                    </div>
                    <div class="form-group">
                    <input type="email" class="form-control border-0 p-4" placeholder="Your
email" required="required" />
                    </div>
                    <div class="form-group">
                      <select class="custom-select border-0 px-4" style="height: 47px;">
                        <option selected>Select a course</option>
                        <option value="1">Course 1</option>
                        <option value="2">Course 1</option>
                        <option value="3">Course 1</option>
                      </select>
                    </div>
                    <div>
                    <button class="btn btn-dark btn-block border-0 py-3" type="submit">Sign
Up Now</button>
                    </div>
                  </form>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    <!-- Registration End -->
```

```
<!-- Team Start -->
<div class="container-fluid py-5">
    <div class="container pt-5 pb-3">
        <div class="text-center mb-5">
                    <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Teachers</h5>
            <h1>Meet Our Teachers</h1>
        </div>
        <div class="row">
            <div class="col-md-6 col-lg-3 text-center team mb-4">
                <div class="team-item rounded overflow-hidden mb-2">
                    <div class="team-img position-relative">
                        <img class="img-fluid" src="{% static 'img/team-1.jpg' %}" alt="">
                        <div class="team-social">
                            <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab fa-twitter"></i></a>
                            <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab fa-facebook-f"></i></a>
                            <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab fa-linkedin-in"></i></a>
                        </div>
                    </div>
                    <div class="bg-secondary p-4">
                        <h5>Jhon Doe</h5>
                        <p class="m-0">Web Designer</p>
                    </div>
                </div>
            </div>
            <div class="col-md-6 col-lg-3 text-center team mb-4">
                <div class="team-item rounded overflow-hidden mb-2">
                    <div class="team-img position-relative">
```

```
                <img class="img-fluid" src="{% static 'img/team-2.jpg' %}" alt="">
                <div class="team-social">
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-twitter"></i></a>
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-facebook-f"></i></a>
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-linkedin-in"></i></a>
                </div>
            </div>
            <div class="bg-secondary p-4">
                <h5>Jhon Doe</h5>
                <p class="m-0">Web Designer</p>
            </div>
        </div>
    </div>
    <div class="col-md-6 col-lg-3 text-center team mb-4">
        <div class="team-item rounded overflow-hidden mb-2">
            <div class="team-img position-relative">
                <img class="img-fluid" src="{% static 'img/team-3.jpg' %}" alt="">
                <div class="team-social">
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-twitter"></i></a>
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-facebook-f"></i></a>
                    <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-linkedin-in"></i></a>
                </div>
            </div>
            <div class="bg-secondary p-4">
                <h5>Jhon Doe</h5>
                <p class="m-0">Web Designer</p>
```

```
                </div>
            </div>
        </div>
        <div class="col-md-6 col-lg-3 text-center team mb-4">
            <div class="team-item rounded overflow-hidden mb-2">
                <div class="team-img position-relative">
                    <img class="img-fluid" src="{% static 'img/team-4.jpg' %}" alt="">
                    <div class="team-social">
                        <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-twitter"></i></a>
                        <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-facebook-f"></i></a>
                        <a class="btn btn-outline-light btn-square mx-1" href="#"><i class="fab
fa-linkedin-in"></i></a>
                    </div>
                </div>
                <div class="bg-secondary p-4">
                    <h5>Jhon Doe</h5>
                    <p class="m-0">Web Designer</p>
                </div>
            </div>
        </div>
    </div>
</div>
<!-- Team End -->

<!-- Testimonial Start -->
<div class="container-fluid py-5">
    <div class="container py-5">
        <div class="text-center mb-5">
```

```
                <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Testimonial</h5>
            <h1>What Say Our Students</h1>
        </div>
        <div class="row justify-content-center">
            <div class="col-lg-8">
                <div class="owl-carousel testimonial-carousel">
                    <div class="text-center">
                        <i class="fa fa-3x fa-quote-left text-primary mb-4"></i>
                            <h4 class="font-weight-normal mb-4">Dolor eirmod diam stet kasd sed. Aliqu rebum est eos. Rebum elitr dolore et eos labore, stet justo sed est sed. Diam sed sed dolor stet amet eirmod eos labore diam</h4>
                        <img class="img-fluid mx-auto mb-3" src="{% static 'img/testimonial-1.jpg' %}" alt="">
                        <h5 class="m-0">Client Name</h5>
                        <span>Profession</span>
                    </div>
                    <div class="text-center">
                        <i class="fa fa-3x fa-quote-left text-primary mb-4"></i>
                            <h4 class="font-weight-normal mb-4">Dolor eirmod diam stet kasd sed. Aliqu rebum est eos. Rebum elitr dolore et eos labore, stet justo sed est sed. Diam sed sed dolor stet amet eirmod eos labore diam</h4>
                        <img class="img-fluid mx-auto mb-3" src="{% static 'img/testimonial-2.jpg' %)" alt="">
                        <h5 class="m-0">Client Name</h5>
                        <span>Profession</span>
                    </div>
                    <div class="text-center">
                        <i class="fa fa-3x fa-quote-left text-primary mb-4"></i>
                            <h4 class="font-weight-normal mb-4">Dolor eirmod diam stet kasd sed. Aliqu rebum est eos. Rebum elitr dolore et eos labore, stet justo sed est sed. Diam sed sed dolor stet amet eirmod eos labore diam</h4>
                        <img class="img-fluid mx-auto mb-3" src="{% static 'img/testimonial-3.jpg' %)" alt="">
```

