SECURED AUTOMATIC QUESTION PAPER GENERATION

A Project Report
Submitted
in Partial Fulfilment of the Requirements
for the Degree of

BACHELOR OF TECHNOLOGY

in
Information Technology
by

Akash Pandey (2107340130002)	**Krishna Pratap Singh** (2107340130029)
Rajnish Chaube (2107340130049)	**Ranjeet Singh** (2107340130052)

Under the Supervision of

Dr. Vibhash Yadav

(Associate Professor)

RAJKIYA ENGINEERING COLLEGE BANDA, ATARRA, BANDA

to the

Department of Information Technology

DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, LUCKNOW, UTTAR PRADESH

May, 2025

DECLARATION

We hereby declare that the work presented in this report entitled "SECURED AUTOMATIC QUESTION PAPER GENERATION", was carried out by us. We have not submitted the matter embodied in this report for the award of any other degree or diploma from any other University or Institute.

We have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, and results, that are not our original contributions. We have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

We affirm that no portion of our work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, we shall be fully responsible and answerable.

Submitted By: **Candidate's Signature:**

Akash Pandey (2107340130002)

Krishna Pratap Singh (2107340130029)

Rajnish Chaube (2107340130049)

Ranjeet Singh (2107340130052)

CERTIFICATE

Certified that **Akash Pandey** (2107340130002), **Krishna Pratap Singh** (2107340130029), **Rajnish Chaube** (2107340130049), and **Ranjeet Singh** (2107340130052) has carried out the research work presented in this major project report entitled **"SECURED AUTOMATIC QUESTION PAPER GENERATION"** for the award of Bachelor of Technology from Dr. A.P.J. Abdul Kalam Technical University, Lucknow under our supervision. The major project report embodies results of original work, and studies are carried out by the students themselves, and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature:

Dr. Vibhash Yadav

(Associate Professor)

Department of Information Technology

Rajkiya Engineering College Banda, Atarra

Date:

Place:

ABSTRACT

Creating question papers by hand has always been a time-consuming and stressful task for teachers and exam coordinators. It not only takes a lot of effort but is also prone to errors and repetition of questions. More importantly, traditional methods of paper setting are vulnerable to security risks such as unauthorized access, leaks, and tampering. These issues can lead to serious consequences like rescheduling exams and loss of trust in the examination system.

To overcome these challenges, this project introduces a secure and automated question paper generation system. The goal is to make the paper-setting process faster, smarter, and safer by using technology. Our system uses intelligent techniques to select questions based on criteria like difficulty level, marks, and Course Outcomes (COs). It ensures that the generated question paper is balanced, fair, and different each time.

Once the paper is generated, it is automatically converted into a PDF and then protected using strong encryption methods like AES (Advanced Encryption Standard). To make sure the document hasn't been changed or tampered with, a digital signature or hash code is applied using either MD5 or SHA-256 hashing algorithms. These security features ensure both confidentiality and integrity of the question papers.

The system is developed using Python and has a simple and user-friendly Graphical User Interface (GUI) built with Tkinter and ttkbootstrap. This makes it easy for non-technical users such as teachers and administrative staff to operate the software without needing any special training.

Keywords: Automated Question Paper Generation, Security, AES encryption, Confidentiality and Integrity, Graphical User Interface, Difficulty level and Course Outcomes.

ACKNOWLEDGEMENT

We, the team behind *Secured Automatic Question Paper Generation* - **Akash Pandey, Krishna Pratap Singh, Rajnish Chaube, and Ranjeet Singh**, would like to express our deepest gratitude to **Dr. Vibhash Yadav** for his invaluable guidance throughout the course of this project. His expertise, patience, and insights played a crucial role in its successful completion. His constructive feedback and unwavering support kept us motivated and helped refine our approach.

We also extend our heartfelt thanks to the entire **Information Technology Department** for their support and collaboration, making this learning experience truly enriching and rewarding. Their encouragement and willingness to assist us whenever needed made a significant difference.

Additionally, we are profoundly grateful to our **families and friends**, whose constant encouragement, unfailing love, and sincere prayers have been a pillar of strength throughout this journey. Their unwavering support has been a source of motivation during challenging times.

Lastly, we extend our appreciation to our **department staff members and friends**, who have contributed to the successful completion of this internship. Their guidance, encouragement, and willingness to help whenever needed have been invaluable.

Signature:

Akash Pandey (2107340130002)

Krishna Pratap Singh (2107340130029)

Rajnish Chaube (2107340130049)

Ranjeet Singh (2107340130052)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

Acronym	**Full Form**
AES	Advanced Encryption System
AQPGS	Automatic Question Paper Generation System
CBC	Cipher Block Chaining
CO	Course Outcomes
GUI	Graphical User Interface
MD	Message Digest
SHA	Secure Hash Algorithm

CHAPTER 1

INTRODUCTION

1.1. BACKGROUND

In the age of digital transformation, the educational sector is undergoing rapid change, with institutions adopting online learning, digital assessments, and smart campus solutions. Among these, one of the most critical yet vulnerable processes are the preparation and distribution of examination question papers. Traditionally, this process involved significant manual effort and physical security, including locked rooms, physical invigilation, and courier systems. However, these methods are increasingly becoming obsolete and risky.

Unauthorised access of question papers has become alarmingly frequent across various boards, universities, and even national-level competitive exams. Such information breach not only undermine the credibility of the academic institution but also damage student morale and trust. In severe cases, these leaks require the re-conducting of entire exams, incurring financial and administrative losses and wasting student's time.

Thus, the need arises for a solution that can ensure the confidentiality, integrity, and security of question papers from the moment they are generated until they are used in an exam hall.

1.2. MOTIVATION

The motivation behind this project arises from recurring incidents of exam malpractices due to question paper’s unauthorised access and tampering. These incidents often point to flaws in the manual handling process.

There have been numerous real-world examples ranging from CBSE board paper leaks in India to university examination compromises in multiple countries which have shown that secure document handling is not just desirable but essential. Professors and staff members tasked with paper-setting and examination duties are often not trained in security. They require tools that are easy to use, fast, and reliable without compromising on security.

This is where this project, "Secure Question Paper Generation using AES Encryption MD5 and SHA-256" plays a significant role. It aims to address both security and usability, providing a powerful tool that secures question papers at the digital level through industry-standard cryptographic techniques, while still being user-friendly through a GUI-based interface.

1.3. OBJECTIVE

The objective of this project is to design and develop a system that guarantees safe, intelligent, and user-friendly generation of question papers, while ensuring confidentiality and integrity.

1.4. SCOPE OF THE PROJECT

The project primarily focuses on academic institutions like schools, colleges, and universities where question paper handling is a sensitive task.

Functional Scope:

a) Accept question banks as text files with structured formatting.

b) Select questions based on parameters like CO (Course Outcomes), difficulty level, and marks.

c) Generate a well-formatted question paper file using reportlab.

d) Apply AES encryption to protect the generated question paper.

e) Generate hash values (MD5 or SHA-256) for integrity verification.

f) Include GUI for all operations: question selection, encryption, decryption, hash verification.

Non-Functional Scope:

a) Ensure fast processing and file handling (suitable for low-end machines).

b) Maintain modularity so that hashing algorithms can be replaced.

c) Usable on Windows and Linux environments where Python and required libraries are installed.

1.5. PROBLEM STATEMENT

How can we design a system that automatically generates and secures question papers, ensuring both confidentiality and integrity, while still being easy to use by non-technical academic staff.

1.6. PROPOSED SOLUTION

The proposed system is divided into two key modules, each performing specific tasks:

Module 1: Question Paper Generation with AES Encryption

a) Take input from a structured question bank (text file).

b) Filter questions based on Course Outcome (CO), difficulty, and marks.

c) Use logic to ensure:

   i. Balanced CO distribution.

   ii. At least 20% easy and 20% hard questions.

   iii. User specifies total number of questions.

d) Generate a question paper with:

   i. College name

   ii. Date and title

   iii. Question table

   iv. Difficulty distribution chart

e) Apply Fernet-based AES encryption to the question paper.

f) Save the encryption key securely as .key file.

Module 2: Securing Question Paper using MD5 and SHA-256

a) Encrypt any user-selected question paper.

b) Generate an MD5 or SHA-256 hash of the original question paper.

c) Allow decryption of encrypted files.

d) Verify integrity by comparing original and current hashes.

e) GUI options:

   i. Select file to encrypt

   ii. Decrypt encrypted file

   iii. Verify file integrity

These modules are designed independently but follow a common interface and encryption mechanism, making them interoperable.

