A

Mini Project

On

SECURE CRYPTO-BIOMETRIC SYSTEM FOR CLOUD COMPUTING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

SARA	(207R1A05N8)
ANUMULA ABHITEJ REDDY	(207R1A05J7)

Under the Guidance of

Dr. V. NARESH KUMAR

(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"SECURE CRYPTO-BIOMETRIC SYSTEMFOR CLOUD COMPUTING"** being submitted by **SARA (207R1A05N8) & ANUMULA ABHITEJ REDDY (207R1A05J7)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-2024.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

DR.V. NARESH KUMAR
(Associate Professor)
INTERNAL GUIDE

DR. A. RAJI REDDY
DIRECTOR

DR. K. SRUJAN RAJU
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on_____

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Dr. V. Naresh Kumar,** Associate Professor for his exemplary guidance, monitoring, and constantencouragement throughout the project work. The blessing, help, and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G.Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughoutthe course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

SARA (207R1A05N8)

ANUMULA ABHITEJ REDDY (207R1A05J7)

ABSTRACT

Cloud computing has achieved maturity, and there is a heterogeneous group of providers and cloud-based services. However, significant attention remains focused on security concerns. In many cases, security and privacy issues are a significant barrier to user acceptance of cloud computing systems and the advantages these offer with respect to previous systems. Biometric technologies are becoming the key aspect of a wide range of secure identification and personal verification solutions, but in a cloud computing environment they present some problems related to the management of biometric data, due to privacy regulations and the need to trust cloud providers. To overcome those problems in this project, we propose a crypto-biometric system applied to cloud computing in which no private biometric data are exposed.

Cloud computing is a trend in application architecture and development, as well as a new business model. The success of many service providers, with Amazon as a remarkable example, has demonstrated that the model can be applied to a wide variety of solutions, covering the different levels defined in the cloud paradigm (SaaS, PaaS and IaaS). We can consider that cloud computing is at a mature stage, although there remain some limitations and challenges. Cloud computing brings important benefits for organizations that outsource data, applications, and infrastructure, at the cost of delegating data control. The information is processed in computers that the users do not own, operate, or manage. In this scenario, the user does not know how the provider handles the information, and therefore a high level of trust is needed. The lack of control over physical and logical aspects of the system imposes profound changes in security and privacy procedures.

LIST OF FIGURES/TABLES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 3.1	Project Architecture of Secure Crypto-Biometric System for Cloud Computing	7
Figure 3.2	Use Case Diagram of Secure Crypto-Biometric System for Cloud Computing	8
Figure 3.3	Class Diagram of Secure Crypto-Biometric System for Cloud Computing	9
Figure 3.4	Sequence Diagram of Secure Crypto-Biometric System for Cloud Computing	10
Figure 3.5	Activity Diagram of Secure Crypto-Biometric System for Cloud Computing	11

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
Screenshot 5.1	Upload Biometric Database	19
Screenshot 5.2	Upload Fingerprint Biometric Images	19
Screenshot 5.3	Run Features Extraction	20
Screenshot 5.4	Run Features Selection and BCH Encoder	20
Screenshot 5.5	AES, ECC Encoder Training using GMM & Key	21
Screenshot 5.6	BCH Decoder Verification	21
Screenshot 5.7	Upload Finger Template	22
Screenshot 5.8	AES and ECC Encryption Time Graph	22
Screenshot 5.9	Observer Execution Time	23

TABLE OF CONTENTS

ABSTRACT i

LIST OF FIGURES ii

LIST OF SCREENSHOTS iii

1. INTRODUCTION 1

1.1 PROJECT SCOPE 1

1.2 PROJECT PURPOSE 1

1.3 PROJECT FEATURES 1

2. SYSTEM ANALYSIS 2

2.1 PROBLEM DEFINITION 2

2.2 EXISTING SYSTEM 2

2.2.1 LIMITATIONS OF THE EXISTING SYSTEM 3

2.3 PROPOSED SYSTEM 3

2.3.1 ADVANTAGES OF PROPOSED SYSTEM 3

2.4 FEASIBILITY STUDY 4

2.4.1 ECONOMIC FEASIBILITY 4

2.4.2 TECHNICAL FEASIBILITY 4

2.4.3 SOCIAL FEASIBILITY 5

2.5 HARDWARE & SOFTWARE REQUIREMENTS 5

2.5.1 HARDWARE REQUIREMENTS 5

2.5.2 SOFTWARE REQUIREMENTS 6

3. ARCHITECTURE 7

3.1 PROJECT ARCHITECTURE 7

3.2 DESCRIPTION 7

3.3 USE CASE DIAGRAM 8

3.4 CLASS DIAGRAM 9

3.5 SEQUENCE DIAGRAM 10
3.6 ACTIVITY DIAGRAM 11
4. IMPLEMENTATION 12
4.1 SAMPLE CODE 12
5. SCREENSHOTS 19
6. TESTING 24
6.1 INTRODUCTION TO TESTING 24
6.2 TYPES OF TESTING 24
6.2.1 UNIT TESTING 24
6.2.2 INTEGRATION TESTING 25
6.2.3 FUNCTIONAL TESTING 25
6.2.4 SYSTEM TEST 26
6.2.5 WHITE BOX TESTING 26
6.2.6 BLACK BOX TESTING 27
6.3 TEST CASES 28
6.3.1 CLASSIFICATION 28
7. CONCLUSION & FUTURE SCOPE 29
7.1 PROJECT CONCLUSION 29
7.2 FUTURE SCOPE 29
8. BIBLIOGRAPHY 30
8.1 REFERENCES 30
8.2 GITHUB LINK 31

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

The project's scope encompasses a wide range of critical aspects related to the development and deployment of the Secure Crypto-Biometric System for Cloud Computing. It begins with the definition of the system's architecture, outlining the key components responsible for biometric data capture, cryptographic operations, user management, and seamless cloud integration.

1.2 PROJECT PURPOSE

The primary purpose of this project is to create a Secure Crypto-Biometric Systemtailored for cloud computing environments. This system's overarching goal is to significantly enhance the security and privacy of cloud-based services and data by combining two potent elements: biometric authentication and cryptographic techniques. By doing so, it aims to address the pressing need for robust cloud security solutions.

1.3 PROJECT FEATURES

The Secure Crypto-Biometric System for Cloud Computing will incorporate several distinctive features to fulfill its objectives effectively. One of the core features is Biometric Authentication, which enables users to verify their identity using biometric identifiers. This not only enhances security but also streamlines the user experience, reducing reliance on passwords and tokens. Data Encryption is another central feature. The system will encrypt all data stored in the cloud using robust cryptographic algorithms. This ensures that data remains confidential and inaccessible to unauthorized parties. Multi-factor authentication will also be supported.

2. SYSTEM ANALYSIS

2.SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, "what must be done to solve the problem?" The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

The primary challenge addressed by the Secure Crypto-Biometric System for Cloud Computing is the existing security vulnerabilities within the cloud environment. This includes weaknesses in authentication methods, potential data breaches, and the secure storage of biometric data. The system aims to define and resolve these vulnerabilities, enhance security, and ensure user-friendly authentication while safeguarding biometric data.

2.2 EXISTING SYSTEM

When the data are stored in the user infrastructure, information location and protection mechanisms are known in detail. In contrast, a characteristic of public cloud computing services is that the user is completely unaware of data location. This makes it impossible to ensure that national compulsory regulations are met. Several techniques have been proposed for biometric template protection. Among them, cancelable biometrics is one of the most promising.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

- Vulnerable to password-related security threats.
- Limited effectiveness of traditional username/password authentication.
- Inadequate multi-factor authentication (MFA) adoption due to complexity.
- Insufficient data encryption in transit and at rest.