```
                <h5 class="m-0">Client Name</h5>
                <span>Profession</span>
            </div>
        </div>
    </div>
  </div>
</div>
<!-- Testimonial End -->

<!-- Blog Start -->
<div class="container-fluid py-5">
  <div class="container pt-5 pb-3">
    <div class="text-center mb-5">
        <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Our Blog</h5>
      <h1>Latest From Our Blog</h1>
    </div>
    <div class="row pb-3">
      <div class="col-lg-4 mb-4">
        <div class="blog-item position-relative overflow-hidden rounded mb-2">
          <img class="img-fluid" src="{% static 'img/blog-1.jpg' %}" alt="">
          <a class="blog-overlay text-decoration-none" href="">
              <h5 class="text-white mb-3">Lorem elitr magna stet eirmod labore amet labore clita at ut clita</h5>
            <p class="text-primary m-0">Jan 01, 2050</p>
          </a>
        </div>
      </div>
      <div class="col-lg-4 mb-4">
        <div class="blog-item position-relative overflow-hidden rounded mb-2">
```

```
                <img class="img-fluid" src="{% static 'img/blog-2.jpg' %}" alt="">
                <a class="blog-overlay text-decoration-none" href="">
                    <h5 class="text-white mb-3">Lorem elitr magna stet eirmod labore amet labore clita at ut clita</h5>
                  <p class="text-primary m-0">Jan 01, 2050</p>
                </a>
              </div>
            </div>
            <div class="col-lg-4 mb-4">
              <div class="blog-item position-relative overflow-hidden rounded mb-2">
                <img class="img-fluid" src="{% static 'img/blog-3.jpg' %}" alt="">
                <a class="blog-overlay text-decoration-none" href="">
                    <h5 class="text-white mb-3">Lorem elitr magna stet eirmod labore amet labore clita at ut clita</h5>
                  <p class="text-primary m-0">Jan 01, 2050</p>
                </a>
              </div>
            </div>
          </div>
        </div>
    </div>
    <!-- Blog End -->


    <!-- Back to Top -->
    <a href="#" class="btn btn-lg btn-primary btn-lg-square back-to-top"><i class="fa fa-angle-double-up"></i></a>


    {% endblock %}
```

Login.html

```
{% load static %}

{% block content %}

{% include 'header.html' %}


   <!-- Contact Start -->

   <div class="container-fluid py-5">

      <div class="container py-5">

         <div class="text-center mb-5">

                        <h5 class="text-primary text-uppercase mb-3" style="letter-spacing:
5px;">Login</h5>

            <h1></h1>

         </div>

         <div class="row justify-content-center">

            <div class="col-lg-8">

               <div class="contact-form bg-secondary rounded p-5">

                  <div id="success"></div>

                              <form name="sentMessage" id="contactForm" action="addlogin"
method="POST" novalidate="">

                     {% csrf_token %}

                     <div class="control-group">

                              <input type="email" class="form-control border-0 p-4" id="Email"
name="email" placeholder="Enter your Email" required="required" data-validation-required-
message="Please enter your name" />

                        <p class="help-block text-danger"></p>

                     </div>

                     <div class="control-group">

                        <input type="password" class="form-control border-0 p-4" id="Password"
name="password" placeholder="Enter your Password" required="required" data-validation-
required-message="Please enter your email" />

                        <p class="help-block text-danger"></p>
```

```
                    </div>

                    <div class="text-center">
                                <button class="btn btn-primary py-3 px-5" type="submit"
id="Submit">Login</button>
                    </div>
                  </form>
                </div>
              </div>
            </div>
          </div>
    </div>
    <!-- Contact End -->



    <!-- Contact Javascript File -->
    <script src="mail/jqBootstrapValidation.min.js"></script>
    <script src="mail/contact.js"></script>