1.7. TECHNOLOGIES USED

Technology	**Purpose**
Python	Core development language
Tkinter	GUI interface
ttkbootstrap	Styled widgets and GUI themes
hashlib	MD5 and SHA-256 hashing
PyCrypto	AES encryption and decryption
reportlab	PDF generation
matplotlib	Graphing (difficulty distribution)
PyPDF2	PDF password protection and page handling
numpy	Weighted random selection of questions
re (Regex)	Parsing question bank structure

1.8. ADVANTAGES OF THE PROPOSED SYSTEM

a) **Security**: Ensures end-to-end encryption and integrity.

b) **Automation**: Reduces human error in question paper preparation.

c) **User Experience**: GUI makes it accessible for faculty and exam controllers.

d) **Extensibility**: Modules can be enhanced for biometric login, audit trails, or blockchain.

e) **Reliability**: Detects even single-bit changes in the file.

1.9. CHALLENGES ENCOUNTERED

During the development of this project, certain challenges were identified:

a) Ensuring that the encrypted file, if modified even slightly, fails hash verification.

b) Handling file corruption gracefully during decryption.

c) Making GUI intuitive without cluttering with too many buttons or technical options.

d) Balancing ease-of-use with security complexity.

CHAPTER 2

LITERATURE REVIEW

2.1. INTRODUCTION TO CRYPTOGRAPHIC HASH FUNCTIONS

Cryptographic hash functions serve as fundamental building blocks in modern information security systems. These mathematical algorithms take an input (or "message") and return a fixed-size string of bytes, typically a digest that appears random. The output, known as the hash value or message digest, should be unique to the input data, making hash functions crucial for data integrity verification, password storage, digital signatures, and authentication protocols (Menezes et al., 1996).

The ideal cryptographic hash function possesses five key properties:

a) Deterministic - The same input always produces the same hash.

b) Quick Computation - The hash value should be computable quickly for any given input.

c) Pre-image Resistance - It should be computationally infeasible to reverse the hash function to recover the original input.

d) Small Changes Cause Avalanche Effect - A minor modification to the input should drastically change the output hash.

e) Collision Resistance - It should be extremely difficult to find two different inputs that produce the same hash output.

Among the various hash functions developed over the years, the MD (Message Digest) family and SHA (Secure Hash Algorithm) family have been most widely studied and implemented. This literature survey examines the evolution, strengths, weaknesses, and modern enhancements of these algorithms, with particular focus on MD5 and SHA-256. Additionally, we explore the application of automated question paper generation systems in educational technology, analysing various algorithmic approaches and their effectiveness.

2.2. HISTORICAL DEVELOPMENT OF HASH FUNCTIONS

2.2.1. Early Hash Functions and the MD Family

The development of cryptographic hash functions began in the late 1970s with the creation of hash functions based on block cipher designs. However, the first dedicated cryptographic hash function was MD2, developed by Ronald Rivest in 1989 (Rivest, 1992). MD2 was designed for 8-bit machines and produced a 128-bit hash value. While innovative for its time, cryptanalysts soon discovered vulnerabilities, leading to its deprecation.

Rivest subsequently developed MD4 in 1990 as a more efficient alternative for 32-bit systems (Rivest, 1992). MD4 introduced significant speed improvements but was found to have serious cryptographic weaknesses within years of its introduction. Notably, Dobbertin (1996) demonstrated practical collision attacks against MD4, showing that collisions could be found in under a minute on typical hardware of that era.

These developments led to the creation of MD5 in 1991 as a strengthened version of MD4 (Rivest, 1992). While retaining the 128-bit output size, MD5 incorporated additional security measures described as "safety belts" by Rivest. For nearly a decade, MD5 was widely adopted across internet protocols, software distribution, and security systems. However, as computational power increased and cryptanalysis advanced, researchers began identifying vulnerabilities in MD5 as well.

2.2.2. The SHA Family and Evolution to SHA-256

Recognizing the need for more secure hash functions, the National Institute of Standards and Technology (NIST) developed the Secure Hash Algorithm (SHA) family. The original SHA (now called SHA-0) was published in 1993 but was quickly withdrawn due to undisclosed flaws (NIST, 1995). Its replacement, SHA-1, became widely adopted in security protocols like TLS and SSL, as well as in version control systems like Git.

However, theoretical attacks against SHA-1 began emerging in the early 2000s, culminating in Google's practical collision demonstration in 2017 (Stevens et al., 2017). This prompted the development and standardization of the SHA-2 family in 2001, which includes SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, and SHA-512/256 (NIST, 2001). Among these, SHA-256 has emerged as the most widely implemented due to its balance between security and computational efficiency.

The SHA-3 standard, based on the Keccak algorithm, was introduced in 2015 as an alternative rather than replacement for SHA-2 (NIST, 2015). Unlike the Merkle-Damgard construction used in SHA-2, SHA-3 employs a sponge construction, providing resistance against certain types of cryptanalytic attacks that might affect SHA-256.

2.3. COMPREHENSIVE REVIEW OF MD5

2.3.1. Algorithmic Structure of MD5

The MD5 algorithm processes input messages in 512-bit blocks, producing a 128-bit hash value. The computation involves the following steps (Rivest, 1992):

a) Padding: The message is padded so its length is congruent to 448 modulo 512. Padding consists of a single '1' bit followed by '0' bits, ending with a 64-bit representation of the original message length.

b) Initialization: Four 32-bit registers (A, B, C, D) are initialized to fixed hexadecimal values derived from the sine function.

c) Processing: Each 512-bit block undergoes 64 rounds of processing using four auxiliary functions (F, G, H, I) that perform bitwise operations. Each round uses a different 32-bit constant derived from the sine function.

d) Output: After all blocks are processed, the contents of the four registers are concatenated to form the 128-bit hash value.

2.3.2. Vulnerabilities and Attacks on MD5

Despite its initial security claims, MD5 has been thoroughly compromised through various cryptanalytic attacks:

a) Collision Attacks: The first practical collision attack was demonstrated by Wang et al. (2004), who could generate colliding messages in hours on commodity hardware. This attack exploited differential paths in MD5's compression function (Wang et al., 2004).

b) Chosen-Prefix Collisions: Stevens et al. (2007) extended these attacks to create chosen-prefix collisions, enabling malicious actors to forge digital certificates (Stevens et al., 2007). The Flame malware famously used this technique in 2012 to spoof Microsoft code signatures.

c) Preimage Attacks: While more computationally intensive than collision attacks, Sasaki and Aoki (2009) demonstrated a theoretical preimage attack against MD5 (Sasaki & Aoki, 2009).

These vulnerabilities have led to MD5 being deprecated for most security applications, though it persists in non-cryptographic uses like file integrity checks where collision resistance isn't critical.

2.4. COMPREHENSIVE REVIEW OF SHA-256

2.4.1. Algorithmic Structure of SHA-256

SHA-256 processes messages in 512-bit blocks to produce a 256-bit hash value. Its structure includes (NIST, 2001):

a) Padding: Similar to MD5 but extends the message to be congruent to 448 modulo 512, then appends a 64-bit length.

b) Initialization: Eight 32-bit registers (a-h) are initialized to constant values derived from the square roots of the first eight prime numbers.

c) Message Schedule: Each 512-bit block is expanded into sixty-four 32-bit words using a message schedule that incorporates shift and XOR operations.

d) Compression Function: The core operation uses eight working variables updated through 64 rounds involving bitwise operations (AND, OR, XOR, NOT), modular addition, and fixed constants derived from cube roots of primes.

e) Finalization: The hash value is the concatenation of the eight registers after processing all blocks.

2.4.2. Security Analysis of SHA-256

While no practical attacks have broken SHA-256's preimage or collision resistance, several theoretical analyses have explored its limitations:

a) Reduced-Round Attacks: Researchers have found collisions for up to 24 rounds of SHA-256 (compared to its full 64 rounds), but extending these attacks remains computationally infeasible (Mendel et al., 2013).

b) Quantum Vulnerability: Grover's algorithm could theoretically reduce SHA-256's preimage resistance from 2^{256} to 2^{128} operations on a quantum computer (Bernstein, 2009). However, current quantum technology remains far from this capability.

c) Side-Channel Attacks: Implementations of SHA-256 may be vulnerable to timing or power analysis attacks if not properly secured (Mangard et al., 2007).