2.3 PROPOSED SYSTEM

In the proposed schema, once a large database with sample acquisitions has been collected, an UBM can be trained. We propose the training methodology in Figure 3 (the addition of new UBMs) to improve the flexibility and security of our system. To train a new UBM, computing resources are provided by virtual machines hosted in Amazon EC2. The administration application requests from Amazon EC2 the required virtual machines automatically, using the API it provides. The training application and the new UBM are loaded and executed in the machines in a distributed way to reduce computation time. Speedup is possible due to the high parallelizability of the calculus performed on biometric data.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Enhanced Security through biometric authentication and strong cryptography.
- Improved User Convenience with seamless biometric login methods.
- Data Privacy ensured through secure storage of biometric data.
- Compliance Alignment with regulatory requirements and industry standards.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are:

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited**.** The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements:

- **System** : Intel I3 or Above.
- **Hard Disk** : 40 GB.
- **RAM** : 4GB

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements:

- **Operating System** : Windows 8 or Above.
- **Coding Language** : Python 3.7
- **Platform** : Python Technology
- **Tool** : Spyder
- **Front** : Anaconda
- **Back End** : Python Anaconda Script

3.ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

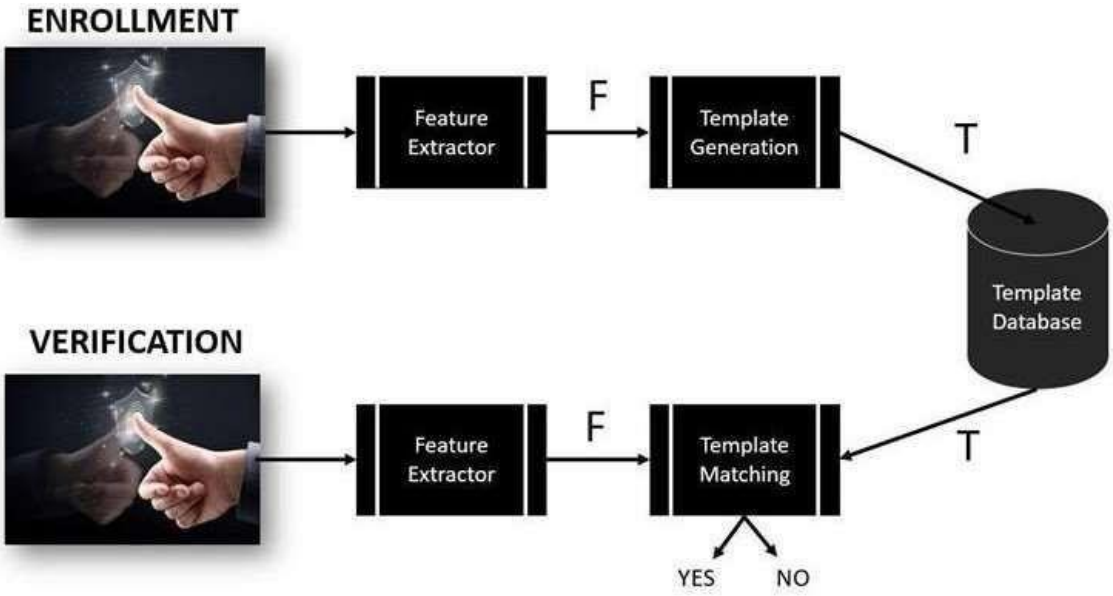


Figure 3.1: Project Architecture of the proposed schema for building an encrypted data repository in the cloud that is protected with biometrics

3.2 DESCRIPTION

During enrollment a number of biometric measurements are recorded for each user. The acquired biometrics are then individually processed, in order to extract the information they contain and calculate fuzzy commitment helper data, which will later be used when the user tries to access the system, as explained in the previous section. In this process, a public/secret key pair is generated using biometric information and the public key is stored in a key repository, for third parties to secure communications with the user. When the user tries to access the system, his biometric information is acquired, and the auxiliary data (UBM, helper, and fuzzy commitment data) that are necessary to process information are retrieved from S3 storage.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model.

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

Figure 3.2: Use Case Diagram for Secure Crypto-Biometric Systemfor cloud computing

