A

Mini Project

On

DESIGNING SECURE AND EFFICIENT BIOMETRIC-BASED SECURE ACCESS MECHANISM FOR CLOUD SERVICES

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

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

BY

K.AKSHITH REDDY (217R1A0532)

P.DOLIKA (227R5A0503)

N.SUNIL (217R1A0540)

UNDER THE GUIDANCE OF

Mrs.NAJEEMA AFRIN

(Assistant 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.

2021-25

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the dissertation work entitled "**DESIGNING SECURE AND EFFICIENT BIOMETRIC – BASED SECURE ACCESS MECHANISM FOR CLOUD SERVICES**" being submitted by **K.Akshith Reddy(217R1A0532), P.Dolika(227R5A0503), N.Sunil(217R1A0540)** in partial fulfillment for the degree of Bachelor of Technology in "COMPTURE SCIENCE AND ENGINEERING" during the academic year 2024-2025.

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

Mrs.Najeema Afrin **Dr. A. Raji Reddy**

Assistant Professor **DIRECTOR**

INTERNAL GUIDE

Dr. Nuthanakanti Bhaskar **EXTERNAL EXAMINER**

HOD

Submitted for viva voice Examination held on _____

ACKNOWLEGDEMENT

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

Mrs.Najeema Afrin, Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her 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 Project Review Committee (PRC) Coordinators: **Dr. K.Maheswari, Dr. S.Suma, Mr. J.Narasimha Rao, Mrs. K.Shilpa, Mr. K.Ranjith Reddy** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. Nuthanakanti Bhaskar,** 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 throughout the course of this project. We would like to 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.

K.AKSHITH REDDY(217R1A0532)

P.DOLIKA(227R5A0503)

N.SUNIL(217R1A0540)

ABSTRACT

The demand for remote data storage and computation services is increasing exponentially in our data-driven society; thus, the need for secure access to such data and services. In this project, we design a new biometric-based authenticationprotocol to provide secure access to a remote (cloud) server.In the proposed approach, we consider biometric data of a user as a secret credential. We then derive a unique identity from the user's biometric data, which is further used to generate the user's private key.In addition, we propose an efficient approach to generate a session key between two communicating parties using two biometric templates for a secure message transmission.In other words, there is no need to store the user's private key anywhere and the session key is generated without sharing any prior information.The approach is validated as secure against various attacks.

LIST OF FIGURES

RESULTS AND DISCUSSIONS

TABLE OF CONTENTS

1.INTRODUCTION

1.INTRODUCTION

1.1 PROJECT SCOPE

This project is titled as “Designing Secure and Efficient Biometric-Based Secure Access Mechanism for Cloud Services”. The project aims to develop a secure biometric authentication system using fingerprint and facial recognition for cloud access This project uses machine-learning methods. First, we use a convolutional neural networks to classify fingerprint and facial key points for each image. We then compare whether the images were matched or not,if matched we can acess the files in the cloud storage and also upload the files or else we cant access the files.

1.2 PROJECT PURPOSE

The purpose of the project is to design and implement a secure, efficient, and user-friendly biometric authentication mechanism for accessing cloud services. By utilizing fingerprint and facial recognition, the project aims to provide enhanced security and data protection for cloud users, while reducing reliance on traditional password-based methods. The ultimate goal is to ensure that only authorized users can access cloud resources easily.Additionally, the project will focus on optimizing performance to provide fast and seamless authentication

1.3 PROJECT FEATURES

Utilizes fingerprint and facial recognition technology for secure user authentication. Secure Cloud Integration, this helps in Seamlessly integrates with various cloud services, allowing users to authenticate and access resources securely. Designed to accommodate a growing number of users, maintaining performance even as demand increases. Optimized for quick authentication processes, minimizing delays for users accessing cloud services.

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 increasing reliance on cloud services for sensitive data storage has heightened the need for secure user authentication. Traditional password-based systems are vulnerable to theft, weak password choices, and user fatigue, leading to unauthorized access. This project aims to address these issues by implementing a biometric authentication mechanism using fingerprint and facial recognition, providing a secure and user-friendly solution that enhances cloud service security and meets data protection regulations.

2.2 EXISTING SYSTEM:

Over the past years, multiple approaches have been proposed to solve the problem of Secure Biometric Detection.A number of authentication mechanisms have been proposed in the many areas, such as those based on Kerberos , Oauth.FIDO Standards for secure biometric authentication, used by various cloud services to ensure strong user verification.The FIDO (Fast Identity Online) standard is widely used for improving security in cloud services, particularly in the context of authentication. FIDO aims to eliminate the reliance on passwords and provides a more secure, user-friendly method of authenticating.

2.2.1 LIMITATIONS OF THE EXISTING SYSTEM:

- Limited Accuracy in certain conditions.
- High Computational Cost..
- False Positive/Negative
- Recovery

2.3 PROPOSED SYSTEM:

- In the proposed approach, we consider a fingerprint,face image of a user as a secret credential.
- From the fingerprint,face image, we generate a private key that is used to enroll the user's credential secretly in the database of an authentication server.
- In the authentication phase, we capture a new biometric fingerprint,face image of the user, and subsequently generate the private key and encrypt the biometric data as a query.
- This queried biometric data is then transmitted to the authentication server for matching with the stored data.
- Once the user is authenticated successfully, he/she is ready to access his/her service from the desired server. Using two datas, we present a fast and robust approach to generate the session key.
- We uses the CNN Algorithm.

2.3.1 ADVANTAGES OF PROPOSED SYSTEM:

- Enhanced Security.
- Improved User Experience.
- Fraud Detection and Prevention.
- Scalability and Efficiency.
- Accessibility.

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. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY:

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

•The costs conduct a full system investigation.

•The cost of the hardware and software.

•The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

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. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 BEHAVIORAL FEASIBILITY:

This includes the following questions:

•Is there sufficient support for the users?

•Will the proposed system cause harm?