2.5. Modern Enhancements to SHA-256

Recent research has focused on enhancing SHA-256's security without significantly impacting its performance:

2.5.1. Chaotic S-Box Integration

Wang et al. (2021) proposed replacing SHA-256's nonlinear functions with chaotic S-boxes derived from the Lorenz system (Wang et al., 2021). Their approach:

a) Uses three 8×8 S-boxes generated via a firework optimization algorithm.

b) Implements linear P-boxes for diffusion.

c) Achieves better confusion/diffusion properties than standard SHA-256.

d) Maintains comparable computational efficiency.

2.5.2. Parallel Processing Implementations

With the rise of multi-core processors and GPUs, researchers have developed parallel implementations of SHA-256:

a) GPU Acceleration: Jiang et al. (2018) achieved 5.2 Gbps throughput on NVIDIA GPUs by optimizing message scheduling (Jiang et al., 2018).

b) Hardware Optimization: Martino and Gilardo (2020) designed an FPGA implementation achieving 8.9 Gbps while reducing power consumption by 23% (Martino & Gilardo, 2020).

These optimizations make SHA-256 practical for high-throughput applications like blockchain and big data processing.

2.6. AUTOMATED QUESTION PAPER GENERATION SYSTEMS

2.6.1. Traditional vs Automated Approaches

Traditional question paper generation involves manual selection of questions by educators, which is:

a) Time-consuming.

b) Prone to bias in question selection.

c) Difficult to ensure proper syllabus coverage.

d) Challenging to maintain consistency across multiple versions.

2.6.2. Algorithmic Approaches in AQPGS

Various algorithmic approaches have been employed in automated question paper generation:

Fuzzy Logic Systems

Mohandas et al. (2015) proposed a fuzzy logic-based question paper generation system designed to improve the adaptability and precision of question selection. In their model, each question in the database was assigned a membership value based on two parameters: difficulty level and topic relevance. This allowed the system to treat inputs as imprecise or fuzzy values rather than binary categories, which mirrors real-world academic grading more accurately.

Apriori Algorithm for Syllabus Mapping

In an extension of the fuzzy logic approach, Tendolkar et al. (2017) introduced a hybrid model that combined fuzzy logic inference with the Apriori algorithm to enhance syllabus coverage and efficiency in question paper generation. The Apriori algorithm was applied to the question bank to identify frequent patterns among questions that shared common learning objectives. This pattern mining technique enabled the system to recommend question sets that historically appeared together, thereby reinforcing conceptual integrity and learning outcome alignment.

Randomization Techniques

Dubey et al. (2020) developed a Python-based system centered on randomization strategies to enhance question diversity and reduce predictability in examination papers. The system employed weighted random selection, where each question carried a weight based on metadata such as difficulty level, course outcome relevance, and marks. This allowed the system to maintain a balanced representation of questions while introducing sufficient randomness to prevent repetition or bias.

CHAPTER 3

METHEDOLOGY

3.1. INTRODUCTION

The successful design of a secure automatic question paper generation hinges on the integration of cryptographic algorithms, structured question selection logic, and user-friendly design through GUI. This chapter provides a detailed technical exposition of how the project was built, the structure of the codebase, flow diagrams of data processing, and the purpose behind each design choice.

The overall system is composed of two interconnected modules:

a) A question paper generator that automatically selects questions from a structured question bank and uses AES encryption for confidentiality. This module is responsible for the automated selection and generation of question papers from a structured question bank. It employs a rule-based selection algorithm that takes into account multiple academic criteria, including the Course Outcome (CO) mapping, difficulty levels (easy, medium, hard), and mark distribution.

b) A Security system using MD5 and SHA-256 hashing for file integrity verification. the second module focuses on file integrity verification using cryptographic hashing algorithms namely MD5 (Message Digest 5) and SHA-256 (Secure Hash Algorithm 256-bit).

3.2. SYSTEM ARCHITECTURE OVERVIEW

Below is a high-level view of the system's architecture:

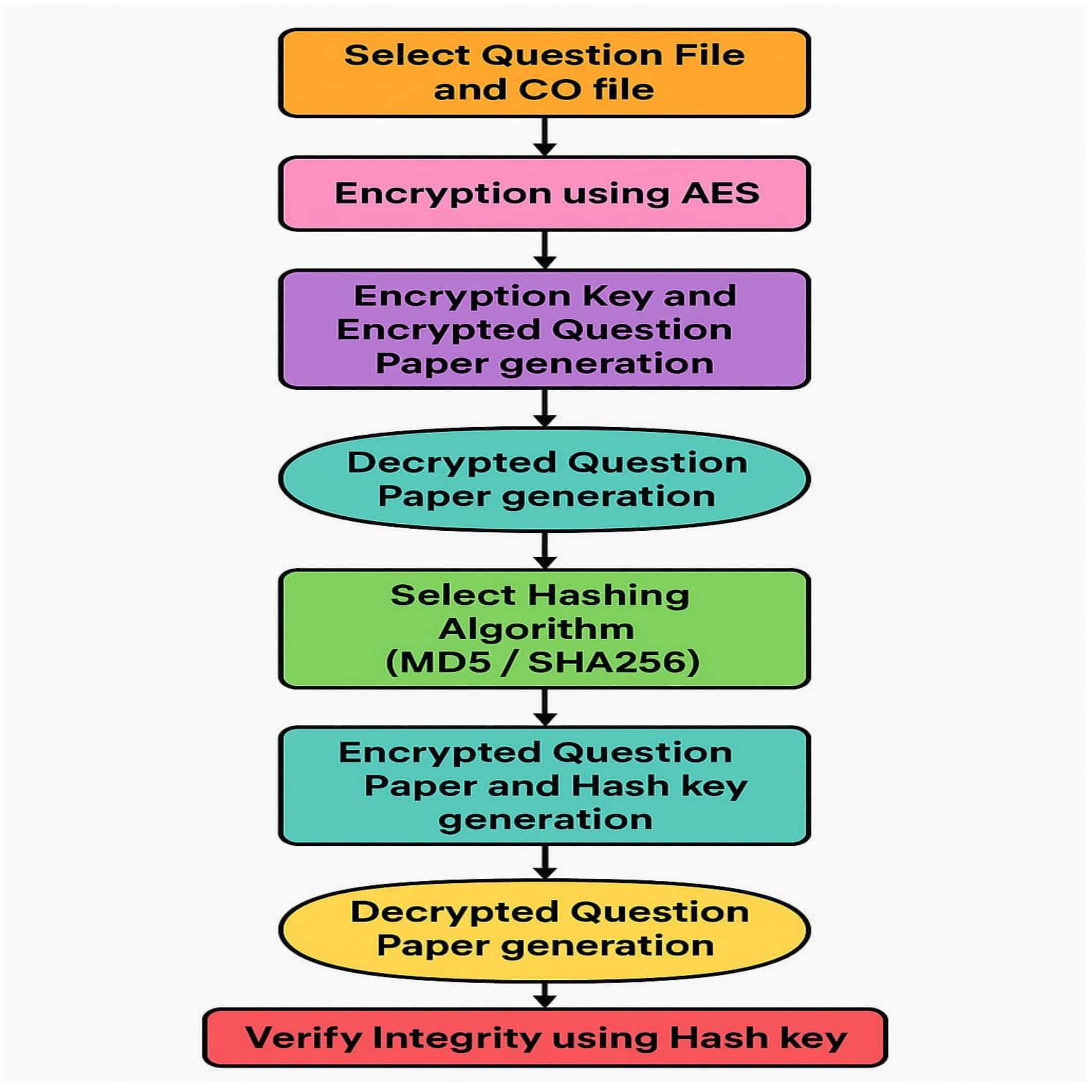

Select Question File and CO file
Encryption using AES
Encryption Key and Encrypted Question Paper generation
Decrypted Question Paper generation
Select Hashing Algorithm (MD5 / SHA256)
Encrypted Question Paper and Hash key generation
Decrypted Question Paper generation
Verify Integrity using Hash key

Fig. 3.2: Flowchart of the project

3.3. MODULE 1: QUESTION PAPER GENERATION WITH AES ENCRYPTION

This is the central component of the system. It is designed to read a structured question bank from a .txt file and select questions based on specific criteria, including Course Outcome (CO), difficulty level (categorized as Easy, Medium, or Hard), and marks weightage. The module ensures that user specified number of questions are selected, with a minimum of twenty percent questions each from the Easy and Hard categories. Furthermore, it guarantees balanced representation of all relevant CO.