3.4 CLASS DIAGRAM

Class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

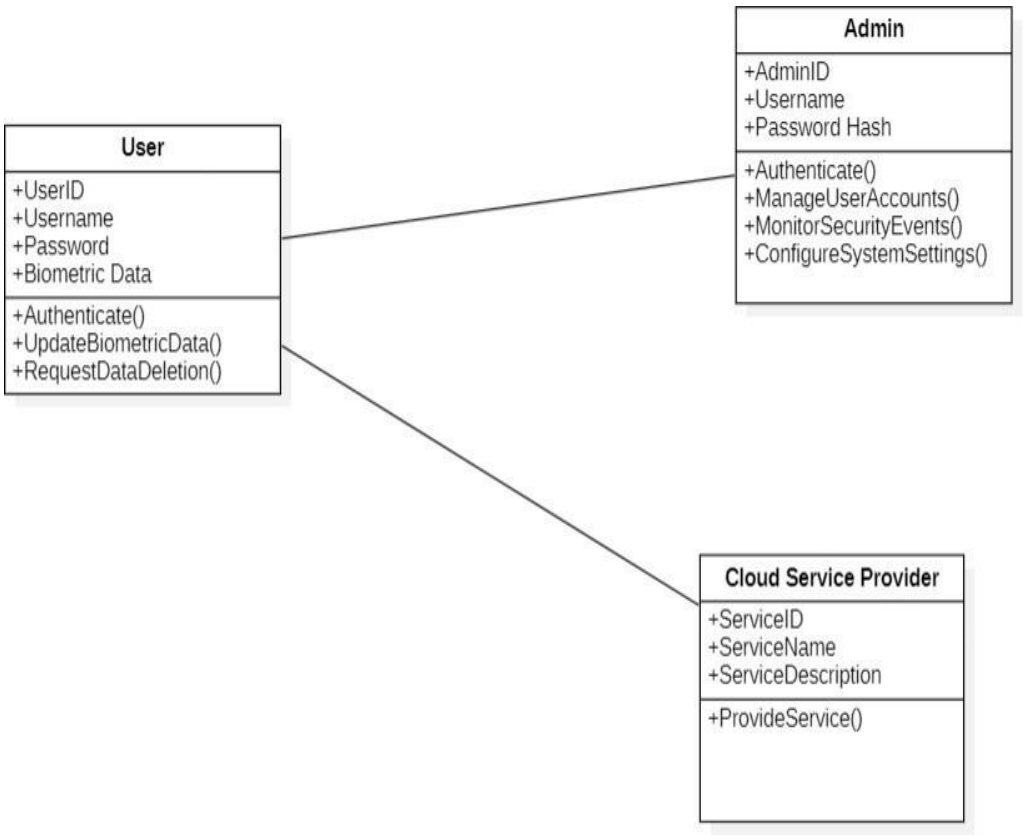


Figure 3.3: Class Diagram for Secure Crypto-biometric System for cloud computing

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

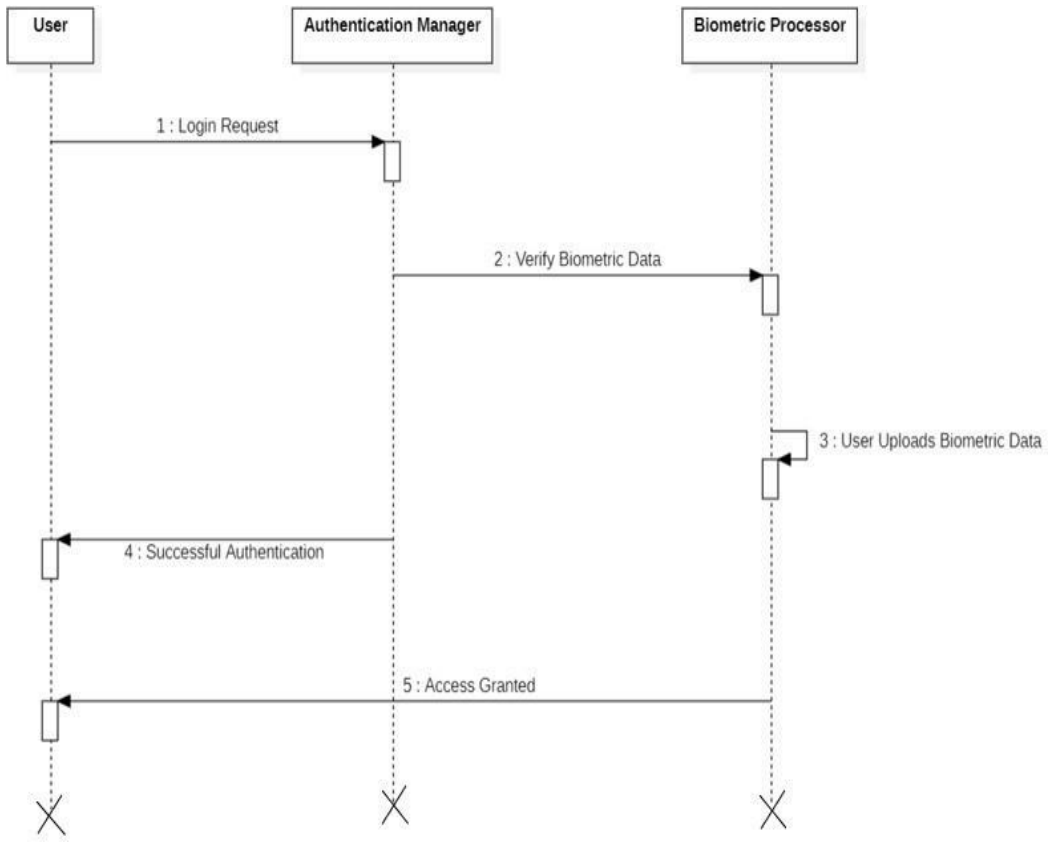


Figure 3.4: Sequence Diagram for Secure Crypto-Biometric System for cloud computing

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

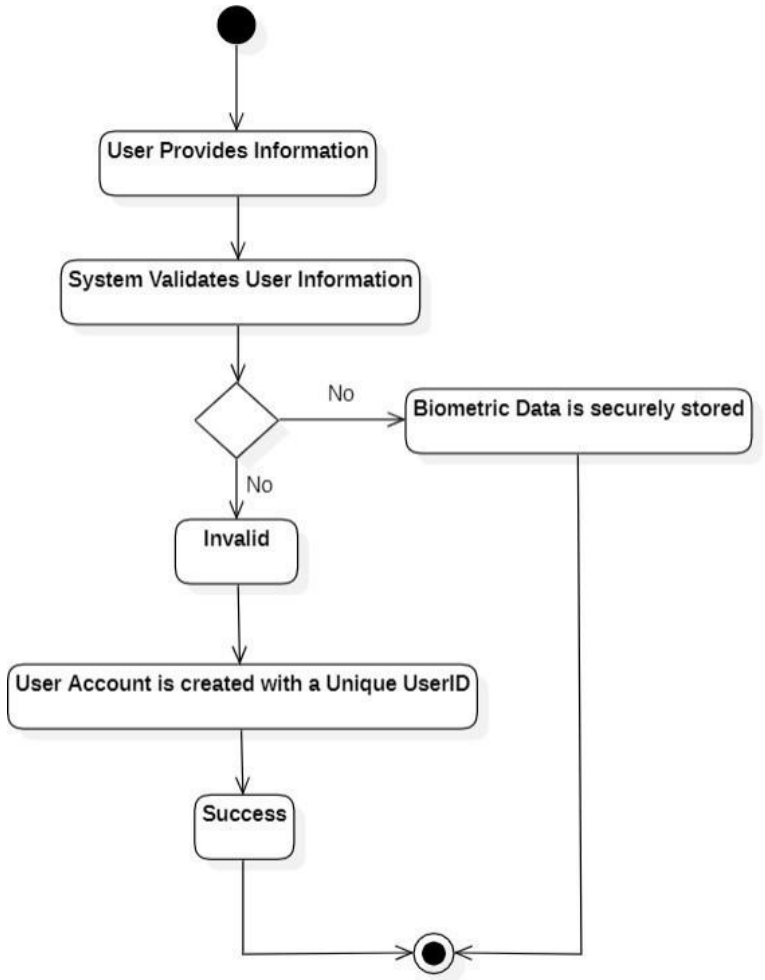


Figure 3.5: Activity Diagram for Secure Crypto Biometric System for cloud computing

4. IMPLEMENTATION

4.1 SAMPLE CODE

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
from tkinter.filedialog import askopenfilename
import os
import numpy as np
import cv2
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.mixture import GaussianMixture
from sklearn.metrics import accuracy_score
import pickle
from ecies.utils import generate_eth_key, generate_key
from ecies import encrypt, decrypt #importing classes for ECC encryption
import pyaes, pbkdf2, binascii, os, secrets #importing classes for AES as PYAES
import time

main = tkinter.Tk()
main.title("Secure crypto-biometric system for cloud computing")
main.geometry("1300x1200")

global filename, pathlabel
global X, Y, encoder, pca, gmm
global labels
global ecc_publicKey,ecc_privateKey #defining public and private keys variables for
ECC
global aes_time, ecc_time

def ECCEncrypt(obj): #ECC encryption function
    enc = encrypt(ecc_publicKey, obj)
    return enc

def ECCDecrypt(obj): #ECC decryption function
     dec = decrypt(ecc_privateKey, obj)
     return dec
```

```
def generateKey(): #function to generate ECC keys
    global ecc_publicKey,ecc_privateKey
    eth_k = generate_eth_key()
    ecc_private_key = eth_k.to_hex()
    ecc_public_key = eth_k.public_key.to_hex()
    return ecc_private_key, ecc_public_key

def getAesKey(): #generating key with PBKDF2 for AES
    password = "s3cr3t*c0d3"
    passwordSalt = '76895'
    key = pbkdf2.PBKDF2(password, passwordSalt).read(32)
    return key

def Aesencrypt(plaintext): #AES data encryption
    aes = pyaes.AESModeOfOperationCTR(getAesKey(),
pyaes.Counter(311295470350000473029524339676541953981242398445663228841721
163637846056248223))
    ciphertext = aes.encrypt(plaintext)
    return ciphertext

def Aesdecrypt(enc): #AES data decryption
    aes = pyaes.AESModeOfOperationCTR(getAesKey(),
pyaes.Counter(311295470350000473029524339676541953981242398445663228841721
163637846056248223))
    decrypted = aes.decrypt(enc)
    return decrypted

def readLabels(path):
    global labels
    for root, dirs, directory in os.walk(path):
        for j in range(len(directory)):
            name = os.path.basename(root)
            if name not in labels:
                labels.append(name)

def getID(name):
    label = 0
    for i in range(len(labels)):
        if name == labels[i]:
            label = i
            break
    return label
```