The project would be beneficial because it satisfies the objectives when developed and installed. All behavioral aspects are considered carefully and conclude that the project is behaviorally feasible.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

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

- Processor : Intel Core i5
- Hard disk : 40GB
- RAM : 4GB and above

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 10 or 11
- Coding Language : Python 3.7

3.ARCHITECTURE

3.SYSTEM ARCHITECTURE

3.1 PROJECT ARCHITECTURE

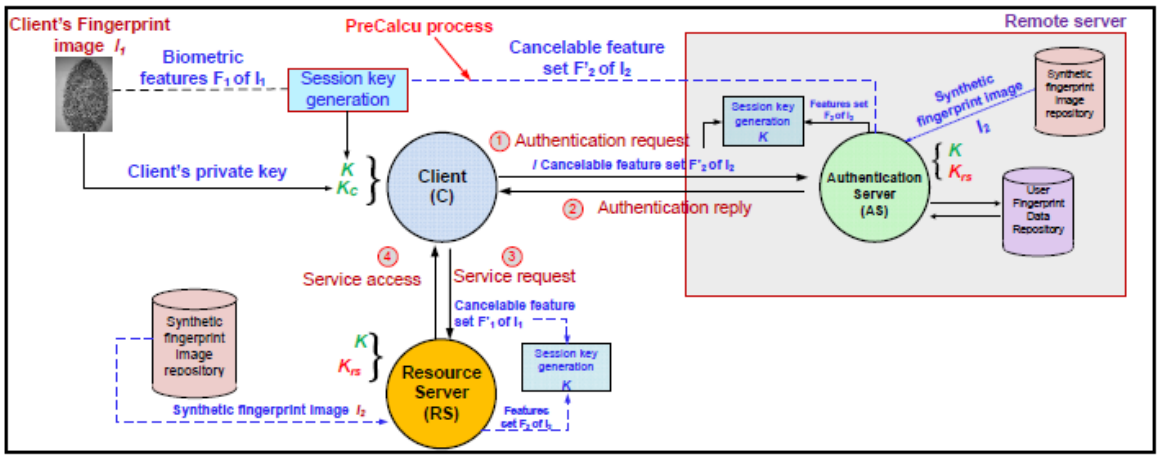

Figure 3.1: Designing Secure and Efficient Biometric-Based Secure Access Mechanism for Cloud Services

3.2 DESCRIPTION:

1).Biometric Feature Extraction:The client captures a biometric sample (like a fingerprint).Unique features from this sample are extracted to form a set of biometric

2).Session Key Generation:The client generates a session key for secure communication with the authentication and resource servers.This key is derived from the extracted biometric features or a combination of the client's credentials, ensuring it is unique for each session.

3).Authentication Request:The client sends an authentication request to the authentication server, including the extracted biometric features and session key.This request verifies the client's identity without exposing sensitive biometric data.

4).Verification by Authentication Server:The authentication server receives the request and compares the provided biometric features with the stored features in its database.It checks the validity of the session key to ensure the request is genuine.\

5).Authentication Reply:If the biometric features match the stored data, the authentication server sends an authentication reply back to the client, indicating successful verification.If the features do not match, an error response is sent.

6).Service Request:Upon receiving a successful authentication reply, the client requests access to specific services from the resource server.This request may include the session key for secure access.

7).Access Granted by Resource Server:The resource server processes the service request, verifies the session key, and checks the validity of the client's authentication.If everything checks out, the resource server grants the client access to the requested services.

8).We can able to access the resources.

3.3 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

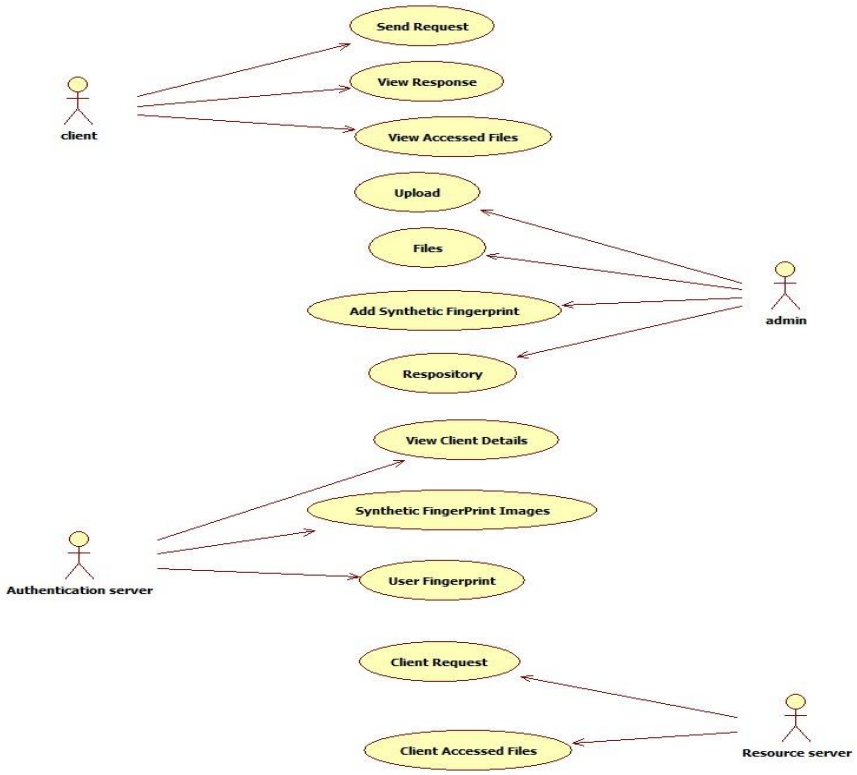

Figure 3.3: Use Case Diagram for Designing Secure And Efficient Biometric Based Secure Access Mechanism For Cloud Services.

This diagram represents a biometric-based authentication and file management system with four main actors: the Client, Admin, Authentication Server, and Resource Server. The Client initiates interactions by sending requests, viewing responses, accessing files, and uploading new files. The Admin has management capabilities, such as viewing and organizing files, adding synthetic fingerprints for security, and accessing client details stored in the repository.The Authentication Server is responsible for verifying the client's identity using synthetic fingerprint images and user fingerprints, processing client requests securely. Once authenticated, the Resource Server grants the client access to specific files. This structured interaction ensures secure and efficient file access, supported by robust biometric authentication across different system components.

3.4 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) 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 the classes. It explains which class contains information.

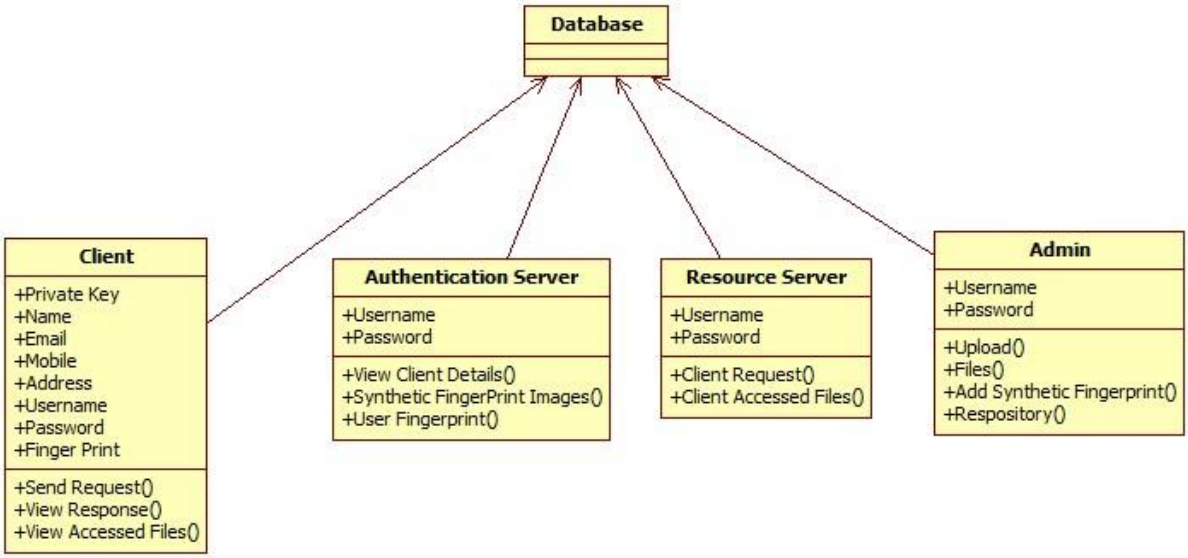

Figure 3.4: Class Diagram for Designing Secure And Efficient Biometric Based Secure Access Mechanism For Cloud Services

The class diagram illustrates a system with four main components: Client, Authentication Server, Resource Server, and Admin. All components interact with a central Database.