3.3.1. Input File Structure

The question bank is stored in a .txt file, where each question is followed by a metadata line that specifies the marks, difficulty level, and course outcome. The format appears as follows:

Question Text Here

10:2:1

In this format, 10 denotes the marks assigned to the question, 2 indicates the difficulty level (with 1 = Easy, 2 = Medium, 3 = Hard), and 1 represents the Course Outcome number (e.g., CO-1). A regular expression is used to parse this structure, with the

$$\text{Pattern} = \#\#(.*?)\#\#\backslash s*([0-9]+)\#([0-9]+)\#([0-9]+)\#$$

3.3.2. Selection Logic

The system groups questions based on their CO using defaultdict. From each CO group, two questions are selected through a process of weighted random sampling where medium-difficulty questions are given higher probability weights. The weighting is handled using the logic:

weights.append(3 if q[2] == 2 else 1)

After selection, the system validates that at least twenty percent easy and twenty percent hard questions are present. If this condition is not met, the system replaces medium-difficulty questions with those of the required difficulty level to satisfy the constraints.

3.3.3. Output Generation

Once the question selection is finalized, a question paper is generated using the reportlab library. The document contains a table where each row includes the question number, question text, difficulty level, and CO number. Additionally, a graphical representation of question difficulty distribution is generated using matplotlib and embedded directly into the question paper.

3.3.4. Question Paper Encryption Using AES

After the question paper has been created, it is encrypted using Advanced Encryption Standard (AES) symmetric encryption to ensure confidentiality.

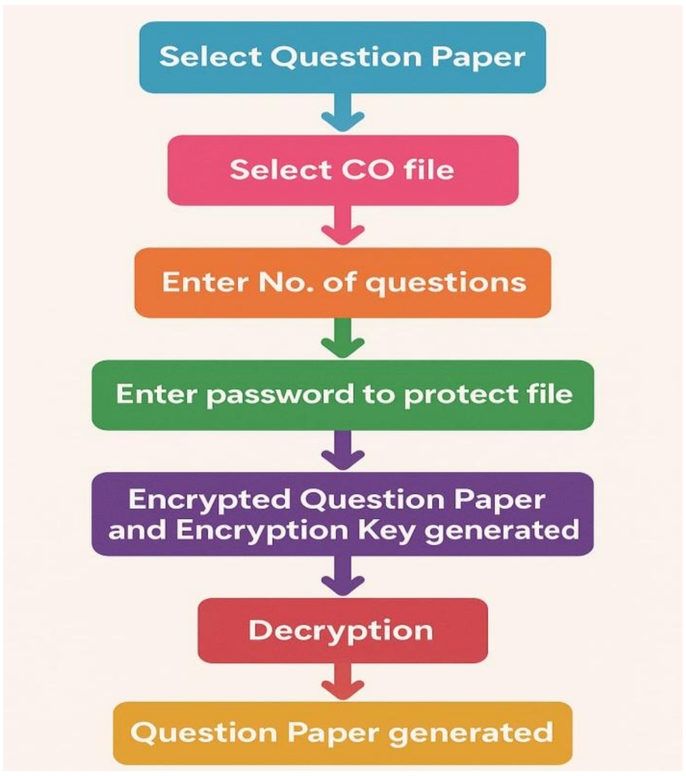


Fig. 3.3: Flowchart of module 1

3.4. MODULE 2: SECURING QUESTION PAPER USING MD5 AND SHA-256

This module features a GUI-based system that allows users to select a question paper and perform operations such as MD5 and SHA-256 hash generation, decryption, and integrity verification by comparing the stored and computed hash values. This enhanced security makes the module ideal for use by sensitive organizations such as government agencies, examination boards, and recruitment authorities.

3.4.1. Hash Generation Logic

To compute the MD5 hash, the system reads the question paper in chunks and updates the hash function iteratively:

```
hash_func = hashlib.md5()
while chunk := f.read(4096):
hash_func.update(chunk)
```

To compute the SHA-256 hash, the system reads the question paper in chunks and updates the hash function iteratively:

```
hash_func = hashlib.sha256()
while chunk := f.read(4096):
hash_func.update(chunk)
```

3.4.2. GUI Workflow (Tkinter + ttkbootstrap)

This module includes a graphical interface built using Tkinter and ttkbootstrap. The interface contains buttons labeled "Encrypt", "Decrypt", and "Verify". The user selects the desired file, initiates an operation, and receives status feedback through message boxes.

3.5. GUI INTEGRATION

The complete system is integrated into a GUI, ensuring user-friendliness and eliminating the need for command-line interaction. This design supports quicker adoption by academic institutions and enhances accessibility for non-technical users. The interface uses the "Cyborg" theme from ttkbootstrap and includes buttons labelled "Select Question File," "Select CO File," "Generate & Encrypt Paper," and "Decrypt Paper."

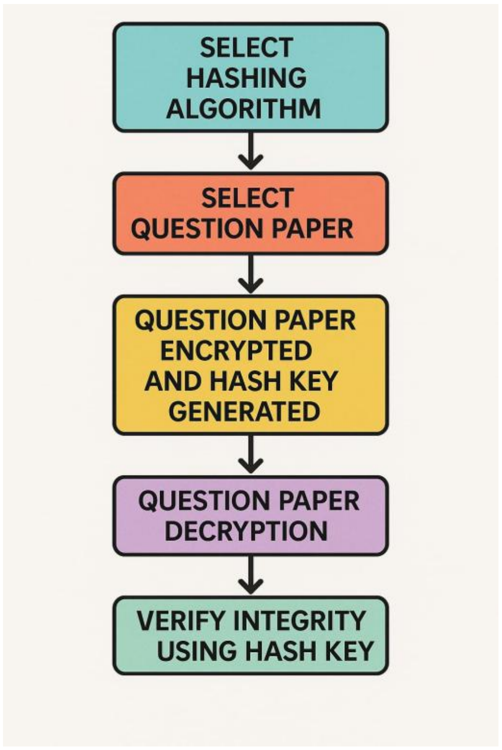

SELECT HASHING ALGORITHM
SELECT QUESTION PAPER
QUESTION PAPER ENCRYPTED AND HASH KEY GENERATED
QUESTION PAPER DECRYPTION
VERIFY INTEGRITY USING HASH KEY

Fig. 3.5: Flowchart of module 2

3.6. SECURITY WORKFLOW

The security procedure follows a structured sequence: first, a question paper question paper is generated. Next, the file is encrypted using AES. A hash (either MD5 or SHA-256) is then generated. Both the encrypted question paper and its corresponding hash are saved. Upon decryption, the file is decrypted and it’s integrity is verified. If a mismatch between the original and computed hash is detected, an alert is triggered. In such cases, a warning is displayed.

3.7. INTEGRITY VERIFICATION

The system computes a hash from the received question paper and compares it to the original hash. This is done using the following logic:

```
computed_hash = generate_hash(received_Question paper, algo)
if computed_hash == original_hash:
    return True
```

Any alteration to the file even a one-byte change results in a different hash. This guarantees the detection of tampering, authenticates the version of the file, and enables traceability in the event of a breach.

CHAPTER 4

RESULTS AND DISCUSSION

This chapter presents the results, and the subsequent discussion related to the implementation of the Secure Question Paper Generation project, including the integration of all two modules: secure question paper generation with AES encryption and question paper security through SHA-256 and MD5. The primary goal of this system is to ensure the integrity and confidentiality of question papers, by using advanced cryptographic techniques. This chapter will provide an in-depth analysis of the system's functionality, performance, and security features. It will also include a discussion of the results from various tests conducted to verify the effectiveness of each module and the system.

In this chapter, we evaluate the performance and security of the system from several perspectives, such as system efficiency, encryption strength and user interface. The results will be discussed in relation to the initial goals set for the project, along with potential areas of improvement and limitations observed during the system's implementation.