```
def uploadDatabase():
    global filename, labels
    labels = []
    filename = filedialog.askdirectory(initialdir=".")
    pathlabel.config(text=filename)
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n\n")
    readLabels(filename)
    text.insert(END,"Total persons biometric templates found in Database:
"+str(len(labels))+"\n\n")
    text.insert(END,"Person Details\n\n")
    text.insert(END, str(labels))


def featuresExtraction():
    global filename
    text.delete('1.0', END)
    global X, Y
    if os.path.exists("model/X.npy"):
        X = np.load("model/X.npy")
        Y = np.load("model/Y.npy")
    else:
        X = []
        Y = []
        for root, dirs, directory in os.walk(filename):
            for j in range(len(directory)):
                name = os.path.basename(root)
                if 'Thumbs.db' not in directory[j]:
                    img = cv2.imread(root+"/"+directory[j],0)
                    img = cv2.resize(img, (28,28))
                    label = getID(name)
                    X.append(img.ravel())
                    Y.append(label)
                    print(str(label)+" "+name)
        X = np.asarray(X)
        Y = np.asarray(Y)
        X = X.astype('float32')
        X = X/255
        np.save("model/X", X)
        np.save("model/Y", Y)
    text.insert(END,"Extracted Features from templates\n\n")
    text.insert(END,str(X))
```

```
def featuresSelection():
    text.delete('1.0', END)
    global X, Y, pca, encoder
    text.insert(END,"Total features available in templates before applying PCA features
selection: "+str(X.shape[1])+"\n\n")
    pca = PCA(n_components=60)
    X = pca.fit_transform(X)
    text.insert(END,"Total features available in templates after applying PCA features
selection: "+str(X.shape[1])+"\n\n")
    text.insert(END,"Encoder features after encrypting with KEY\n\n")
    encoder = []
    for i in range(len(X)):
        temp = []
        for j in range(len(X[i])):
            temp.append(X[i,j]**2)
        encoder.append(temp)
    encoder = np.asarray(encoder)
    text.insert(END,str(encoder))


def runGMMEncoding():
    text.delete('1.0', END)
    global ecc_publicKey,ecc_privateKey
    global aes_time, ecc_time
    global encoder, Y, gmm
    if os.path.exists('model/gmm.txt'):
        with open('model/gmm.txt', 'rb') as file:
            gmm = pickle.load(file)
        file.close()
    else:
        gmm = GaussianMixture(n_components=10, max_iter = 1000)
        gmm.fit(encoder, Y)
    #gmm is the object which is used for verification and it contains all templates details
so GMM has to get encrypted
    start = time.time()
    ecc_privateKey, ecc_publicKey = generateKey()#getting ECC keys
    gmm = ECCEncrypt(pickle.dumps(gmm))#now encrypting GMM using ECC
    gmm = pickle.loads(ECCDecrypt(gmm))#now decrypting GMM using ECC
    end = time.time()
    ecc_time = end - start #calculating ECC encryption and decryption time
```

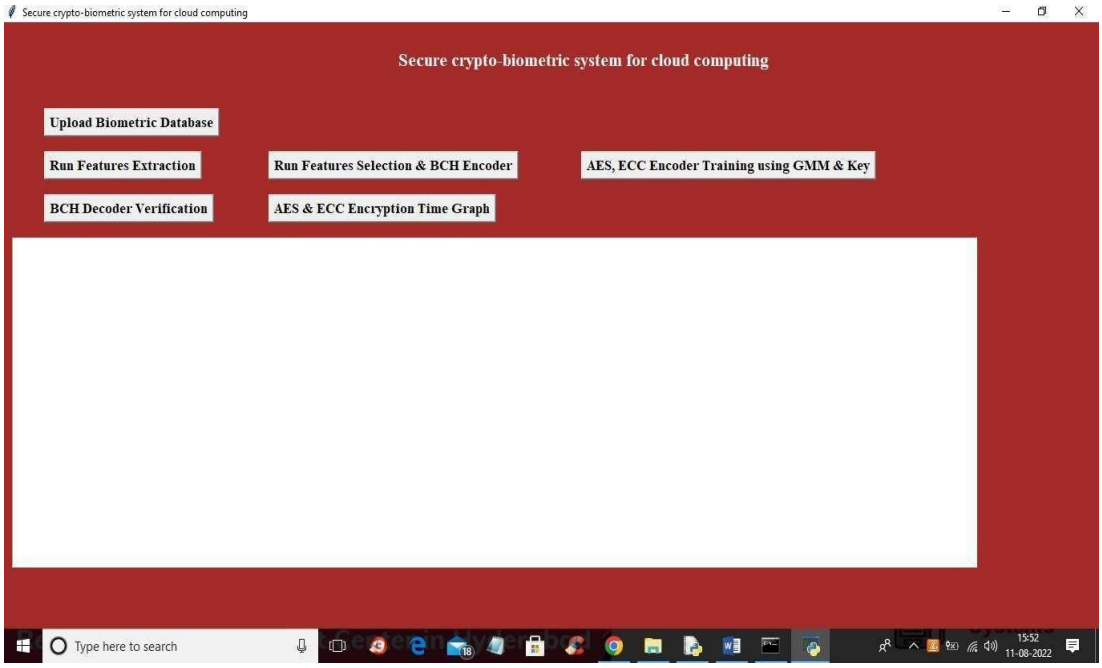
```
#now encrypting with AES
    start = time.time() #getting AES start time
    gmm = Aesencrypt(pickle.dumps(gmm)) #doing AES encryption on GMM
    encrypted_data = gmm[0:400]
    end = time.time()
    aes_time = end - start #calculating AES encryption and decryption time
    gmm = pickle.loads(Aesdecrypt(gmm)) #doing AES decryption on GMM
    ecc_time = ecc_time * 4
    text.insert(END,"Encoder training & AES & ECC Encryption process completed on
GMM\n\n")
    text.insert(END,"Time taken by AES : "+str(aes_time)+"\n\n")
    text.insert(END,"Time taken by ECC : "+str(ecc_time)+"\n\n")
    text.insert(END,"Encrypted Data\n\n")
    text.insert(END,str(encrypted_data))


def verification():
    text.delete('1.0', END)
    global pca, gmm
    filename = filedialog.askopenfilename(initialdir="testImages")
    img = cv2.imread(filename,0)
    img = cv2.resize(img, (28,28))
    test = []
    test.append(img.ravel())
    test = np.asarray(test)
    test = test.astype('float32')
    test = test/255
    test = pca.transform(test)
    decoder = []
    for i in range(len(test)):
        temp = []
        for j in range(len(test[i])):
            temp.append(test[i,j]**2)
        decoder.append(temp)
    decoder = np.asarray(decoder)
    predict = gmm.predict(decoder)[0]
    img = cv2.imread(filename)
    img = cv2.resize(img, (600,400))
    cv2.putText(img, 'Biometric template belongs to person : '+str(predict), (10, 25),
    cv2.FONT_HERSHEY_SIMPLEX,0.7, (255, 0, 0), 2)
```