The Client class represents the end-user, with attributes like name, email, and fingerprint. The Authentication Server handles user authentication and authorization, while the Resource Server manages and provides access to resources. The Admin class has administrative privileges for system management and data manipulation.

The diagram shows relationships and interactions between these components, including methods for authentication, resource access, and data management. This system likely involves secure access control, biometric authentication, and efficient resource management.

3.5 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

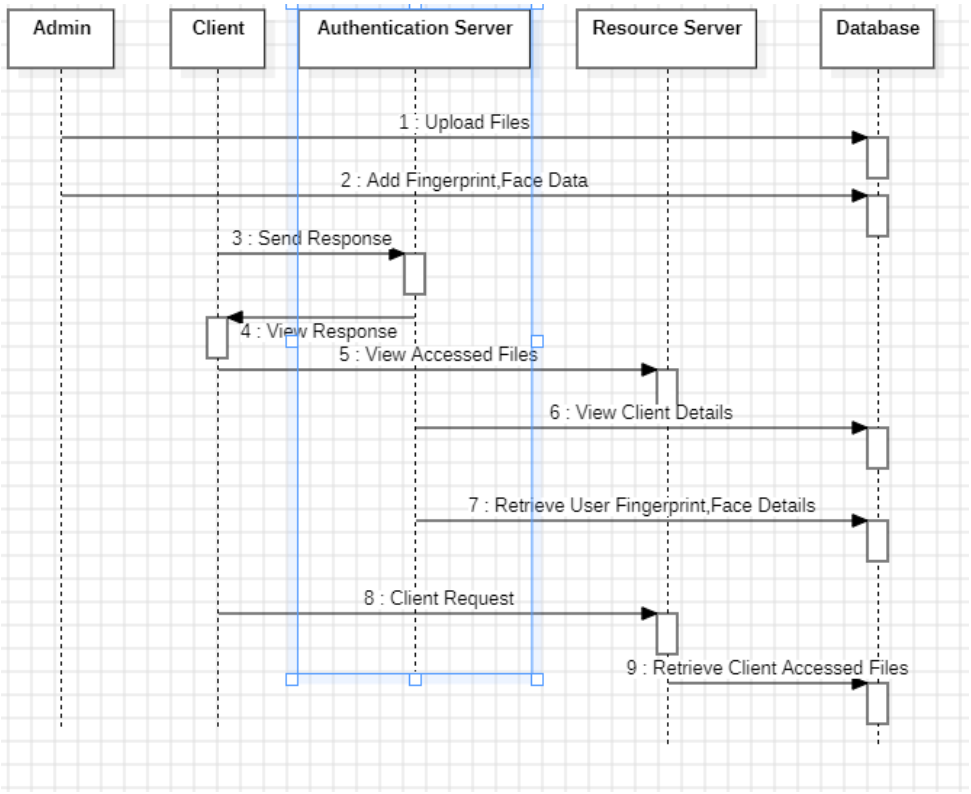

Figure 3.5: Sequence Diagram for Designing Secure And Efficient Biometric Based Secure Access Mechanism For Cloud Services

3.6 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency.

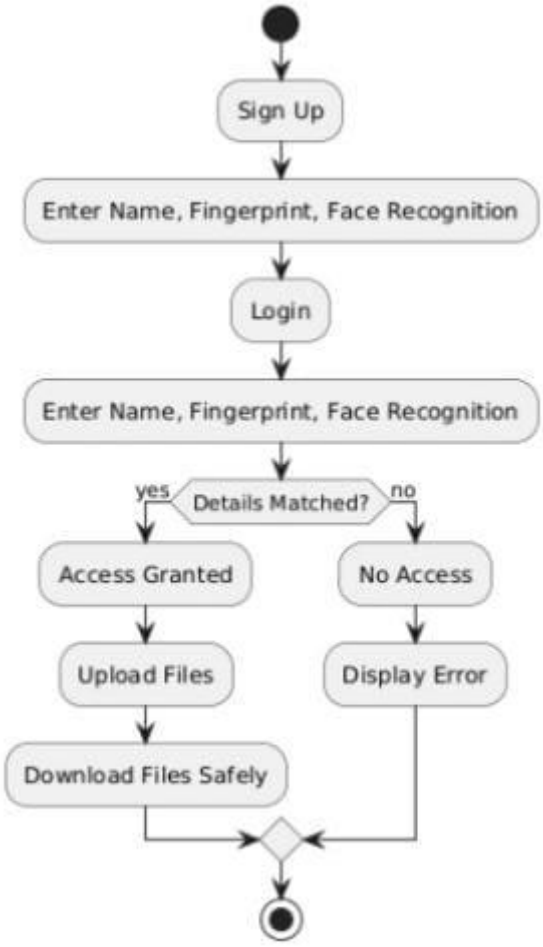

Figure 3.6 Activity Diagram for Designing Secure And Efficient Biometric Based Secure Access Mechanism For Cloud Services

This Activity Diagram illustrates a process for user authentication and file access. The process begins with either signing up or logging in. In both cases, the user is required to enter their name, fingerprint, and facial recognition data.