4.1. EVALUATION CRITERIA FOR THE SYSTEM

To thoroughly assess the performance and security of the Secured Automatic Question Paper Generation project, several evaluation criteria were established. These criteria were chosen to ensure that the system met both the functional and non-functional requirements, particularly in terms of security, usability, and performance.

a) Performance

Performance evaluation includes testing the time taken to generate a secure question paper, the time required for hashing and encryption of question paper files, and the scalability of the system. This includes handling large question papers or question papers and assessing how the system performs under various conditions, such as file size and encryption complexity.

b) Usability

The usability of the system was tested through user interaction with the graphical user interface (GUI). The ease of use, intuitive design, and responsiveness of the system were evaluated to ensure it is user-friendly, especially for exam controllers and professors who would use it.

c) Security

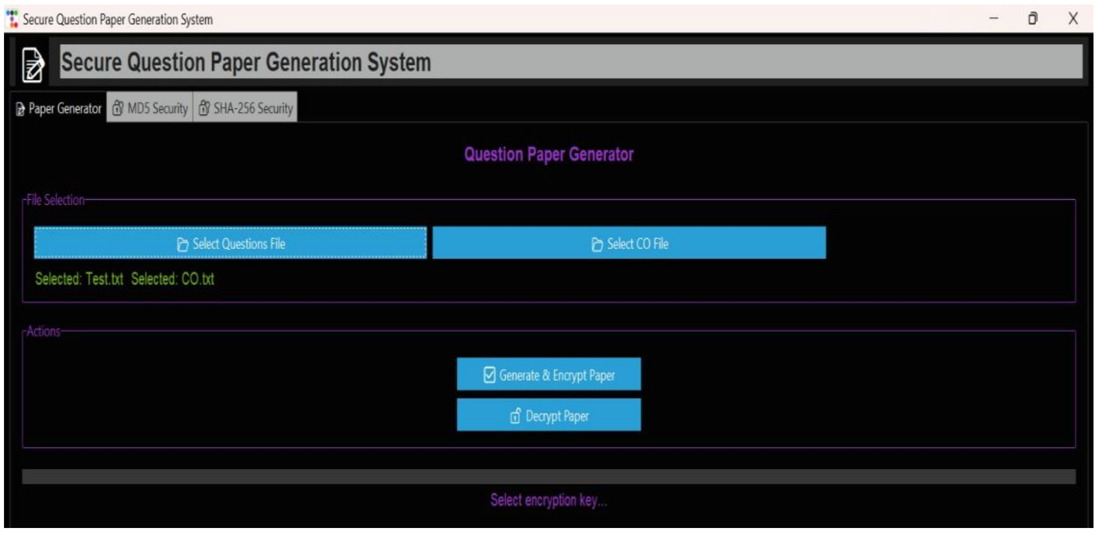
The primary focus of the system is to ensure both the confidentiality and integrity of the generated question papers. The security measures are evaluated based on the effectiveness of the cryptographic algorithms used, including the SHA-256 hashing algorithm for integrity verification and AES encryption for confidentiality.

d) Integration of Modules

One of the most important aspects of this project is the integration of the two modules (secure question paper generation with AES-based encryption and question paper integrity checking). This criterion evaluates how well these modules work together to create a workflow from generating the question paper to ensuring its security.

4.2. RESULTS FROM MODULE 1: SECURE QUESTION PAPER GENERATION

The first module of the system is designed to generate secure question papers automatically. This module is crucial for ensuring that question papers are both randomized and protected from unauthorized access. It was implemented with specific functionalities, including random question selection based on difficulty level, subject matter, and question type, ensuring that no two question papers are identical.



Selected: Test.txt Selected: CO.txt

Select encryption key...