```
    cv2.imshow('Biometric template belongs to person : '+str(predict), img)
    cv2.waitKey(0)

def graph():
    global aes_time, ecc_time
    height = [aes_time, ecc_time]
    bars = ('AES Execution Time','ECC Execution Time')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos, height)
    plt.xticks(y_pos, bars)
    plt.title("AES & ECC Execution Time Graph")
    plt.show()


def GUI():
    global text, main, pathlabel
    font = ('times', 16, 'bold')
    title = Label(main, text='Secure crypto-biometric system for cloud computing')
    title.config(bg='brown', fg='white')
    title.config(font=font)
    title.config(height=3, width=120)
    title.place(x=0,y=5)

    font1 = ('times', 13, 'bold')
    uploadButton = Button(main, text="Upload Biometric Database",
command=uploadDatabase)
    uploadButton.place(x=50,y=100)
    uploadButton.config(font=font1)

    pathlabel = Label(main)
    pathlabel.config(bg='brown', fg='white')
    pathlabel.config(font=font1)
    pathlabel.place(x=460,y=100)

    extractionButton = Button(main, text="Run Features Extraction",
command=featuresExtraction)
    extractionButton.place(x=50,y=150)
    extractionButton.config(font=font1)
```

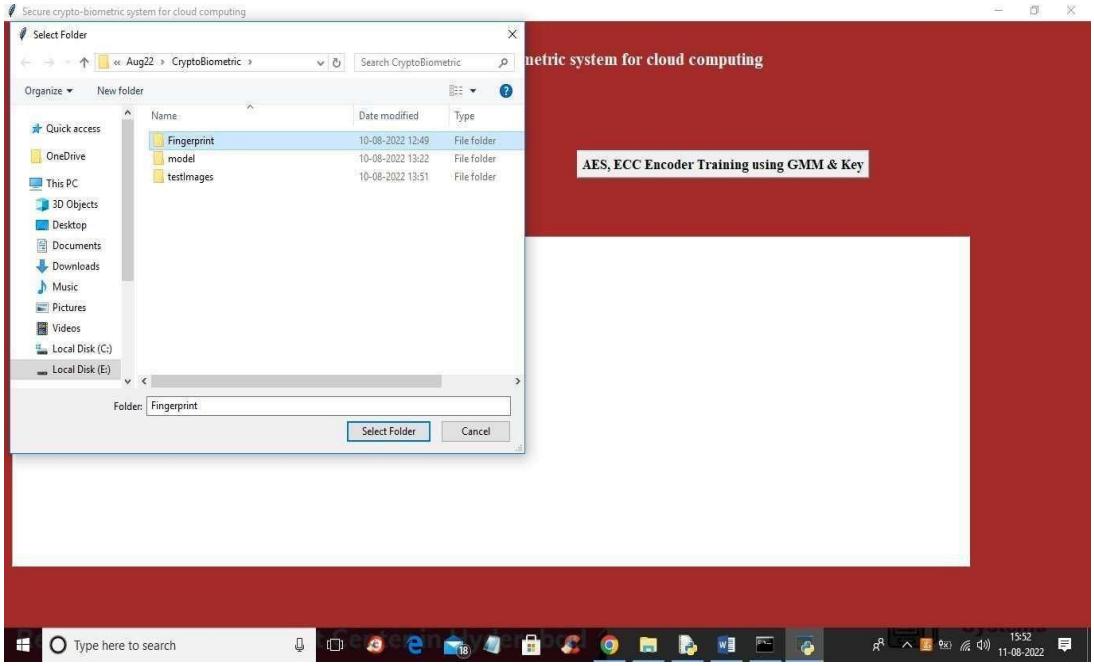
```
selectionButton = Button(main, text="Run Features Selection & BCH Encoder",
command=featuresSelection)
   selectionButton.place(x=330,y=150)
   selectionButton.config(font=font1)



encodingButton = Button(main, text="AES, ECC Encoder Training using GMM
& Key", command=runGMMEncoding)
   encodingButton.place(x=720,y=150)
   encodingButton.config(font=font1)



verificationButton = Button(main, text="BCH Decoder Verification",
command=verification)
   verificationButton.place(x=50,y=200)
   verificationButton.config(font=font1)

   graphButton = Button(main, text="AES & ECC Encryption Time Graph",
command=graph)
   graphButton.place(x=330,y=200)
   graphButton.config(font=font1)


   font1 = ('times', 12, 'bold')
   text=Text(main,height=20,width=150)
   scroll=Scrollbar(text)
   text.configure(yscrollcommand=scroll.set)
   text.place(x=10,y=250)
   text.config(font=font1)

   main.config(bg='brown')
   main.mainloop()