4.IMPLEMENTATION

4. IMPLEMENTATION

4.1 Sample Code

```
from django.shortcuts import render
from django.template import RequestContext
from django.contrib import messages
from django.http import HttpResponse
import os
from django.core.files.storage import FileSystemStorage
import pymysql
from PIL import Image
import cv2
import base64
import numpy as np
import ftplib
import urllib
global username, password, contact, gender, email, address, finger, finger_image
face_detection = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
recognizer = cv2.face_LBPHFaceRecognizer.create()
def DownloadFileAction(request):
    if request.method == 'GET':
        global username
        img = request.GET.get('fname', False)
        infile = open("SecureBiometricApp/static/files/"+img, 'rb')
        data = infile.read()
        infile.close()
        response = HttpResponse(data, content_type='text/plain')
        response['Content-Disposition'] = 'attachment; filename=%s' % img
        return response
```

```
def Download(request):
    if request.method == 'GET':
        global username
        font = '<font size="" color="black">'
     con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'securebiometric',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select * FROM upload")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == username:
                    output+="<tr><td>"+font+row[0]+"</td><td>"+font+row[1]+"</td>"
                     output+='<td><a href=\'DownloadFileAction?fname='+row[1]+'\'><font size=3 color=black>Click Here</font></a></td></tr>'
        context= {'data':output}
        return render(request, "Download.html", context)
def UploadAction(request):
    if request.method == 'POST':
        global username
        file = request.FILES['t1']
        filename = request.FILES['t1'].name
        fs = FileSystemStorage()
        fs.save('SecureBiometricApp/static/files/'+filename, file)
        ftp = ftplib.FTP_TLS("ftp.drivehq.com")
        ftp.login("cdaproject", "Offenburg965#")
        ftp.prot_p()
        file = open('SecureBiometricApp/static/files/'+filename, "rb")
        ftp.storbinary("STOR "+filename, file)
        file.close()
```

```
        ftp.close()
        db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'securebiometric',charset='utf8')
        db_cursor = db_connection.cursor()
student_sql_query ="INSERT INTO upload(username,filename) VALUES('"+str(username)+"','"+filename+"')"
        db_cursor.execute(student_sql_query)
        db_connection.commit()
        print(db_cursor.rowcount, "Record Inserted")
        if db_cursor.rowcount == 1:
            output = filename+' saved at driveHQ Cloud'
        context= {'data':output}
        return render(request, 'Upload.html', context)
def Upload(request):
    if request.method == 'GET':
        return render(request, 'Upload.html', {})
def ValidateFace(request):
    if request.method == 'GET':
        return render(request, 'ValidateFace.html', {})
def Login(request):
    if request.method == 'GET':
        return render(request, 'UserLogin.html', {})
def index(request):
    if request.method == 'GET':
        return render(request, 'index.html', {})
def Signup(request):
    if request.method == 'GET':
        return render(request, 'Signup.html', {})
def getUserImages():
    names = []
```

```
    ids = []
    faces = []
    dataset = "SecureBiometricApp/static/profile"
    count = 0
    for root, dirs, directory in os.walk(dataset):
        for j in range(len(directory)):
            pilImage = Image.open(root+"/"+directory[j]).convert('L')
            imageNp = np.array(pilImage,'uint8')
            name = os.path.splitext(directory[j])[0]
            names.append(name)
            faces.append(imageNp)
            ids.append(count)
            count = count + 1
    print(str(names)+" "+str(ids))
    return names, ids, faces
def getName(predict, ids, names):
    name = "Unable to get name"
    for i in range(len(ids)):
        if ids[i] == predict:
            name = names[i]
            break
    return name
def ValidateFaceAction(request):
    if request.method == 'POST':
        global username
        status = "unable to predict user"
        img = cv2.imread('SecureBiometricApp/static/photo/test.png')
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        face_component = None
```

```
faces=face_detection.detectMultiScale(img,scaleFactor=1.1,minNeighbors=5,minSize=(30,30),flags=cv2.CASCADE_SCALE_IMAGE)
        status = "Unable to predict.Please retry"
        #for (x, y, w, h) in faces:
        #    face_component = gray[y:y+h, x:x+w]
        faces = sorted(faces, reverse=True,key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
        (fX, fY, fW, fH) = faces
        face_component = gray[fY:fY + fH, fX:fX + fW]
        if face_component is not None:
            names, ids, faces = getUserImages()
            recognizer.train(faces, np.asarray(ids))
            predict, conf = recognizer.predict(face_component)
            print(str(predict)+" === "+str(conf))
            if(conf < 80):
                validate_user = getName(predict, ids, names)
                print(str(validate_user)+" "+str(username))
                if validate_user == username:
                    status = "success"
        else:
            status = "Unable to detect face"
        if status == "success":
            context= {'data':"Welcome "+username+" Both finger & face successfully matched"}
            return render(request, 'UserScreen.html', context)
        else:
            context= {'data':status+". Please try again"}
            return render(request, 'ValidateFace.html', context)
def UserLoginAction(request):
    global username, finger
    if request.method == 'POST':
```

```
    username = request.POST.get('t1', False)
    password = request.POST.get('t2', False)
    finger_image = request.FILES['t3'].read()
    index = 0
    msg = "Login or finger matching failed"
    finger = ""
    page = 'UserLogin.html'
        con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'securebiometric',charset='utf8')
    with con:
        cur = con.cursor()
        cur.execute("select username,password,finger FROM signup")
        rows = cur.fetchall()
        for row in rows:
            print(str(row[2]))
            if row[0] == username and password == row[1]:
                finger = row[2]
                index = 1
                break
    if index == 1:
        with open('SecureBiometricApp/static/finger/'+finger, "rb") as file:
            content = file.read()
        file.close()
        if content == finger_image:
            msg = "Login & Finger Matching Successful"
            page = 'ValidateFace.html'
    context= {'data':msg}
    return render(request, page, context)
def SignupAction(request):
```

```
    if request.method == 'POST':
        global username, password, contact, gender, email, address, finger, finger_image
        username = request.POST.get('t1', False)
        password = request.POST.get('t2', False)
        contact = request.POST.get('t3', False)
        gender = request.POST.get('t4', False)
        email = request.POST.get('t5', False)
        address = request.POST.get('t6', False)
        finger_image = request.FILES['t7']
        finger = request.FILES['t7'].name
        output = "none"
            con = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database = 'securebiometric',charset='utf8')
        with con:
            cur = con.cursor()
            cur.execute("select username FROM signup")
            rows = cur.fetchall()
            for row in rows:
                if row[0] == username:
                    output = username+" Username already exists"
                    break
        if output == 'none':
            fs = FileSystemStorage()
            filename = fs.save('SecureBiometricApp/static/finger/'+finger, finger_image)
            context= {'data':username+" please capture your face"}
            return render(request, 'CaptureFace.html', context)
        else:
            context= {'data':username+" already exists"}
            return render(request, 'Signup.html', context)
```

```
def WebCam(request):
    if request.method == 'GET':
        data = str(request)
        formats, imgstr = data.split(';base64,')
        print(data)
        if os.path.exists("SecureBiometricApp/static/photo/test.png"):
            os.remove("SecureBiometricApp/static/photo/test.png")
        with open('SecureBiometricApp/static/photo/test.png', 'wb') as f:
            f.write(data)
        f.close()
        context= {'data':"done"}
        return HttpResponse("Image saved")
def CaptureFaceAction(request):
    if request.method == 'POST':
        global username, password, contact, gender, email, address, finger
        img = cv2.imread('SecureBiometricApp/static/photo/test.png')
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        face_component = None
        faces = face_detection.detectMultiScale(gray, 1.3,5)
        for (x, y, w, h) in faces:
            face_component = img[y:y+h, x:x+w]
        if face_component is not None:
            cv2.imwrite('SecureBiometricApp/static/profile/'+username+'.png',face_component)
            db_connection = pymysql.connect(host='127.0.0.1',port = 3306,user = 'root', password = 'root', database =
'securebiometric',charset='utf8')
            db_cursor = db_connection.cursor()
            student_sql_query = "INSERT INTO signup(username,password,contact_no,gender,email,address,finger)
VALUES('"+username+"','"+password+"','"+contact+"','"+gender+"','"+email+"','"+address+"','"+finger+"')"
            db_cursor.execute(student_sql_query)
            db_connection.commit()
```

```
        print(db_cursor.rowcount, "Record Inserted")
        if db_cursor.rowcount == 1:
            context= {'data':'Signup Process Completed'}
            return render(request, 'Signup.html', context)
        else:
            context= {'data':'Unable to detect face. Please retry'}
            return render(request, 'CaptureFace.html', context)
```

