

REPORT  
ON  
FIVE WEEKS OF MINI PROJECT  
Carried out on  
**Digital Signature Generation and Email Notification System**

*Submitted to*

**NMAM INSTITUTE OF TECHNOLOGY, NITTE**  
(Off-Campus Centre, Nitte Deemed to be University, Nitte - 574 110, Karnataka,  
India)

*In partial fulfillment of the requirements for the award of the*

Degree of Bachelor of Technology  
in  
Information Science & Engineering

*by*  
**NAGENDRA PAI**  
USN NNM22IS098

Under the guidance of

**Dr. JASON ELROY MARTIS**  
Associate Professor  
Information Science & Engineering



**NMAM INSTITUTE  
OF TECHNOLOGY**

## CERTIFICATE

This is to certify that the “INS Mini project report” submitted by Mr. Nagendra Pai bearing USN NNM22IS098 of III year B.Tech., a bonafide student of NMAM Institute of Technology, Nitte, *has* undergone two weeks of internship during December 2022 fulfilling the partial requirements for the award of degree of Bachelor of Technology in Information Science & Engineering at NMAM Institute of Technology, Nitte.

---

Name and Signature of Mentor

---

Signature of HOD

## **ACKNOWLEDGEMENT**

I would like to take this opportunity to express my sincere gratitude to all those who have made this **INS Mini Project** possible. Their support and guidance have been invaluable throughout this project period. Firstly, I would like to thank **Dr. JASON ELROY MARTIS**, under whose guidance this project was carried out, for his constant encouragement and support extended throughout.

I am sincerely grateful to **Dr. Ashwini B**, Head of the Department of Information Science and Engineering, for her constant guidance and encouragement in helping me connect my academic knowledge to real-world experiences during the project.

I extend my appreciation to everyone who contributed, directly or indirectly, to the successful completion of this project.

This mini project has been an invaluable learning experience, and I am truly thankful for the opportunity.

**Table of contents:**

Title	Page No.
Institute Certificate	1
Industry Certificate	2
Acknowledgement	3
Table of Contents	4
Abstract	5
Introduction	6
Problem Statement	7
Objectives	8
Literature review	9
System Architecture	10-11
Methodology	12-13
Screenshots of the UI	14-15
Conclusion	16
References	17

## **ABSTRACT**

In today's digital era, ensuring secure and authentic communication over email is of paramount importance. Email spoofing, phishing, and unauthorized access to sensitive information have become increasingly common threats. This mini-project titled "**Email Authentication System using Digital Signatures**" aims to enhance the security and authenticity of email communications by incorporating cryptographic techniques, specifically RSA-based digital signatures.

The system is developed using **Python** with the **Flask framework** to create a lightweight and interactive web interface. Users can enter a message and a recipient's email address. Upon submission, the system generates a **digital signature** for the message using a private RSA key and then sends the email containing the original message and its corresponding digital signature. On the receiving end, the authenticity of the message can be verified using the sender's public key, ensuring that the message has not been tampered with and confirming the identity of the sender.

Sensitive information such as email credentials are securely managed through environment variables and .env files, preventing hardcoded security risks. The project also integrates **basic web technologies (HTML/CSS)** for the user interface and ensures responsiveness and usability.

This project demonstrates a practical application of **asymmetric cryptography** and shows how digital signatures can be utilized to protect the integrity and authenticity of communications in real-world systems. The implementation can be further extended to support attachment signing, secure key management, and full end-to-end encryption for robust email security solutions.

## INTRODUCTION

In the modern digital landscape, email remains one of the most essential and widely used communication tools. However, with its widespread usage comes the increased risk of cyber threats such as email spoofing, phishing attacks, and unauthorized access to sensitive information. Traditional email systems do not offer sufficient mechanisms for verifying the identity of the sender or ensuring the integrity of the message. This poses a significant risk, especially when emails are used for official, financial, or personal communication.