    <!-- Template Javascript -->
    <script src="js/main.js"></script>
</body>

</html>
{% include 'footer.html' %}
{% endblock %}
```

Viewassignsub.html

```
{% load static %}
{% block content %}
{% include 'header.html' %}
```

```
<!-- Contact Start -->
<div class="container-fluid py-5">
  <div class="container py-5">
    <div class="text-center mb-5">
        <h5 class="text-primary text-uppercase mb-3" style="letter-spacing: 5px;">Login</h5>
      <h1></h1>
    </div>
    <div class="row justify-content-center">
      <div class="col-lg-8">
        <div class="contact-form bg-secondary rounded p-5">
          <div id="success"></div>
            <form name="sentMessage" id="contactForm" action="addlogin" method="POST" novalidate="">
            {% csrf_token %}
            <div class="control-group">
              <input type="email" class="form-control border-0 p-4" id="Email" name="email" placeholder="Enter your Email" required="required" data-validation-required-message="Please enter your name" />
              <p class="help-block text-danger"></p>
            </div>
            <div class="control-group">
              <input type="password" class="form-control border-0 p-4" id="Password" name="password" placeholder="Enter your Password" required="required" data-validation-required-message="Please enter your email" />
              <p class="help-block text-danger"></p>
            </div>

            <div class="text-center">
              <button class="btn btn-primary py-3 px-5" type="submit" id="Submit">Login</button>
            </div>
```

```
                    </form>
                  </div>
                </div>
              </div>
            </div>
          </div>
    <!-- Contact End -->



    <!-- Contact Javascript File -->
    <script src="mail/jqBootstrapValidation.min.js"></script>
    <script src="mail/contact.js"></script>

    <!-- Template Javascript -->
    <script src="js/main.js"></script>
</body>

</html>
{% include 'footer.html' %}
{% endblock %}
```

Question_type.html

```
{% include 'header.html' %}
<br>
<br>
<br>
<br>
<center>
  <section class="contact_section  long_section">
    <div class="container">
```

```
    <div class="row">
      <div class="col-md-12">
        <div class="form_container">
          <div class="heading_container">
            <h2>
              Select the Question type
            </h2>
          </div>
          <input type="hidden" name="subject_name" value="{{ subject_name }}">

          <a href="{% url 'normal_question' subject_name %}" class="btn btn-primary">Normal
Question</a>
          <a href="{% url 'mcq' subject_name %}" class="btn btn-primary">MCQ</a>
        </div>
      </div>
    </div>
  </div>

      </Center>

            {% include 'footer.html' %}
```

Studentresult.html

```
{% include 'header.html' %}

<style>
  .result-container {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    margin-top: 50px;
```

```
  }
</style>

<section class="contact_section long_section">
  <div class="container">
    <div class="row">
      <div class="col-md-12">
        <div class="form_container">
          <div class="heading_container mb-5">
            <h2 class="text-center">Test Result</h2>
          </div>

          <div class="result-container">
            <table class="table table-striped">
              <thead>
                <tr>
                  <th scope="col">Sl No</th>
                  <th scope="col">Subject</th>
                  <th scope="col">Normal Questions</th>
                  <th scope="col">MCQ</th>
                  <th scope="col">Total</th>
                </tr>
              </thead>
              <tbody>
                {% if results %}
                  {% for subject, data in results.items %}
                    <tr>
                      {% if data.normal or data.mcq %}
                      <td>{{ forloop.counter }}</td>
                      <td>{{ subject }}</td>
```

```
            {% if data.normal %}
            <td>{{ data.normal }}/100</td>
            {% else %}
            <td>N/A</td>
            {%endif%}
            {% if data.mcq %}
            <td>{{ data.mcq }}/10</td>
            {% else %}
            <td>N/A</td>
            {%endif%}
            <td>{{ data.total }}/110</td>
            {% else %}
            <td>N/A</td>
            <td>N/A</td>
            <td>N/A</td>
            <td>N/A</td>
            <td>N/A</td>
            {% endif %}
          </tr>
        {% endfor %}
      {% else %}
        <tr>
          <td colspan="5" class="text-center">No results available</td>
        </tr>
      {% endif %}
    </tbody>
  </table>
  <div class="text-center">
  <button class="btn btn-primary" onclick="downloadPDF()">Download PDF</button>
  </div>
```

```
          </div>

        </div>
      </div>
    </div>
  </div>
</section>

<script>
  function downloadPDF() {
    const { jsPDF } = window.jspdf;
    const doc = new jsPDF();

    // Title
    doc.setFont("helvetica", "bold");
    doc.setFontSize(16);
    doc.text("Test Results", 80, 10);