5.RESULTS AND DISCUSSIONS

5. RESULTS AND DISCUSSIONS

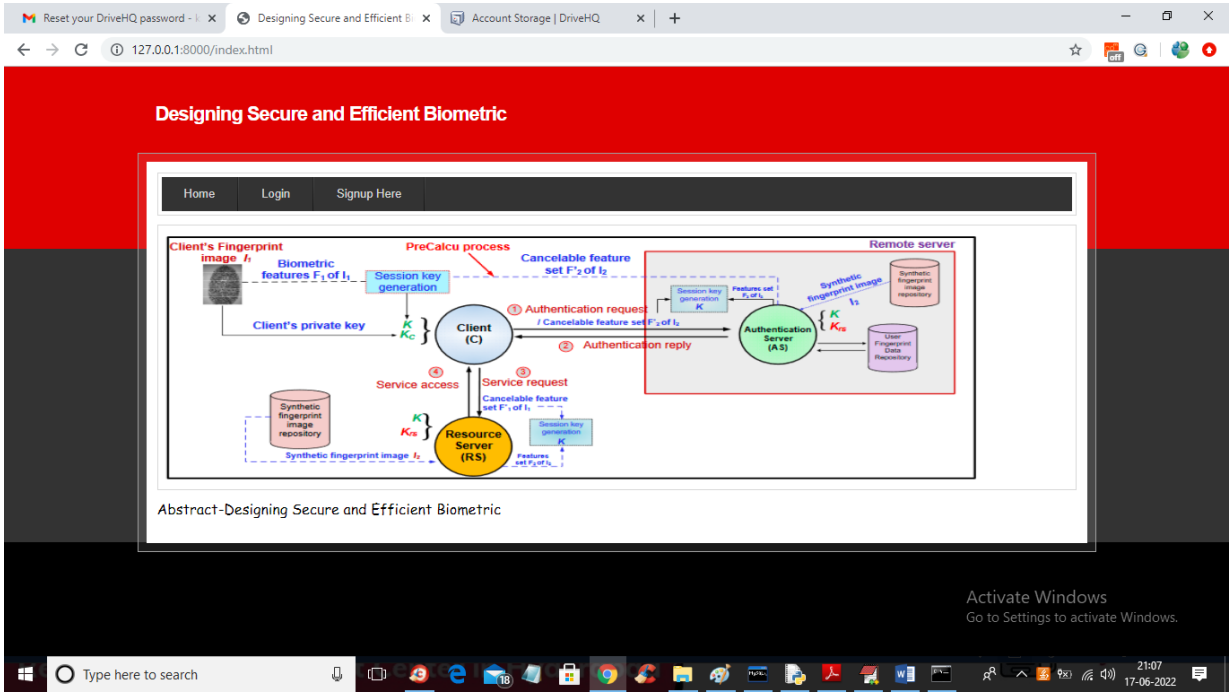

Figure 5.1:Home Page

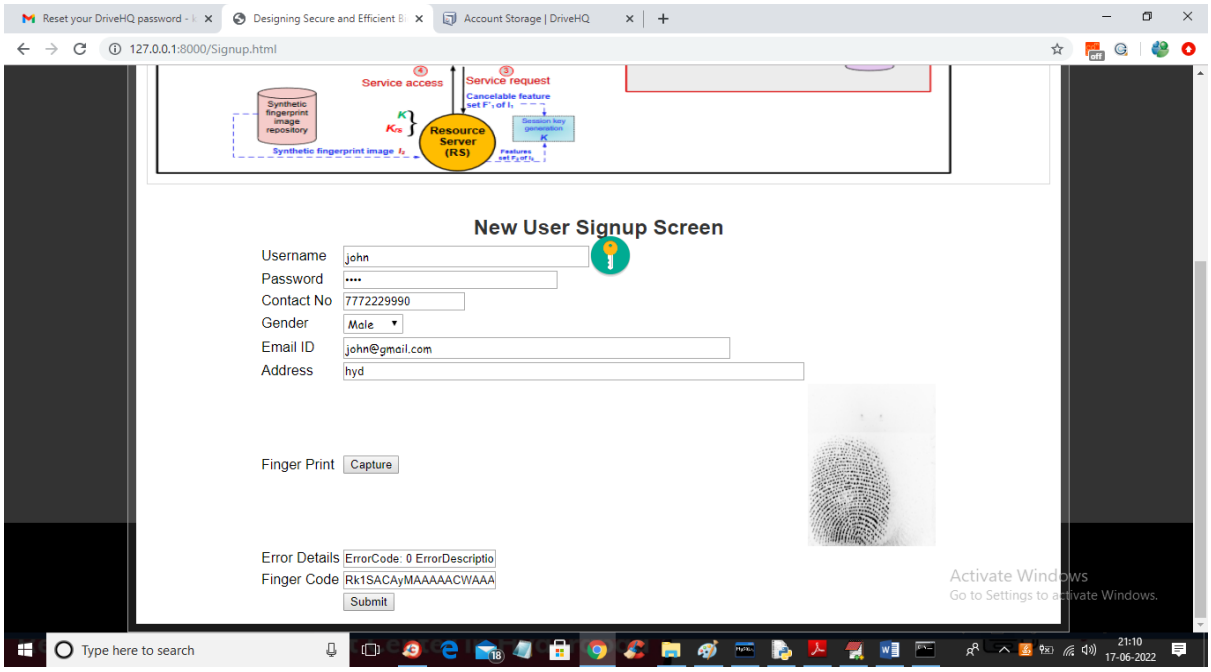

Figure 5.2:User Signup Page

User need to enter the details for signup.