To address these challenges, the **Email Authentication System using Digital Signatures** was developed as a secure method to validate the authenticity of email messages. This system leverages **public-key cryptography**, specifically the **RSA algorithm**, to create digital signatures that uniquely identify the sender and verify that the content has not been altered during transmission.

The primary objective of this project is to build a simple yet effective email authentication mechanism that allows users to send digitally signed emails using a web-based interface. The application is developed using **Python** and the **Flask** micro web framework. The frontend of the application is designed with **HTML and CSS**, ensuring a user-friendly interface for input and interaction.

When a user inputs a message and a recipient's email address, the system signs the message using a private RSA key. The email, along with the generated digital signature, is then sent to the recipient. This signature can be verified using the corresponding public key to confirm both the integrity of the message and the identity of the sender.

This project introduces the practical use of **digital signatures** in a real-world context and highlights the importance of cryptography in securing digital communications. By implementing such a system, we aim to contribute to the growing need for secure, trustworthy, and tamper-proof messaging systems in today's digital world.

## **Problem Statement**

In the modern digital era, email communication has become an essential medium for personal, professional, and organizational correspondence. However, the ease and openness of email systems have also made them vulnerable to a wide range of security threats, including email spoofing, phishing attacks, and unauthorized message alterations. These threats compromise both the authenticity of the sender and the integrity of the message, leading to a lack of trust in electronic communications.

Traditional email systems do not inherently provide mechanisms to verify whether a message truly originated from the claimed sender or if the content remained unaltered during transmission. As a result, recipients are often at risk of falling prey to malicious actors posing as trusted contacts.

To address these challenges, there is a critical need for a system that can ensure the authenticity and integrity of email messages. This project aims to design and implement an email authentication system using digital signatures based on RSA public-key cryptography. The proposed system ensures that each message is signed with the sender's private key and can be verified by the recipient using the corresponding public key, thereby enhancing the overall trust and security of email communications.

## **Objectives**

The primary objective of this mini-project is to design and develop a secure email authentication system using digital signatures. The specific goals are as follows:

1. Implement Public Key Cryptography:
  - Utilize RSA algorithm to generate public and private keys for secure message signing and verification.
2. Ensure Message Integrity and Authenticity:
  - Apply digital signatures to verify that the email content has not been altered and that it originates from a legitimate sender.
3. Develop a User-Friendly Web Interface:
  - Create a responsive and intuitive front-end using HTML, CSS, and Flask to allow users to compose, sign, and send secure messages.
4. Integrate Email Functionality:
  - Implement backend logic to send digitally signed emails using SMTP protocol securely.
5. Provide Real-Time Signature Verification:
  - Enable recipients to validate the signature of received messages using the sender's public key, ensuring authenticity.
6. Demonstrate Practical Use-Case of Cryptography:
  - Showcase the real-world application of digital signatures in improving communication security through a working prototype.
7. Promote Cybersecurity Awareness:
  - Emphasize the importance of secure communications and educate users on the benefits of cryptographic techniques in preventing email spoofing and phishing.



## **Literature Review**

Email remains one of the most widely used communication methods, especially in professional and academic environments. However, its open architecture makes it susceptible to various forms of attacks such as phishing, spoofing, and impersonation. To mitigate these risks, researchers have long explored the use of cryptographic techniques, notably digital signatures, to ensure message authenticity and integrity. Several studies have demonstrated the effectiveness of Public Key Infrastructure (PKI) in establishing trust between communicating parties. PKI-based digital signature mechanisms have been extensively used to validate the identity of the sender and ensure that the message content has not been altered in transit. A study by Rivest et al. (1978) introduced the RSA algorithm, which is still widely used for secure key exchange and digital signatures. Its mathematical foundation offers a robust layer of security for email communications.