Fig. 4.2(a): Question Paper Generation

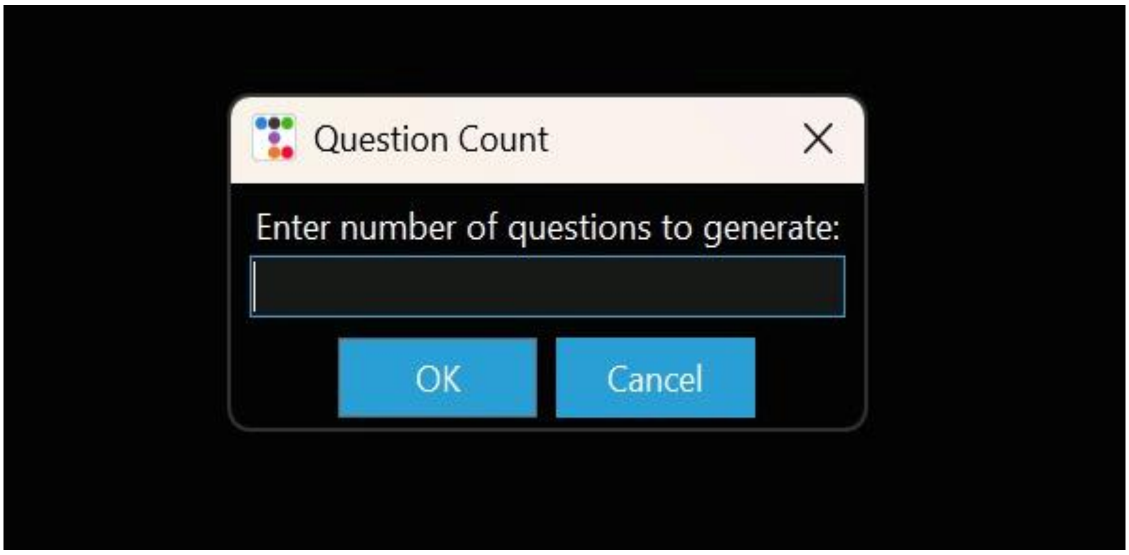


Fig. 4.2(b): Question Selection to Generate

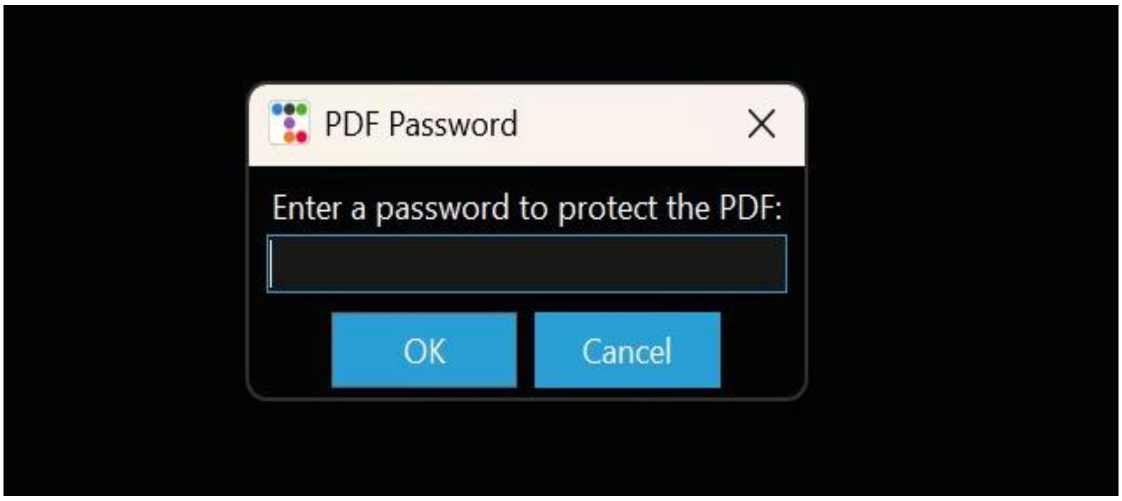
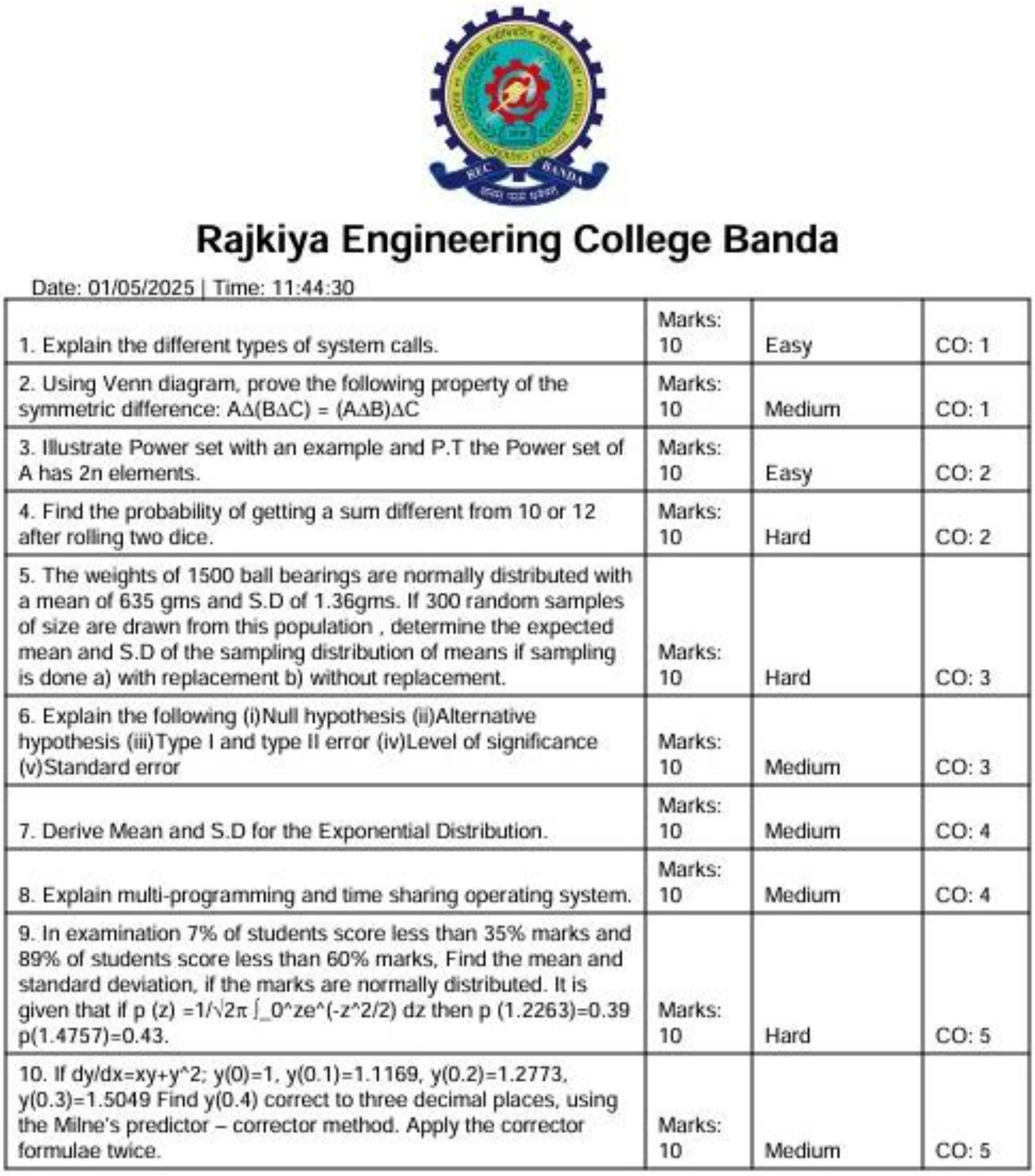


Figure 4.2(c): Password Protection using Fernet



Rajkiya Engineering College Banda

Date: 01/05/2025 | Time: 11:44:30

1. Explain the different types of system calls.	Marks: 10	Easy	CO: 1
2. Using Venn diagram, prove the following property of the symmetric difference: AΔ(BΔC) = (AΔB)ΔC	Marks: 10	Medium	CO: 1
3. Illustrate Power set with an example and P.T the Power set of A has 2n elements.	Marks: 10	Easy	CO: 2
4. Find the probability of getting a sum different from 10 or 12 after rolling two dice.	Marks: 10	Hard	CO: 2
5. The weights of 1500 ball bearings are normally distributed with a mean of 635 gms and S.D of 1.36gms. If 300 random samples of size are drawn from this population , determine the expected mean and S.D of the sampling distribution of means if sampling is done a) with replacement b) without replacement.	Marks: 10	Hard	CO: 3
6. Explain the following (i)Null hypothesis (ii)Alternative hypothesis (iii)Type I and type II error (iv)Level of significance (v)Standard error	Marks: 10	Medium	CO: 3
7. Derive Mean and S.D for the Exponential Distribution.	Marks: 10	Medium	CO: 4
8. Explain multi-programming and time sharing operating system.	Marks: 10	Medium	CO: 4
9. In examination 7% of students score less than 35% marks and 89% of students score less than 60% marks, Find the mean and standard deviation, if the marks are normally distributed. It is given that if p (z) =1/√2π ∫_0^ze^(-z^2/2) dz then p (1.2263)=0.39 p(1.4757)=0.43.	Marks: 10	Hard	CO: 5
10. If dy/dx=xy+y^2; y(0)=1, y(0.1)=1.1169, y(0.2)=1.2773, y(0.3)=1.5049 Find y(0.4) correct to three decimal places, using the Milne's predictor – corrector method. Apply the corrector formulae twice.	Marks: 10	Medium	CO: 5

Fig. 4.2(d): Generated Question Papers

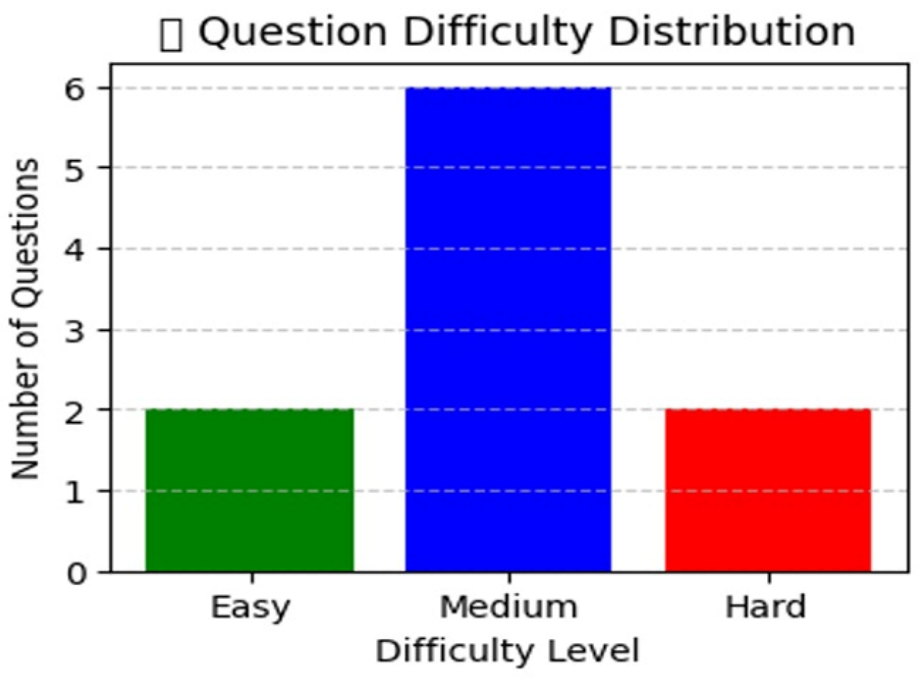


Fig.4.2(e): Question Difficulty Distribution

4.2.1. Performance of the Question Paper Generation

To evaluate the performance of this module, we conducted several tests with different numbers of questions, subjects, and difficulty levels. The system was able to generate different question papers each time.

4.2.2. Security of the Generated Question Papers

The question papers generated by the system were further secured by employing AES encryption and incorporating randomization techniques. The randomization of questions ensures that no two identical question papers are generated.

4.2.3. Analysis

A sample test was conducted to verify that the system generated question papers as expected. A set of predefined questions was input into the system, and the output was compared against the generated question papers. The system successfully randomized questions and adhered to the specified criteria for difficulty and subject.

4.3. RESULTS FROM MODULE 2: SECURING QUESTION PAPER USING SHA-256 AND MD5

The second module focuses on securing question paper through the use of hashing algorithms, specifically SHA-256 and MD5. This module ensures that the integrity of the generated question paper is maintained and that any tampering with the document can be detected by comparing hash values before and after transmission.