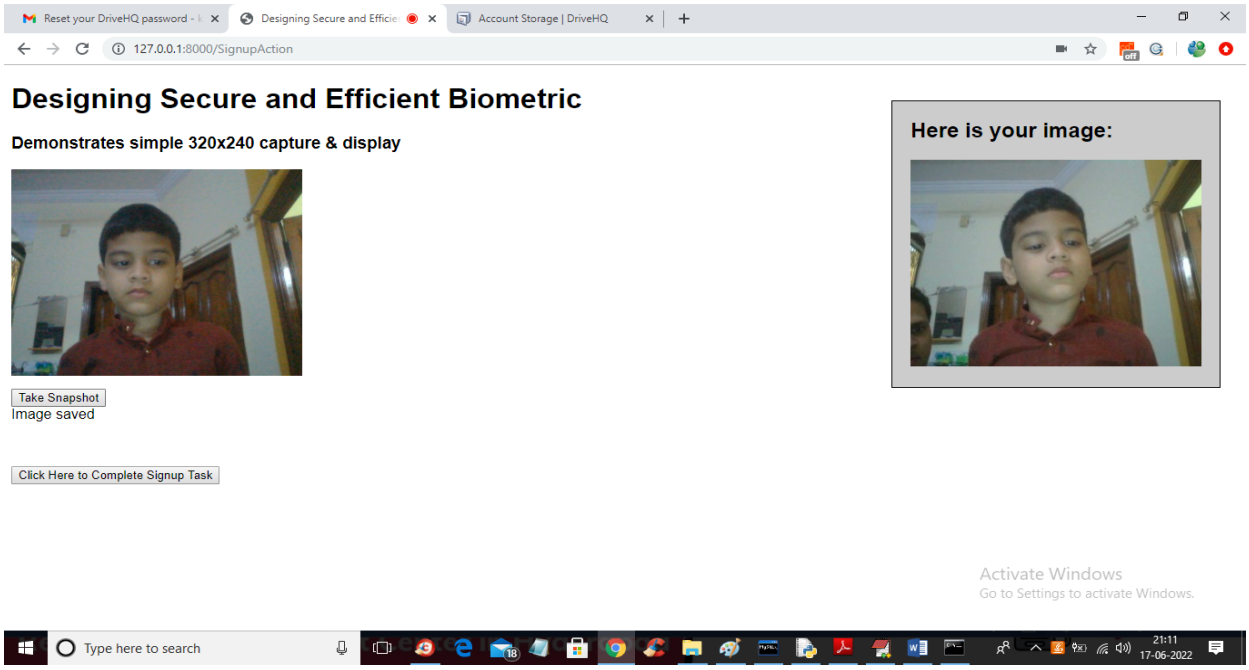

Figure 5.3: Face Snapshot

In above screen captured the face by clicking on 'Take Snapshot' and then pressed on 'Click Here to Complete Signup Task' button to get below output