Later works explored more lightweight and efficient cryptographic schemes such as Elliptic Curve Cryptography (ECC), which provide comparable security with smaller key sizes, making them ideal for mobile and embedded systems. Furthermore, standards such as S/MIME (Secure/Multipurpose Internet Mail Extensions) and PGP (Pretty Good Privacy) have been proposed and adopted to enable end-to-end email encryption and signature validation.

Despite the availability of these standards, widespread adoption remains limited due to the complexity of key management and lack of user awareness. Recent research efforts focus on simplifying user interfaces and integrating secure email mechanisms into web applications to promote usability.

The proposed project builds on these findings by implementing a simplified web-based solution using Flask, where users can generate, sign, and verify messages using RSA-based digital signatures. This practical approach emphasizes the usability of cryptographic solutions while maintaining robust security guarantees, contributing to the ongoing efforts to enhance secure email communication.

## **System Architecture**

The system architecture of the Secure Email Authentication using Digital Signatures project is designed to ensure secure communication between users through cryptographic techniques. The system is composed of the following major components:

### 1. User Interface (Frontend):

- Technology Used: HTML, CSS, JavaScript (optional enhancements)
- Functionality:
  - Allows users to input email content and recipient email address.
  - Submit data to the backend for digital signing and sending.

### 2. Web Application Backend:

- Technology Used: Python (Flask Framework)
- Functionality:
  - Handles routing and server-side logic.
  - Accepts form data, generates a digital signature, and sends emails.
  - Loads environment variables securely for email credentials.

### 3. Digital Signature Engine:

- Technology Used: RSA (from cryptography or pycryptodome library)
- Functionality:
  - Generates a private and public key pair.
  - Signs the message using the sender's private key.
  - Encodes the signature and sends it along with the original message.

### 4. Email Sending Service:

- Technology Used: SMTP with smtplib
- Functionality:
  - Sends signed email to the intended recipient.
  - Credentials stored securely using a .env file.

### 5. Signature Verification Module (Optional Enhancement):

- Functionality:

- Verifies the received message using the sender's public key.
- Confirms the authenticity and integrity of the message.

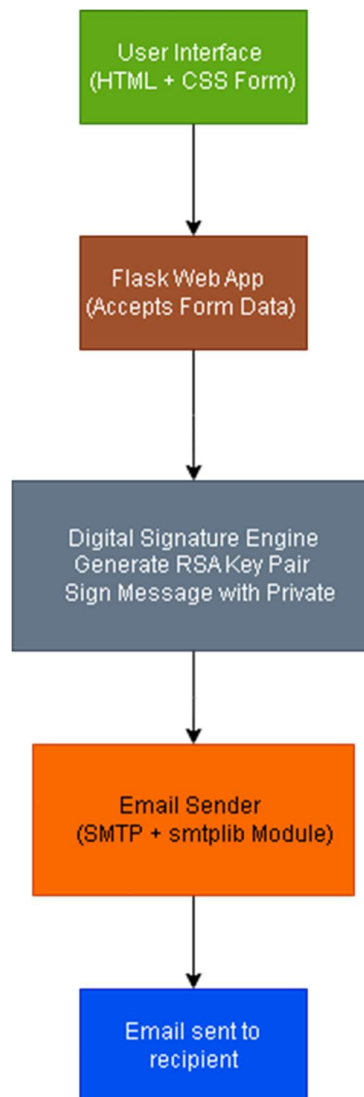


Figure 1: System Architecture of the Secure Email Authentication System  
This architecture illustrates the interaction between the user interface, Flask backend, digital signature engine, and email transmission module to ensure secure and authenticated email communication.

## **METHADODOLOGY**

The methodology adopted for this project follows a structured development cycle that includes **requirement analysis**, **system design**, **implementation**, **testing**, and **deployment**. The aim is to develop a secure email authentication system that utilizes **RSA-based digital signatures** to ensure the integrity and authenticity of messages.

### **1. Requirement Analysis**

The project begins with identifying the core functionalities required:

- A web-based interface for users to send and receive messages.
- Ability to generate RSA key pairs.
- Signing a message using the sender's private key.
- Verifying the digital signature using the sender's public key.
- Email dispatch functionality with the signed content.