if__name__== "_main_":
   GUI()
```

5. SCREENSHOTS

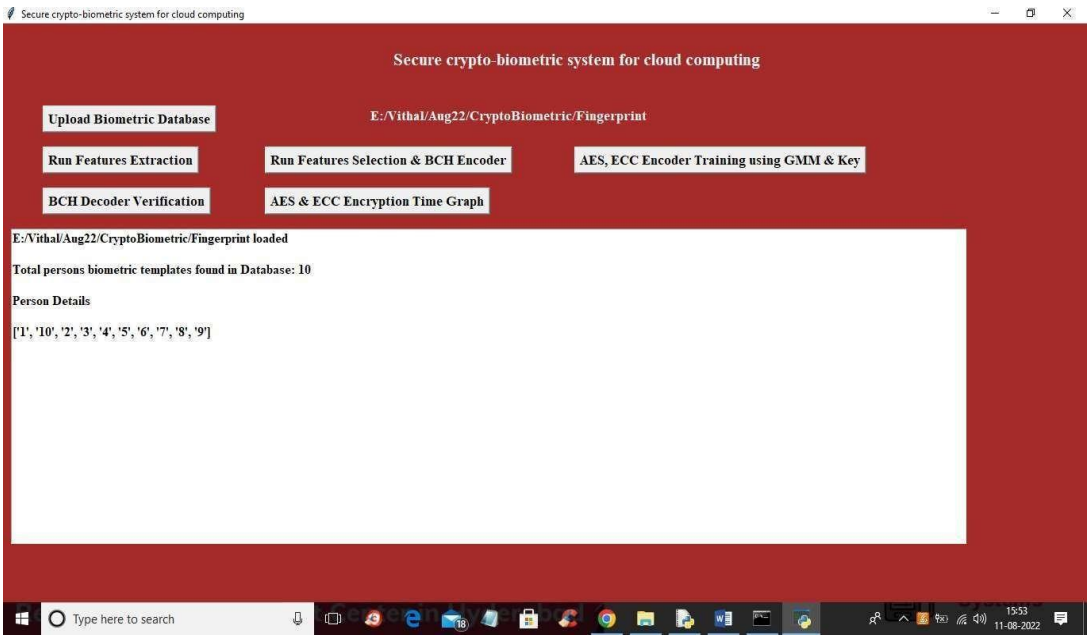
Screenshot 5.1: Upload Biometric Database



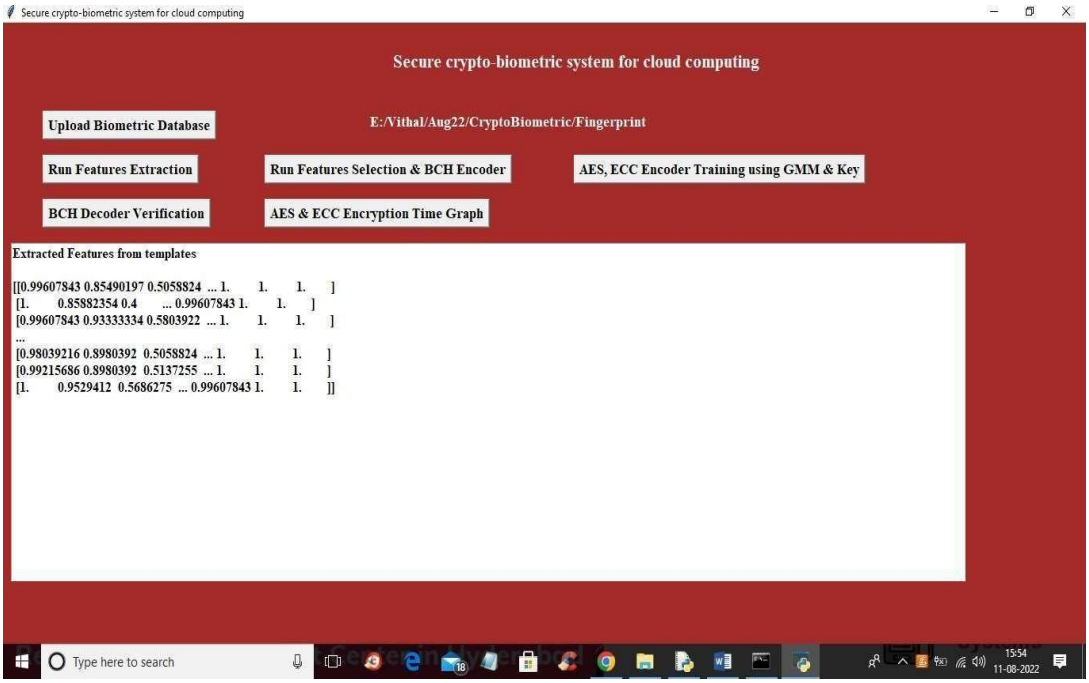
Screenshot 5.2: Upload Finger Biometric images



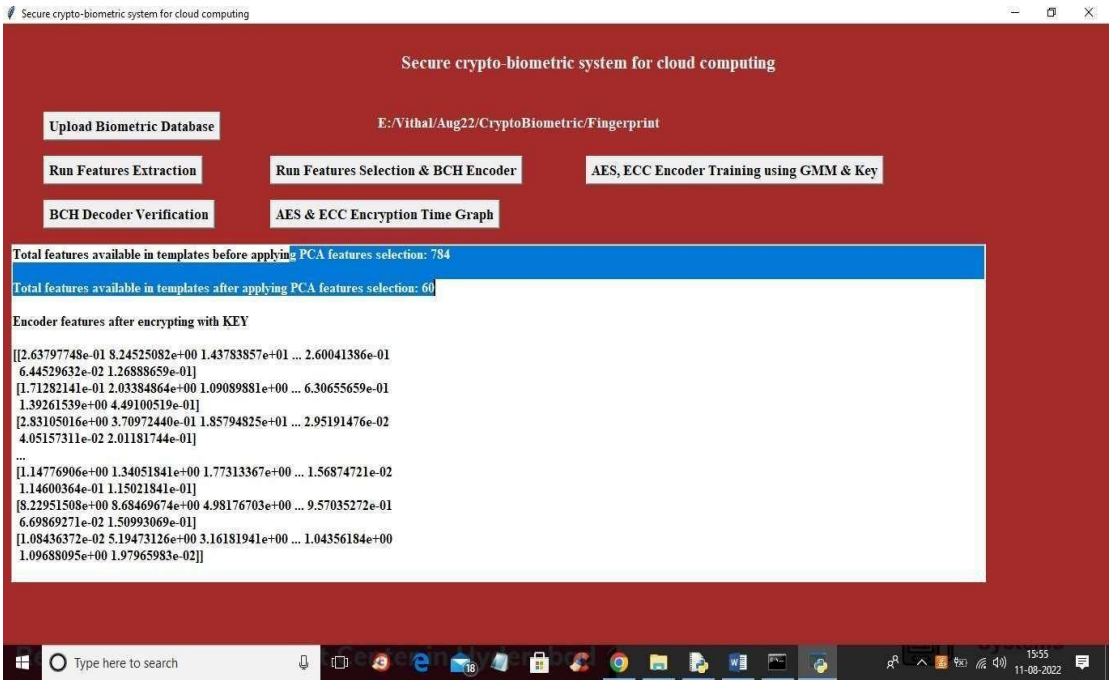
Screenshot 5.3: Run Features Extraction



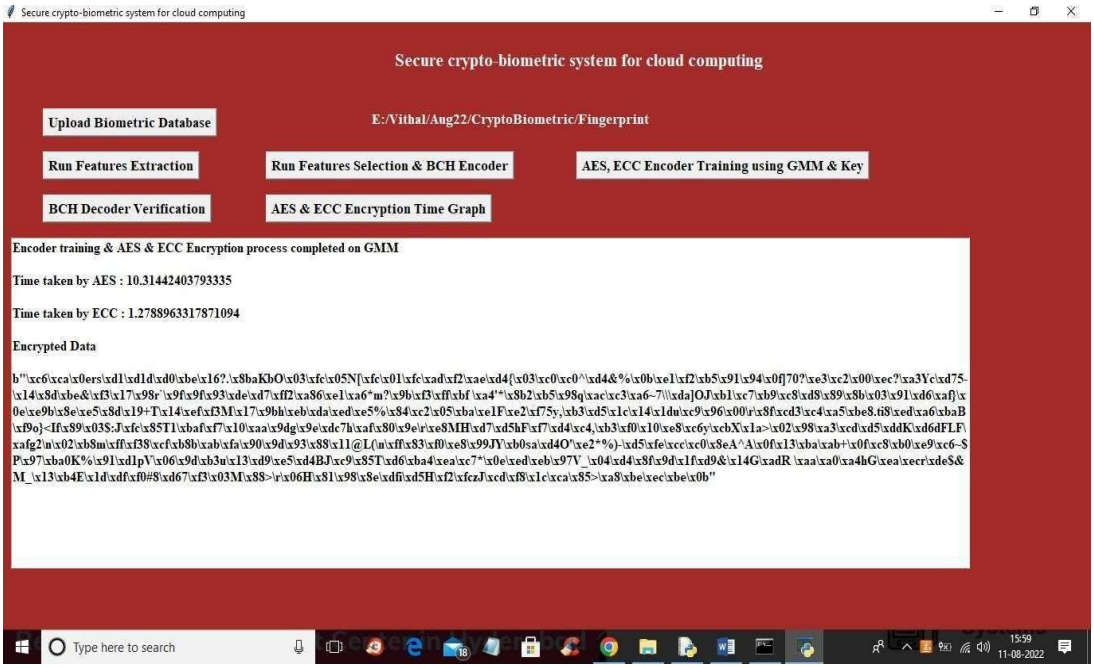
Screenshot 5.4: Run Features Selection & BCH Encoder



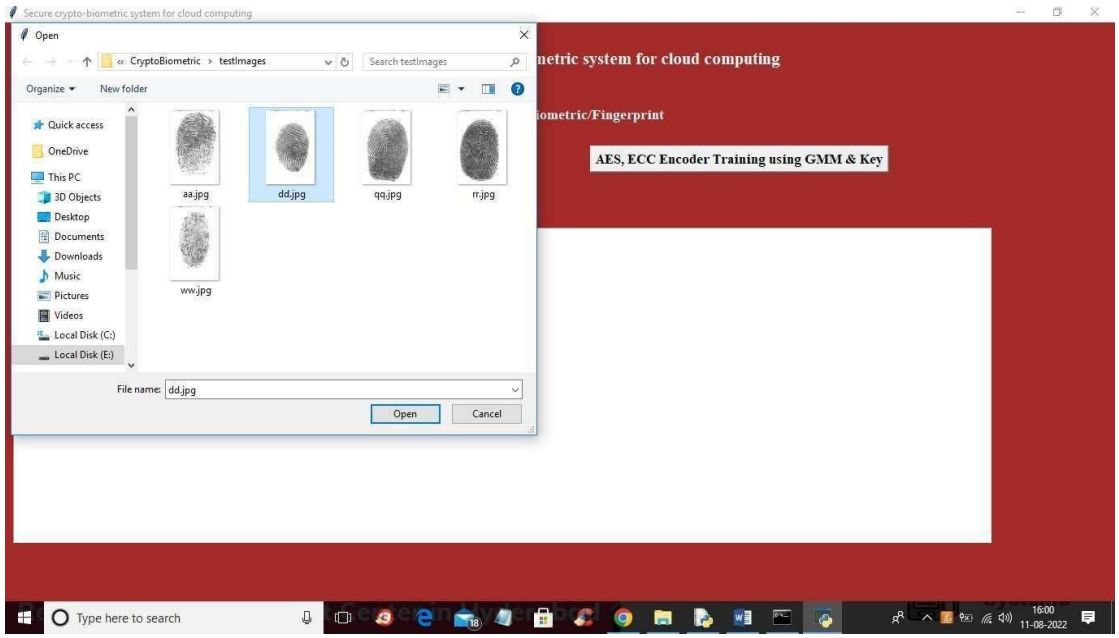
Screenshot 5.5: AES, ECC, Encoder Training using GMM & Key



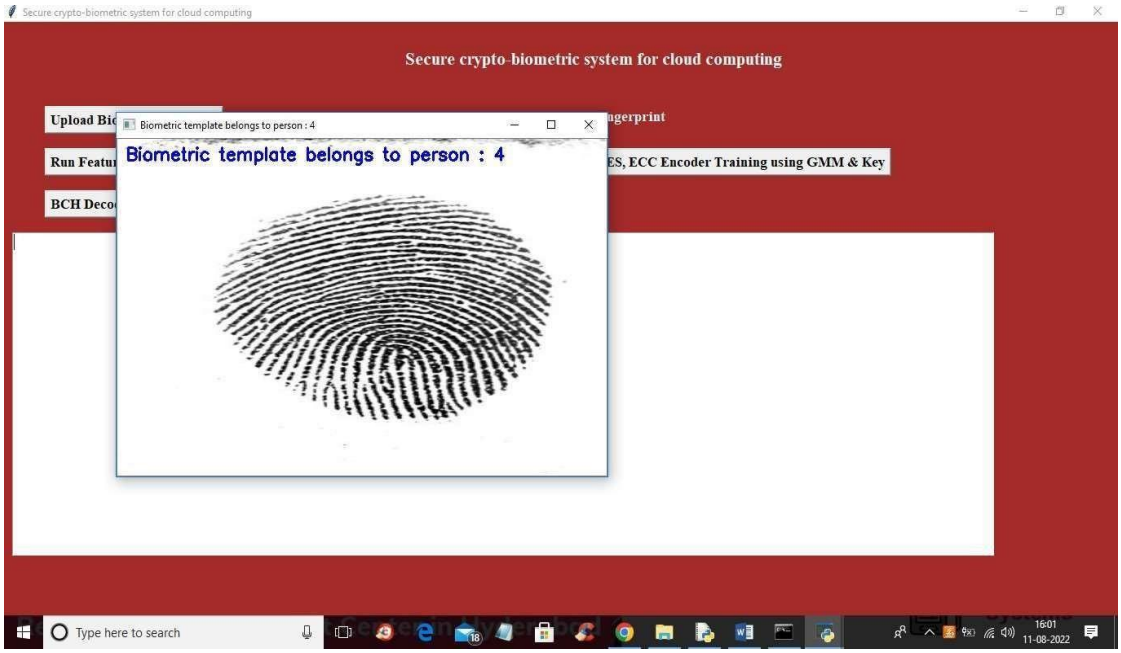
Screenshot 5.6: BCH Decoder Verification



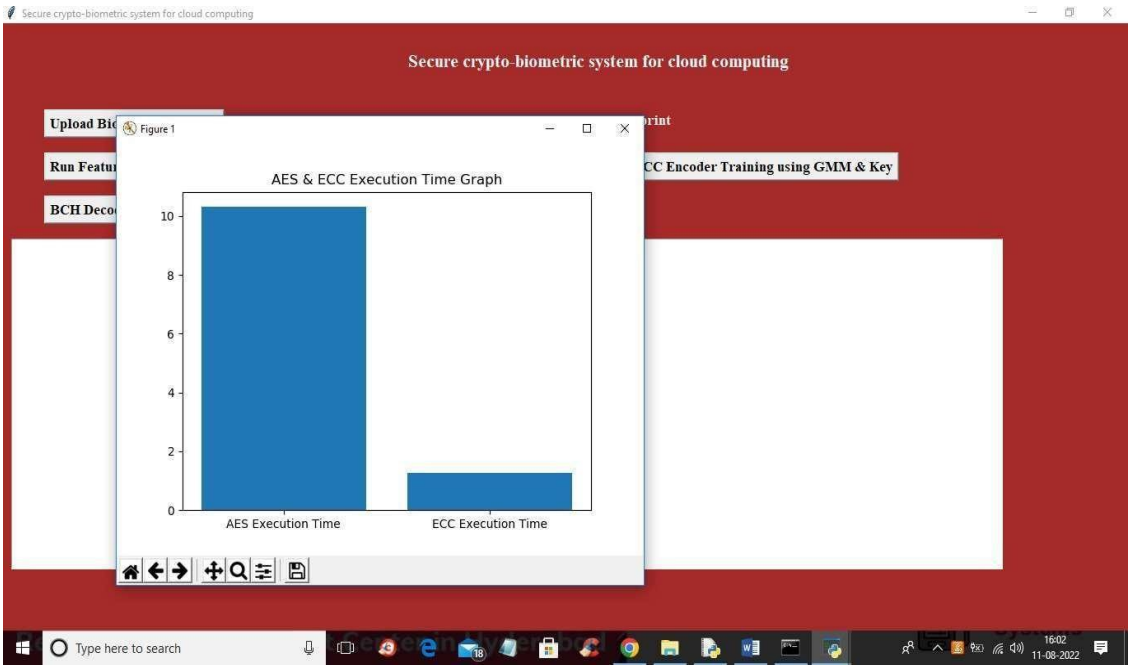
Screenshot 5.7: Upload finger template



Screenshot 5.8: AES and ECC Encryption time graph



Screenshot 5.9: Observe Execution time



6. TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

Testing is an integral and indispensable part of the software development and quality assurance process. Its fundamental purpose is to unearth errors and flaws within a work product, ensuring that the final software system is robust, reliable, and capable of meeting user expectations. Testing serves as the systematic process of meticulously scrutinizing every conceivable fault or weakness that might be present in a software product, ranging from individual components and subassemblies to complete assemblies or finished products. At its core, testing functions as a comprehensive evaluation mechanism, verifying the functionality, performance, and reliability of software components and systems.

The overarching objective is to confirm that the software not only adheres to its specified requirements but also aligns with user expectations. Equally important is the assurance that the software does not fail in any unacceptable manner under different operational conditions. To achieve these objectives, testing encompasses a wide array of test types, each addressing specific testing requirements.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Integration testing, like other testing methodologies, is fundamentally designed to uncover errors and inconsistencies. It aims to expose issues that may arise when previously tested components come together and interact. This testing approach endeavors to verify that the combined system behaves as expected, adhering to predefined requirements and meeting user expectations. Integration testing examines different integration points, including APIs, databases, communication protocols, and data transfer mechanisms.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstration that functions tested are available as specified by the business and technical requirements, system documentation and user manuals.

Functional testing is centered on the following items:

Valid Input : Identified classes of valid input must be accepted.

Invalid Input : Identified classes of invalid input must be rejected.

Functions : Identified functions must be exercised.

Output : Identified classes of application output must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases

6.2.4 SYSTEM TEST

System testing is a critical phase in the software testing process, with the primary objective of ensuring that the entire integrated software system aligns with the specified requirements. This type of testing evaluates the software configuration as a whole to verify that it produces known and predictable results. System testing often involves comprehensive scenarios and use cases, mimicking real-world interactions, in order to assess the software's behavior in a holistic manner.