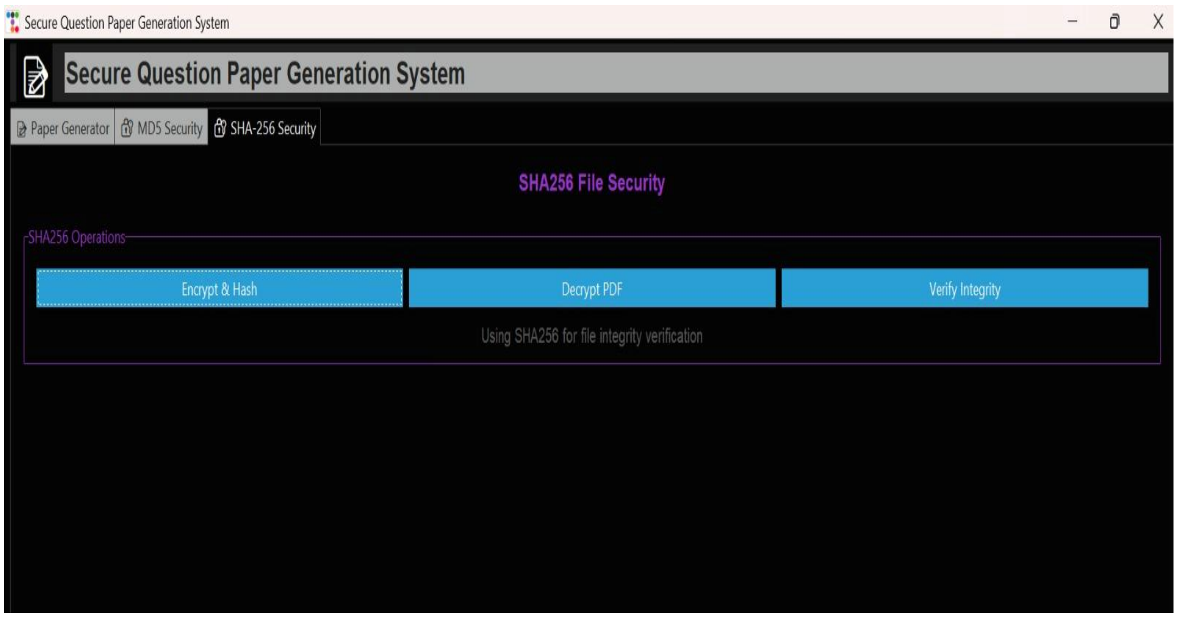

Secure Question Paper Generation System
Secure Question Paper Generation System
Paper Generator
MD5 Security
SHA-256 Security
SHA256 File Security
SHA256 Operations
Encrypt & Hash
Decrypt PDF
Verify Integrity
Using SHA256 for file integrity verification

Fig.4.3(a): Hashing using SHA-256

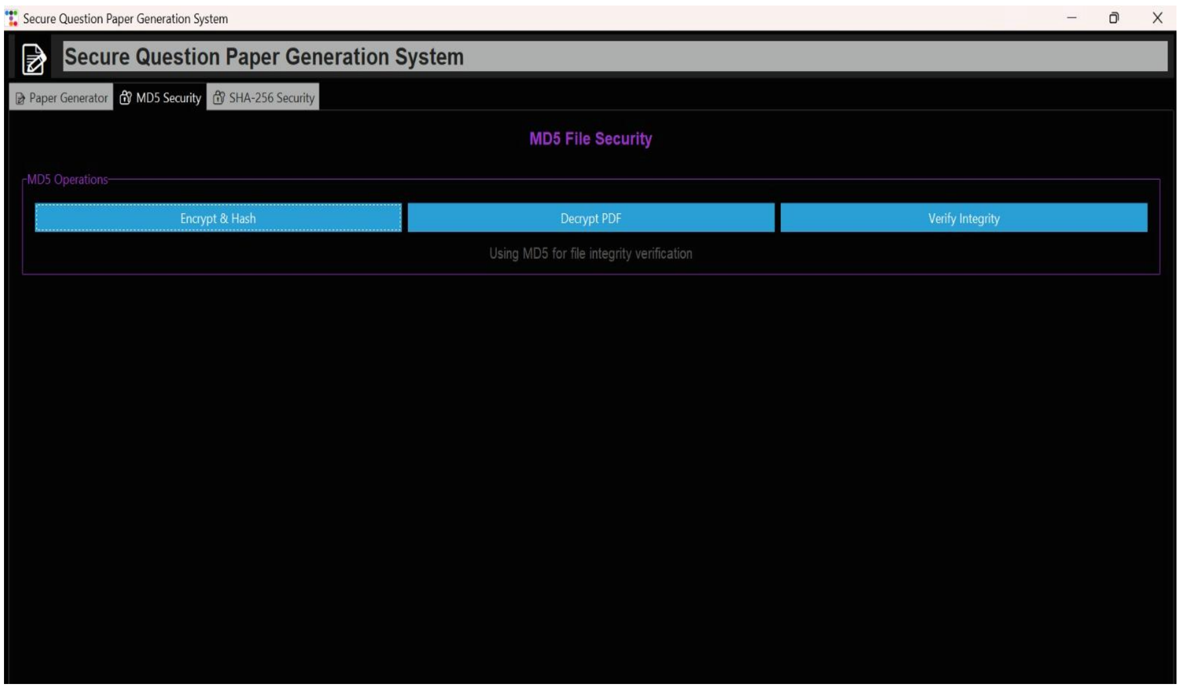

Secure Question Paper Generation System
Secure Question Paper Generation System
Paper Generator
MD5 Security
SHA-256 Security
MD5 File Security
MD5 Operations
Encrypt & Hash
Decrypt PDF
Verify Integrity
Using MD5 for file integrity verification

Fig.4.3(b): Hashing using MD-5

4.3.1. Integrity Verification

The system was tested to verify whether it correctly identified modifications to the question papers. First, the system calculated the hash of an unmodified question paper. Then, we intentionally modified the question paper by altering the content (e.g., adding or removing a question). The system correctly identified that the hash had changed, indicating that the file had been tampered with. This demonstrates the integrity verification capability of the system.

4.3.2. Integrity and Confidentiality

The encrypted question paper files were tested to ensure that both confidentiality and integrity were preserved. Upon decryption, the system re-calculated the MD5 and SHA-256 hash of the decrypted question paper and compared it with the original hash value. If the hash matched, it confirmed that the file had not been tampered.

In terms of confidentiality, AES encryption ensured that unauthorized users could not view the contents of the question paper. This was tested by attempting to open the encrypted question paper without the decryption key. The system correctly denied access, ensuring that the file remained confidential.

4.3.3. Security Analysis

The combination of AES encryption for confidentiality and MD5 or SHA-256 for integrity provided a robust security solution. SHA-256 provided a secure mechanism for detecting any tampering with the file. Together, these mechanisms ensured that the generated question papers were both confidential and untampered.

4.4. SYSTEM INTEGRATION AND OVERALL EVALUATION

The integration of two modules resulted in a fully functional and secure system for generating and securing question papers. The modules worked cohesively, with the first module generating secure question papers providing encryption through AES, the second module ensuring the integrity of the question papers through hashing.

The overall performance of the system was evaluated based on the time taken for each step, from generating the question paper to securing the question paper. The system performed efficiently for moderate file sizes, with acceptable encryption and hashing times.

CHAPTER 5

CONCLUSION & FUTURE SCOPE

5.1. CONCLUSION

The Secure Question Paper Generator project successfully achieved its goal of providing a comprehensive solution for ensuring both the confidentiality and integrity of exam-related documents, particularly question papers. This was accomplished by employing cryptographic techniques, including the SHA-256 and MD5 hashing algorithms, along with AES encryption. The system's development and implementation incorporated two main modules: Question Paper Generation with Security through AES Encryption, question paper integrity verification using SHA-256 and MD5, and. Each of these modules worked cohesively to address the critical concerns of confidentiality, integrity, and authenticity, ensuring that question papers remain secure from unauthorized access and tampering.

The core objective of this project was to create a system that could automatically generate secure question papers while providing mechanisms for verifying their integrity and maintaining confidentiality. This goal was met by implementing an automated question paper generation mechanism, which ensures that the papers are randomized and unique. The use of secure encryption methods and hashing algorithms further ensured that any attempts to alter the document would be immediately detected, preserving the integrity of the question papers.

The integrity of the question paper files was ensured using SHA-256 and MD5 hashing algorithms. SHA-256 was used to verify the authenticity of the documents, as it provides a higher level of security against collision attacks, ensuring that any tampering with the document would be detected. MD5, while faster, was shown to be less secure in comparison due to its vulnerability to collision attacks. Thus, SHA-256 was the preferred hashing method for integrity verification. The system was able to successfully detect any modifications made to the question papers, even if the changes were subtle, such as the addition or removal of a single character.