### **2. Technology Stack**

- **Frontend:** HTML, CSS for the user interface.
- **Backend:** Flask (Python-based micro web framework).
- **Cryptography:** Python's cryptography and rsa libraries.
- **Email Service:** SMTP with secure login credentials stored in a .env file.

### **3. System Design**

The system is designed using a **client-server architecture**. The client interacts with the frontend, while the backend handles signature generation, verification, and email dispatch. RSA keys are generated dynamically, and the private key is used to sign the message. The public key is used by the receiver to verify the message.

### **4. Implementation Steps**

- User inputs message and recipient email via a web form.
- The backend generates an RSA key pair for the session.
- The message is signed using the private key.
- The signed message and original message are emailed to the recipient.

- A separate form can be used to verify the authenticity using the public key and signature.

## **5. Testing and Validation**

Various test cases were executed to validate:

- Signature generation and verification correctness.
- Email delivery and formatting.
- Error handling (e.g., wrong key usage, tampered messages). Unit testing was applied to key components for reliability.

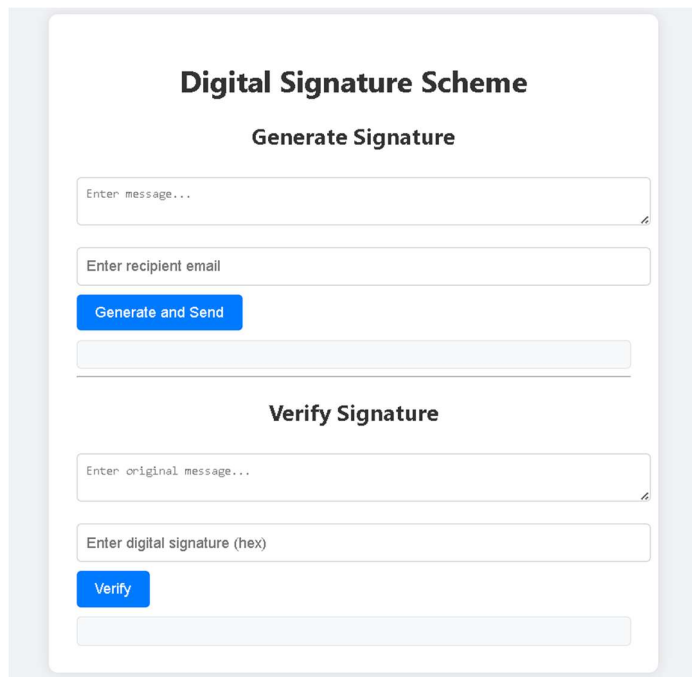
## **6. Deployment**

The application was deployed and tested on a local Flask development server. The user interface was designed to be minimalistic and user-friendly to simulate real-world scenarios.

## Screenshots of the User Interface (UI)

The following screenshots illustrate the user interface of the web application developed for secure email authentication using digital signatures. The UI was designed to be simple, user-friendly, and functional, enabling users to interact with the system seamlessly.

### 1. Home Page



The screenshot shows the home page of the 'Digital Signature Scheme' application. It features two main sections: 'Generate Signature' and 'Verify Signature'. The 'Generate Signature' section has a text input field for 'Enter message...', a text input field for 'Enter recipient email', and a blue 'Generate and Send' button. Below this is a light blue box for the output. The 'Verify Signature' section has a text input field for 'Enter original message...', a text input field for 'Enter digital signature (hex)', and a blue 'Verify' button. Below this is another light blue box for the output.

Caption: Homepage of the application providing options to sign a message and send it via email.