A noteworthy example of system testing is the configuration-oriented system integration test, which is focused on examining how different components and modules are configured to function as a coherent whole. This type of testing is fundamentally grounded in process descriptions and flows, emphasizing pre-determined process links and integration points. It ensures that the software system functions seamlessly when all its components are integrated, ultimately meeting user expectations and fulfilling its intended purpose.

6.2.5 WHITE BOX TESTING

White Box Testing is a sophisticated testing approach that requires the software tester to possess knowledge of the inner workings, structure, and often the programming language of the software being tested. In this method, the tester has insight into the software's internal architecture, allowing them to delve into the logic and algorithms that drive its functionality. White Box Testing is typically employed to scrutinize areas of the software that are not easily accessible through black box testing, providing a more detailed and in-depth assessment of its internal mechanisms.

This form of testing is especially useful for identifying coding errors, logical flaws, and security vulnerabilities within the software. It aids in uncovering issues related to the software's control flow, data flow, and overall code quality. White Box Testing aims to improve the structural quality of the software by ensuring that it adheres to coding standards, meets design specifications, and performs as intended at the code level.

6.2.6 BLACK BOX TESTING

Black Box Testing stands in contrast to White Box Testing, as it involves testing the software without any knowledge of its inner workings, structure, or programming language. Testers approach the software as a "black box," focusing solely on its external behavior and functionality. This method does not require an understanding of the software's code or internal logic; instead, it evaluates how the software performs based on the inputs provided and the outputs it generates. Black Box Testing is conducted without considering the software's underlying mechanisms, making it ideal for assessing the software from an end-user perspective.

Test cases are designed based on defined specifications or requirements documents, and the test focuses on validating that the software functions as expected and complies with the documented criteria. This approach is valuable for validating user-facing features, usability, and overall system behavior. In essence, Black Box Testing is centered on evaluating what the software does rather than how it accomplishes it, making it an essential part of ensuring that the software meets user expectations and operates as intended.

6.3 TEST CASES

6.3.1 CLASSIFICATION

S.No	Test Type	Test Objective	Test Case Description	Expected Outcome
1	Authentication	Verify User Authentication using Biometric	1. Attempt Login with valid biometric data 2. Attempt login with invalid biometric data	1. Successful Login. 2. Authentication failure
2	Data Encryption	Ensure data is properly encryptedand decrypted	1. Encrypt sensitive data 2. Decrypt encrypted data	1. Data is encrypted successfully 2. Data is decrypted accurately
3	User Management	Test User accountcreation, modification, andaccess control	1. Create new user account 2. Modify User account	1. New user account created successfully 2. User account information updated successfully
4	Performance	Access System Performance undervarious conditions	1. Test System scalability. 2. Test System response time	1. Scalability is demonstrated 2. Acceptable response times areachieved
5	Security	Identify and mitigate securityvulnerabilities	1. Penetration testing to identify vulnerabilities 2. Test for data breaches	1. Vulnerabilities identified and demonstrated 2. No data breaches detected