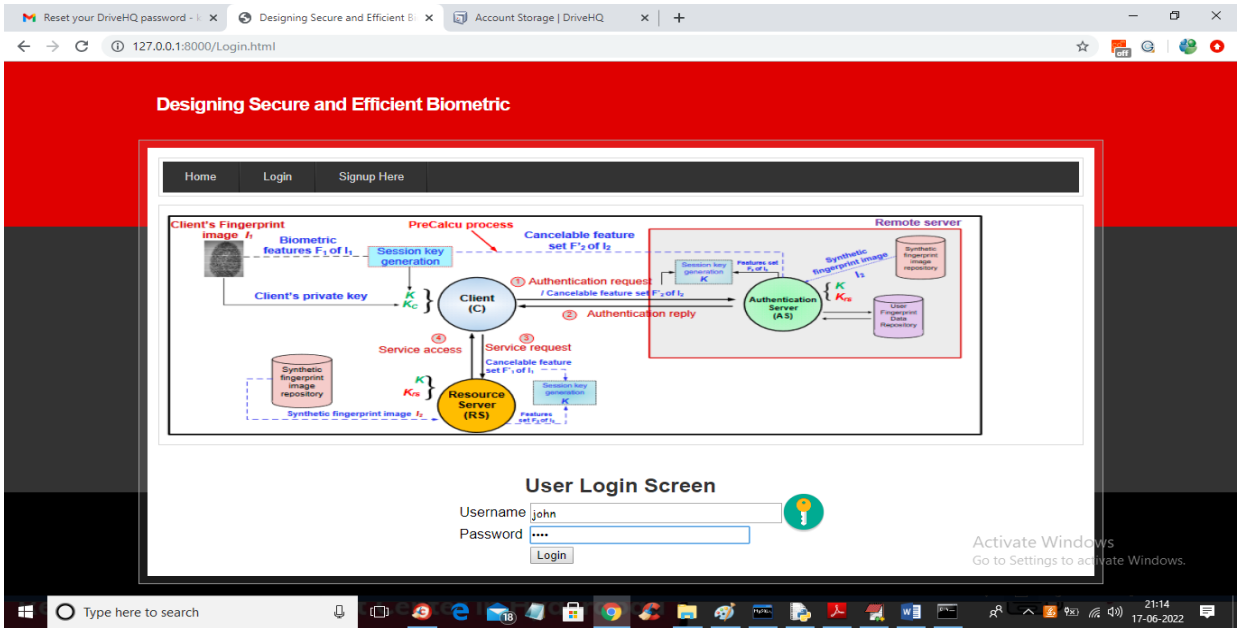

Figure 5.4: Login Page

In this step we need to login with the same fingerprint and face

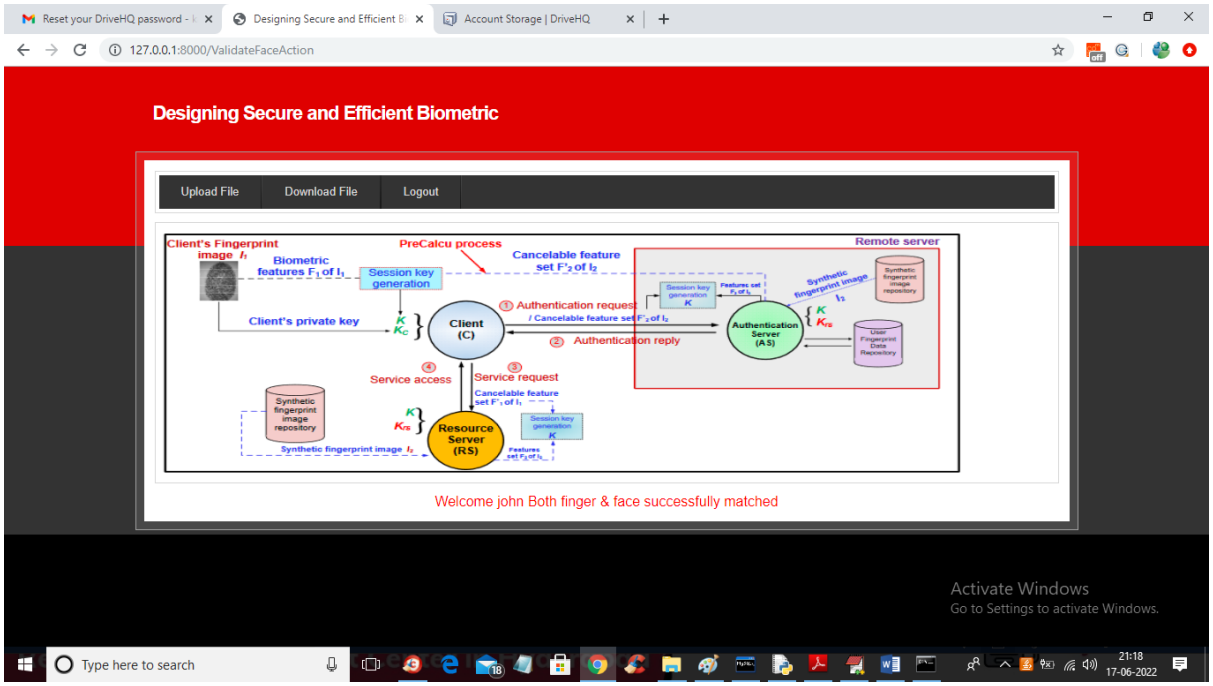

Figure 5.5: Upload Files

In above screen in red colour text we can see both face and finger matched successfully and now click on 'Upload File' button to get below screen

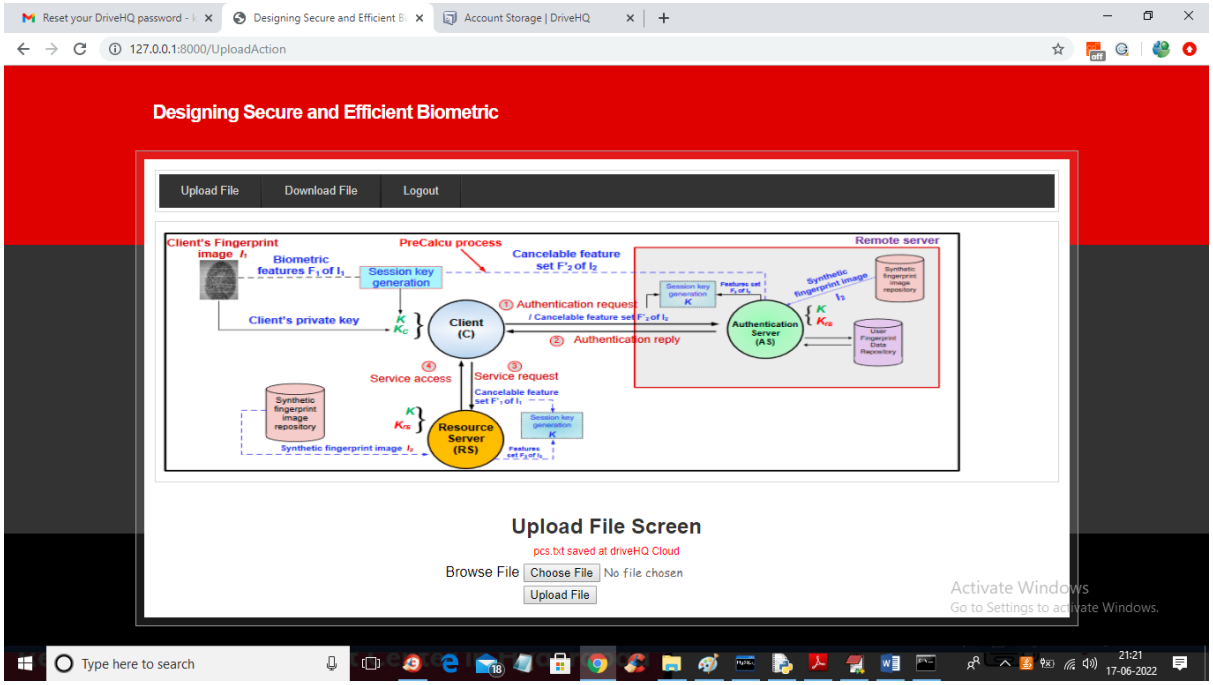

Figure 5.6: Uploaded Files to DriveHQ

In above screen we can see 'pcs.txt' file saved in cloud DriveHQ server and now open DRIVEHQ by entering URL as 'drivehq.com' and then enter username as 'cdaproject' and password as 'Offenburg965#' to get below screen

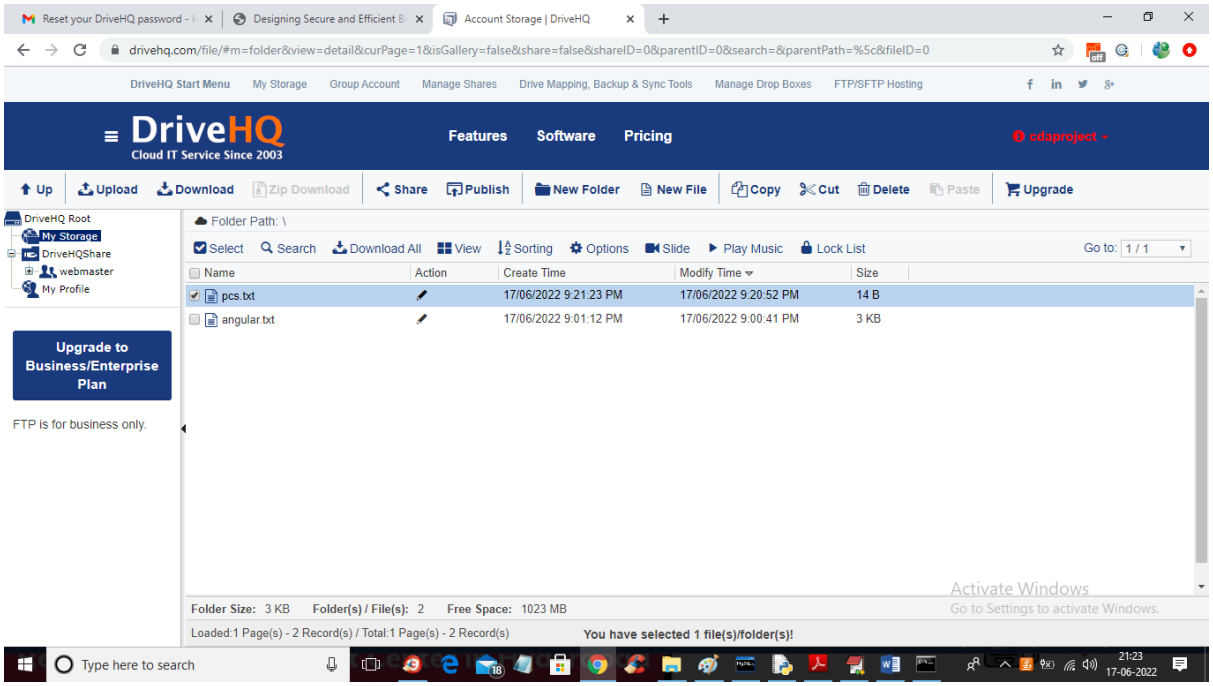

Figure 5.7: View Uploaded Files

In above screen we can see 'pcs.txt' file saved in DRIVEHQ and in application click on 'Download' link to get below screen