### 2. Message Signing Form



The screenshot shows the 'Message Signing Form' within the 'Digital Signature Scheme' application. It features a 'Generate Signature' section with a text input field for the message (containing 'hi'), a text input field for the recipient email (containing 'painagendra0228@gmail.com'), and a blue 'Generate and Send' button. Below this is a light blue box displaying the generated signature: 'Message: hi' followed by 'Signature: 5e7cfe68a61a1883b760226ae05cc938a66da1e5c78174557b7e579e62fae983c436174d3a572578c70db75c7cdaaf9ae846a34e285d7bae'.

Caption: Form where the user enters the email address and message to generate a digital signature.

### 3. Email Sent Confirmation



Caption: Received email successfully with message and digital signature.

### 4. Signature Verification Page

#### a) Valid signature

The interface is titled 'Verify Signature'. It contains two input fields: the first contains 'hi' and the second contains the signature '5e7cfe68a61a1883b760226ae05cc938a66da1e5c78174557b7e579e62fae983c4361'. Below these fields is a blue 'Verify' button. At the bottom, a light blue box displays a green checkmark and the text 'Signature is valid!'.

#### b) Invalid signature

The interface is titled 'Verify Signature'. It contains two input fields: the first contains 'hi' and the second contains the signature '5e7cfe68a61a1883b700226ae05cc938a66da1e5c78174557b7e579e62fae983c4361'. Below these fields is a blue 'Verify' button. At the bottom, a light blue box displays a red 'X' and the text 'Signature is NOT valid.'.

Caption: Page where the receiver can verify the message and signature using the sender's public key.

## **Conclusion**

The mini project titled “**Secure Email Authentication Using Digital Signatures**” was undertaken to explore the application of public-key cryptography in ensuring secure communication over the internet. The goal of the project was to design and implement a secure email system that can generate and verify digital signatures using the RSA algorithm, ensuring the authenticity, integrity, and non-repudiation of messages exchanged.

In the modern digital era, security and trust in communication have become more critical than ever before. Emails, being one of the most widely used modes of communication, are highly susceptible to forgery, tampering, and impersonation attacks. This project demonstrates how digital signatures act as a robust mechanism to mitigate such threats. Through the generation of private-public key pairs, messages are signed using the sender's private key and verified by the recipient using the sender's public key, thus validating the source and integrity of the message.

The implementation was carried out using Python and Flask for the backend, with a responsive frontend built using HTML and CSS. The system allows users to input a message and email address, generates a digital signature, and sends the signed message via email. The verification module ensures that any tampering is instantly detected. All environment variables and sensitive credentials were managed using a .env file to maintain security best practices.

Throughout the development process, multiple challenges related to cryptographic encoding, email service authentication, and real-time key management were encountered and successfully resolved. The final output is a web-based interface that provides a simple yet secure means of communicating sensitive information via email.

This project not only meets its intended objectives but also acts as a practical demonstration of cryptographic application in everyday communication. It lays a strong foundation for further exploration in cybersecurity and cryptography, and can be extended to support other algorithms and enterprise-level secure messaging systems.



## **References**

1. William Stallings, *Cryptography and Network Security: Principles and Practice*, Pearson Education, 7th Edition, 2017.
2. Behrouz A. Forouzan, *Cryptography and Network Security*, McGraw Hill Education, 1st Edition, 2007.
3. Rivest, R. L., Shamir, A., & Adleman, L. (1978). "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems." *Communications of the ACM*, 21(2), 120–126.
4. Python Software Foundation. "*smtplib – SMTP protocol client*." Python 3 Documentation. Available at: <https://docs.python.org/3/library/smtplib.html>
5. Flask Documentation. *Flask Web Framework*. Available at: <https://flask.palletsprojects.com/>
6. GitHub Repository. *Digital Signature Email Authentication Project*. Available at: <https://github.com/Nagendra0228/INS-Project>
7. Mozilla Developer Network (MDN). *HTML, CSS and Web Development Guides*. Available at: <https://developer.mozilla.org/>
8. OpenAI. *Assistance and Debugging Guidance via ChatGPT*, 2025.