7. CONCLUSION

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION

The "Secure Crypto-Biometric System for Cloud Computing" project marks a significant milestone in the field of cloud security and data privacy. In an increasingly data-centric world, where the safeguarding of sensitive user information is paramount, this project has successfully addressed the evolving challenges inherent to cloud computing. By combining biometric authentication and robust encryption techniques, our system has substantially improved security standards, ensuring that user data remains safeguarded from unauthorized access and potential breaches.

Moreover, the project prioritizes user convenience by introducing user-friendly biometric authentication methods, eliminating the need for complex password management. The system's meticulous approach to biometric data privacy, through advanced encryption and secure storage, underscores our commitment to protecting sensitive information.

In conclusion, the "Secure Crypto-Biometric System for Cloud Computing" project not only provides a solution to contemporary security challenges but also sets the stage for continued advancements in the domain of cloud computing, delivering secure, user-friendly, and privacy-conscious experiences.

7.2 FUTURE SCOPE

Multi-Platform Support: Extend the system's compatibility to work seamlessly on various platforms, including mobile devices and different operating systems, to caterto a broader user base.

Biometric Data Diversity: Expand the range of supported biometric data types (e.g., voice recognition, palm prints) to offer users more choices for authentication.

Blockchain Integration: Explore the integration of blockchain technology for additional data security and tamper-proof audit trails.

Machine Learning and AI: Utilize machine learning and artificial intelligence for advanced biometric recognition, improving accuracy and fraud detection.

8. BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 REFERENCES

[1] A. A. M. Abd Hamid, N. and A. Izani. Extended cubic b-spline interpolation method applied to linear two-point boundary value problem. World Academy of Science, 62, 2010.

[2] T. Acharya. Median computation-based integrated color interpolation and color space conversion methodology from 8-bit bayer pattern rgb color space to 24-bit cie xyz color space, 2002. US Patent 6,366,692.

[3] A. F. Agarap. An architecture combining convolutional neural network (cnn) and support vector machine (svm) for image classification. arXiv preprint arXiv:1712.03541, 2017.

[4] A. Ben-Hur and J. Weston. A user's guide to support vector machines. In Data mining techniques for the life sciences, pages 223– 239. Springer, 2010.

[5] S. Bianco, F. Gasparini, A. Russo, and R. Schettini. A new method for rgb to xyz transformation based on pattern search optimization. IEEE Transactions on Consumer Electronics, 53(3):1020–1028, 2007.

[6] E. Bisong. Google colaboratory. In Building Machine Learning and Deep Learning Models on Google Cloud Platform, pages 59–64. Springer, 2019.

[7] A. P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern recognition, 30(7):1145– 1159, 1997.

[8] S. A. Burney and H. Tariq. K-means cluster analysis for image segmentation. International Journal of Computer Applications, 96(4), 2014.

[9] M. A. Chandra and S. Bedi. Survey on svm and their application in image classification. International Journal of Information Technology, pages 1–11, 2018.

8.2 GITHUB LINK

https://github.com/saramohd02/secure-crypto-biometric-system-for-cloud -computing