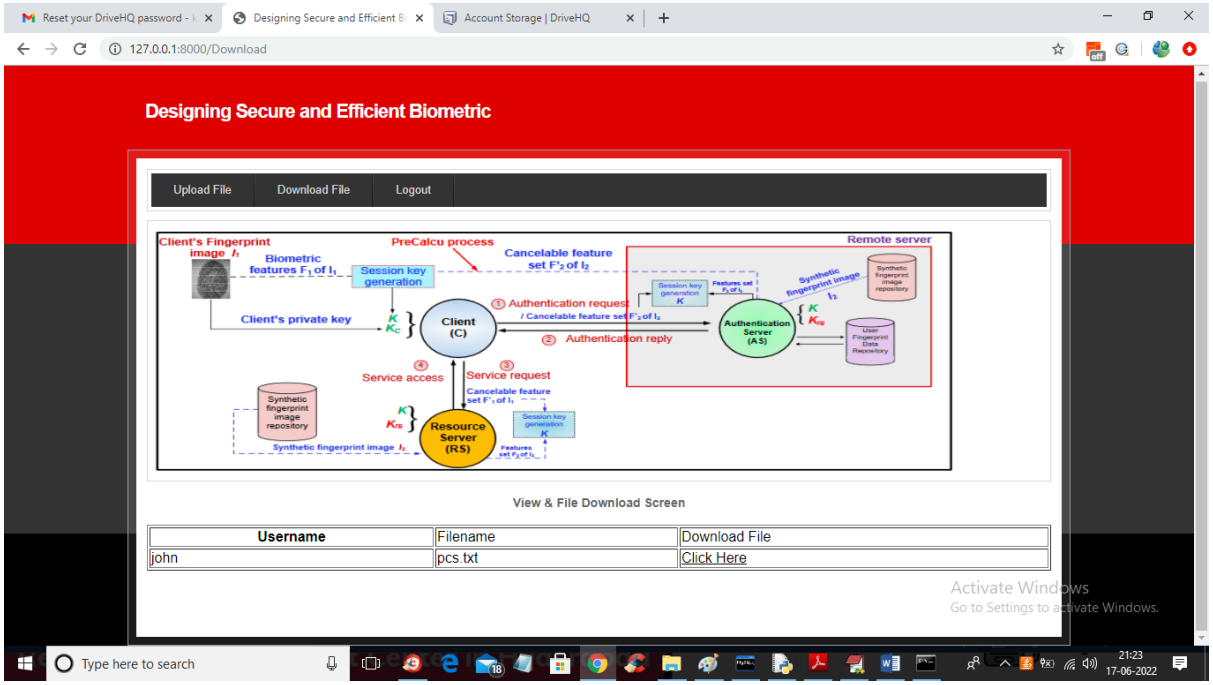

Figure 5.8: Download Files

In above screen user can view all files uploaded and then press 'Click Here' link to download that file and we can upload any number of files.

6.TESTING

6.TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

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. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, 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.

6.2.3 FUNCTIONAL TESTING:

Functional tests provide systematic demonstrations 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 outputs 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. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

6.3 TEST CASES

6.3.1 LOGIN FORM

FUNCTION:	**LOGIN**
EXPECTED RESULTS:	Should Validate the user and check his existence in database
ACTUAL RESULTS:	Validate the user and checking the user against the database
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

6.3.2 USER SIGNUP FORM

FUNCTION:	**USER REGISTRATION**
EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	**No**
HIGH PRIORITY	**Yes**

7.CONCLUSION

7.CONCLUSION & FUTURE SCOPE

7.1 PROJECT CONCLUSION:

Biometric has its unique advantages over conventional password and token-based security system, as evidenced by its increased adoption (e.g., on Android and iOS devices). In this paper, we introduced a biometric-based mechanism to authenticate a user seeking to access services and computational resources from a remote location. Our proposed approach allows one to generate a private key from a fingerprint biometric reveals, as it is possible to generate the same key from a fingerprint of a user with 95.12% accuracy. Our proposed session key generation approach using two biometric data does not require any prior information to be shared. A comparison of our approach with other similar authentication protocols reveals that our protocol is more resilient to several known attacks. Future research includes exploring other biometric traits.

7.2 FUTURE SCOPE:

Now-a-days its very important to store the data in the cloud which will be very secure and also easy data sharing and large amount of the data storage capacity.

It can be expanded to integrate multi-modal biometric systems, such as combining fingerprint, face, and recognition for more robust authentication.

The project could also explore the use of machine learning to enhance the accuracy and adaptability of biometric feature matching.

8.BIBLOGRAPHY

8.BIBLOGRAPHY

8.1 REFERENCES

[1] G.Wettstein, J. Grosen, and E. Rodriguez, “IDFusion: An open architecture for Kerberos based authorization,” Proc. AFS and Kerberos Best Practices Workshop, June

[2] S.Zhu, S.Setia, and S.Jajodia, “LEAP: efficient security mechanisms for large-scale distributed sensor networks,” Washington D.C., USA, October 2003, pp. 62–72. .

[3] W. Yang, S. Wang, J. Hu, G. Zheng, and C. Valli, “Security and Accuracy of Fingerprint-Based Biometrics: A Review,” Symmetry, vol. 11, no. 2, 2019. [Online]. Available: https://www.mdpi.com/2073-8994/11/2/141

[4] Q Jiang,J. Ma, X. Lu, and Y. Tian, “An efficient two-factor user authentication scheme with unlinkability for wireless sensor networks,” Peer-to-Peer Networking and Applications, vol. 8, no. 6, pp. 1070–1081, 2015.

[5] O.Althobaiti, M. Al-Rodhaan, and A. Al-Dhelaan, “An efficient biometric authentication protocol for wireless sensor networks,” International Journal of Distributed Sensor Networks, vol. 2013, pp. 1–13, 2013, Article ID 407971, http://dx.doi.org/ 10.1155/2013/407971.

[6] D. Kang, J. Jung, H. Kim, Y. Lee, and D. Won, “Efficient and Secure Biometric-Based User Authenticated Key Agreement Scheme with Anonymity,” Security and Communication Networks,vol.2018,pp.1–14,2018, Article ID 9046064, https://doi.org/10.1155/2018/9046064.

8.2 GITHUB LINK:

https://github.com/akshithkethireddy/Secure-Biometric-Access-for-Cloud-Services