    // Define table columns and rows
    const headers = [["Sl No", "Subject", "Normal Questions", "MCQ", "Total"]];
    const data = [];

    // Select table rows dynamically
    document.querySelectorAll("tbody tr").forEach((row, index) => {
      const cols = row.querySelectorAll("td");
      if (cols.length === 5) {
        data.push([
          index + 1,
          cols[1].innerText,
          cols[2].innerText,
```

```
        cols[3].innerText,
        cols[4].innerText
      ]);
    }
  });

  // Add table to PDF
  doc.autoTable({
    head: headers,
    body: data,
    startY: 20,
  });

  // Download the PDF
  doc.save("test_results.pdf");
 }
</script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf/2.5.1/jspdf.umd.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/jspdf-autotable/3.5.25/jspdf.plugin.autotable.min.js"></script>
{% include 'footer.html' %}
```

CHAPTER 12

REFERENCES

[1]. Sun, J., Zhang, Y., Li, X., & Wang, Z. (2020). Automatic Question Generation using NLP and Deep Learning. *IEEE Transactions on Learning Technologies*, 13(2), 356-367.

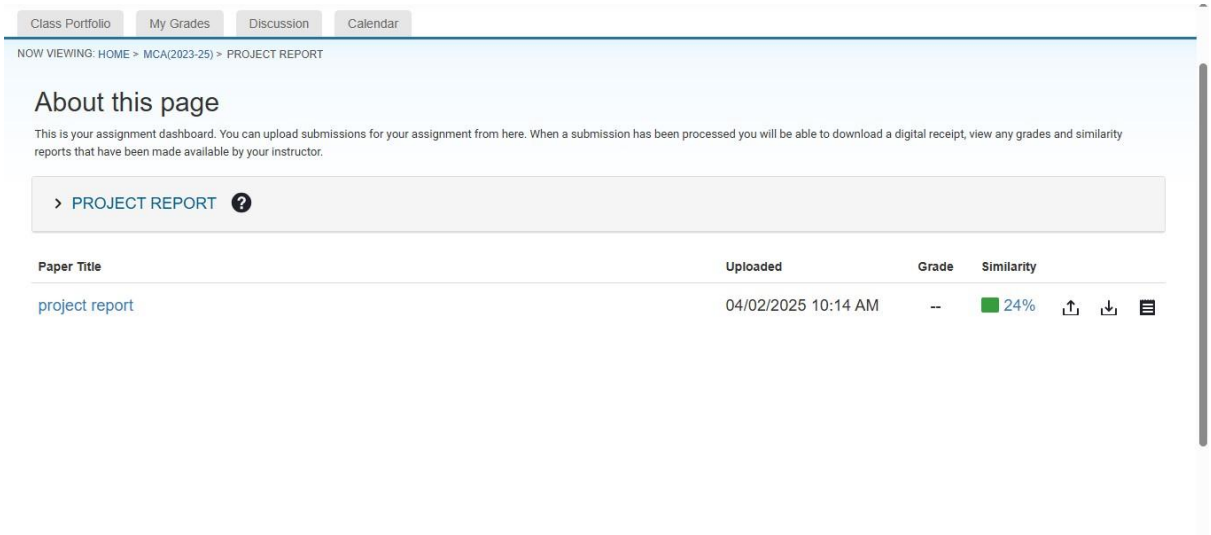
[2]. Xu, X., Wang, H., & Li, Y. (2018). Dual Learning for Visual Question Generation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 85-94.

[3]. Raynaud, T., Dupont, S., & Leclerc, M. (2021). Thematic Question Generation over Knowledge Bases. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 1023-1035.

[4]. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770-778.

[5]. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3), 211-252.

REPORT RESULT ANALYSIS:



turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Abithamol C F
Assignment title:	PROJECT REPORT
Submission title:	project report
File name:	AABIMAIN_merged.pdf
File size:	4.15M
Page count:	90
Word count:	14,291
Character count:	90,986
Submission date:	02-Apr-2025 04:44AM (UTC+0000)
Submission ID:	2632685954

PROJECT REPORT
On
PAPERPRO – QUESTION GENERATOR
Submitted in partial fulfilment for the award of the degree
of
Master of Computer Applications
By
ABITHAMOL C F (MLM23MCA-2003)
Under the Guidance of
KUKKUMOL THOMAS
(Assistant Professor, Department of Computer Applications)
DEPARTMENT OF COMPUTER APPLICATIONS
MANGALAM COLLEGE OF ENGINEERING, ETTUMANOOR
(Affiliated to APJ Abdul Kalam Technological University)
APRIL 2025

Copyright 2025 Turnitin. All rights reserved.