The AES encryption module provided robust confidentiality protection by ensuring that only authorized users could access the content of the question paper question papers. The AES encryption algorithm, with its 256-bit key, is highly resistant to brute-force attacks, ensuring that the question papers remained confidential even when transmitted over potentially insecure channels. The encryption time was acceptable for moderate file sizes, although performance could degrade for very large files. This trade-off between security and performance is a critical aspect to consider in the further development of the system.

A major highlight of the project was the integration of two modules into a cohesive system. The modules worked together, with the question paper generation process feeding directly into the question paper generation and encryption processes. The use of cryptographic hashing and encryption not only ensured the security of the documents but also made the system highly reliable in terms of detecting tampering and unauthorized access.

Despite the success of the project, several challenges were encountered during the development phase. One of the most notable challenges was optimizing the system for large file sizes. As the size of the question paper documents increased, the time taken for encryption and hashing also increased, which could potentially impact the user experience. Another challenge was ensuring that the system was both secure and user-friendly. While the cryptographic algorithms provided a high level of security, they also required a certain level of technical expertise to implement and use effectively. Simplifying the user interface and automating some of the processes could enhance the user experience, particularly for non-technical users.

Overall, the Secure Question Paper Generator project has successfully met its objectives, providing a secure, efficient, and user-friendly solution for generating and securing exam-related documents. The system demonstrates that the use of cryptographic techniques, such as SHA-256, MD5, and AES, can effectively address the security concerns related to exam paper management, particularly in terms of integrity verification and confidentiality protection. However, the system can be further optimized for large-scale use and performance improvements.

5.2. FUTURE SCOPE

While the Secure Question Paper Generator project has proven successful in providing a secure and efficient solution for exam paper generation, there remains significant potential for future enhancements and extensions. These improvements would focus on increasing the system's optimizing performance, enhancing security, and improving the user experience. The following sections outline several key areas for future developments.

5.2.1. Advanced Question Bank Management

a) **Database Integration**: Replace text files with a proper database system (SQL/NoSQL) for better question bank management

b) **Natural Language Processing**: Implement NLP techniques for automatic question categorization and difficulty assessment

c) **AI-powered Question Generation**: Incorporate AI to automatically generate new questions based on course outcomes

5.2.2. Multi-Factor Authentication for Enhanced Security

In the current system, access control is provided by basic role-based authentication. However, to enhance security, especially in environments where unauthorized access poses a serious risk, multi-factor authentication (MFA) can be integrated into the system. MFA would require users to authenticate themselves using multiple methods, such as:

a) Something they know (password),

b) Something they have (smartphone or authentication app), and

c) Something they are (biometric data such as fingerprints or facial recognition).

Implementing MFA would add an additional layer of security, making it more difficult for unauthorized individuals to access or tamper with exam-related documents. This could be particularly useful when exam papers are being stored or transmitted online, where there is a higher risk of unauthorized access.

5.2.3. Advanced Cryptographic Techniques

The security of the system can be further enhanced by exploring more advanced cryptographic techniques beyond the current implementation of SHA-256 and AES. While these algorithms provide strong protection, there are emerging encryption and hashing techniques that could offer additional security benefits:

a) **RSA Encryption**: RSA is a public-key encryption algorithm that can be used for encrypting the question papers with asymmetric encryption. This could be useful if the system needs to allow exam controllers and professors to securely exchange question papers without sharing a common key.

b) **Elliptic Curve Cryptography (ECC)**: ECC provides high security with smaller key sizes, making it more efficient than traditional RSA encryption. Implementing ECC could improve the system's performance while maintaining a high level of security.

Exploring these advanced techniques could further strengthen the system's security and make it more adaptable to emerging cryptographic standards.

5.2.4. Integration with Cloud Storage and Distributed Systems

As educational institutions increasingly adopt cloud-based solutions, integrating the Secure Question Paper Generator system with cloud storage platforms could offer significant advantages. Cloud integration would enable exam controllers and professors to store and access question papers remotely, ensuring that they are always accessible from any location. Furthermore, cloud storage offers scalable storage options, which can accommodate an increasing volume of question papers without requiring significant investment in on-premises infrastructure.

In addition to cloud storage, integrating the system with distributed computing frameworks could provide further scalability. By leveraging cloud-based computing resources, such as Amazon Web Services or Microsoft Azure, the system could process and generate question papers in parallel across multiple nodes, improving performance and reducing the risk of bottlenecks.

5.2.5. Improved User Interface and Experience

While the current system provides a basic graphical user interface (GUI), further improvements can be made to enhance the user experience. An intuitive and user-friendly interface is crucial, especially for non-technical users such as exam controllers and professors who may not be familiar with cryptographic principles. Some key improvements to the GUI could include:

a) **Simplified Workflow**: Streamlining the process of generating and securing question papers could make the system easier to use. A step-by-step wizard or guided process could help users navigate through the system's features more easily.

b) **Batch Processing**: Allowing users to generate and secure multiple question papers simultaneously could save time, especially during exam preparation. This could be achieved by adding batch processing capabilities to the system.

c) **Interactive Dashboards**: Implementing dashboards that display real-time information about the system's status, such as the number of generated question papers or the encryption process, could provide users with valuable insights into the system's operations.

5.2.6. AI-Powered Security Features

Incorporating artificial intelligence and machine learning algorithms could significantly enhance the security of the system. For example, anomaly detection algorithms could be used to detect unusual patterns of access to question papers or modifications to documents. These algorithms could identify potential security threats in real-time, alerting exam controllers or professors if suspicious activity is detected. Additionally, AI could be used to predict and prevent potential vulnerabilities, improving the system's overall security posture.

5.2.7. Compliance with Privacy Regulations

As data protection and privacy regulations become more stringent worldwide (e.g., GDPR in Europe), it is important that the Secure Question Paper Generator system complies with these regulations. Future versions of the system could include features that allow users to manage consent, track access, and audit actions taken on sensitive exam documents. This would ensure that the system aligns with legal requirements and protects users' data rights.

REFERENCES

(a) Sharvari N. Tendolkar, Rupali B. Shirke, Priyanka S. Mayekar, Santosh V. Jadhav. 1, 2, 3, 4 Finolex Academy of Management and Technology, Ratnagiri. International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor:6.887 Volume 5 Issue X, October 2017- Available at www.ijraset.com .

(b) Dubey Harish, Tamore Hardik, Padhi Sagar, Prof. Manisha Bharambe, Students, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India 4Professor, Dept. of Information Technology Engineering, SSJCOE, Maharashtra, India . International Research Journal of Engineering and Technology (IRJET) E-ISSN: 2395-0056 Volume: 07 ISSUE: MAY 2020 WWW.IRJET.NET P-ISSN: 2395-0072.

(c) Mojitha Mohandas, Aishwarya Chavan, Rasika Manjarekar and Divya Karekar, "Automated Question Paper Generator System", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 12, December 2015.

(d) Farhad Ahmed Sagar, "Cryptographic Hashing Functions- MD5", International Conference on Technology for Education, September 2016.

(e) Juan Wang, GE Liu, Yongqi Chen and Shu Wang, "Construction and Analysis of SHA-256 Compression Function Based on Chaos S-Box", College of Information and Communication Engineering, Harbin Engineering University, Harbin 150001, China,", IEEE Global Engineering Education Conference (EDUCON), mail: 1131814@s.hlju.edu.cn, December 2020.

(f) Noor Hasimah Ibrahim Teo, Nordin Abu Bakar and Moamed RezduanAbd Rashid, "Representing Examination Question Knowledge into Genetic Algorithm", IEEE Global Engineering Education Conference (EDUCON), 2014.

(g) Gauri Nalawade and Rekha Ramesh, "Automatic Generation of Question Paper from User Entered Specifications using a Semantically Tagged Question Repository", 2016 IEEE 8th International Conference on Technology for Education.

(h) Prateek Pisat, Shrimangal Rewagad, Devansh Modi and Ganesh Sawan, "Question Paper Generator and Answer Verifier", International Conference on

Energy, Communication, Data Analytics and Soft Computing (ICECDS-2017).

(i) Tejas Barot and Poornima Salunke "Automatic Question Paper Generator System", International Journal of Scientific Research Engineering & Technology (IJSRET), Volume 6, Issue 4, April 2017.