

A CONTEMPORARY CLOUD BASED DYNAMIC AUTHENTICATION SYSTEM FOR MOBILE APPLICATIONS

submitted in partial fulfilment of the requirements for the award of
Bachelor of Technology Degree in Information Technology

by

ABIRAMI M (38120005)

SARAH PHILOMIN V (38120077)



DEPARTMENT OF INFORMATION TECHNOLOGY

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with Grade “A” by NAAC

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119, TAMIL NADU**

MAY – 2022



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with —A1 grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in



DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Abirami M (38120005)** and **V Sarah Philomin (38120077)** who carried out the project entitled **“A CONTEMPORARY CLOUD BASED DYNAMIC AUTHENTICATION SYSTEM FOR MOBILE APPLICATIONS”** under my supervision from **JANUARY 2022** to **MAY 2022**.

Internal Guide

Dr. SENDURU SRINIVASALU, M.E., PH.D.,

Head of the Department

Dr. R. SUBHASHINI M.E., PH.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Abirami M (38120005)** and **V Sarah Philomin (38120077)** hereby declare that the Project Report entitled “**A CONTEMPORARY CLOUD BASED DYNAMIC AUTHENTICATION SYSTEM FOR MOBILE APPLICATIONS**” done by us under the guidance of **Dr. SENDURU SRINIVASALU, M.E., Ph.D.**, is submitted in partial fulfillment of the requirements for the award of **Bachelor of Technology degree in Information Technology**.

DATE:

PLACE: CHENNAI

SIGNATURE OF CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **Sathyabama** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey our thanks to **Dr. T. SASIKALA M.E., Ph.D.**, Dean, School of Computing and **Dr. R. SUBHASHINI M.E., Ph.D.**, Head of the Department, Department of Information Technology for providing us the necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. SENDURU SRINIVASALU, M.E., Ph.D.**, for his valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all teaching and non-teaching staff members of the **Department of Information Technology** who was helpful in many ways for the completion of the project.

ABSTRACT

This work enhances traditional authentication systems based on Personal Identification Numbers (PIN) and One- Time Passwords (OTP) through the incorporation of biometric information as a second level of user authentication. In our proposed approach, users draw each digit of the password on the touchscreen of the device instead of typing them as usual. A complete analysis of our proposed biometric system is carried out regarding the discriminative power of each handwritten digit and the robustness when increasing the length of the password and the number of enrolment samples. The new e-BioDigit database, which comprises on-line handwritten digits from 0 to 9, has been acquired using the finger as input on a mobile device. This database is used in the experiments reported in this work and it is available together with benchmark results in GitHub¹. Finally, we discuss specific details for the deployment of our proposed approach on current PIN and OTP systems, achieving results with Equal Error Rates (EERs) ca. 4.0% when the attacker knows the password. These results encourage the deployment of our proposed approach in comparison to traditional PIN and OTP systems where the attack would have 100% success rate under the same impostor scenario.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	i
	LIST OF FIGURES	iv
	LIST OF ABBREVIATIONS	v
1.	INTRODUCTION	1
1.1.	EXISTING SYSTEM	1
1.2.	PROBLEM DEFINITION	2
1.3.	PROPOSED SYSTEM	2
1.4.	ADVANTAGES	2
2.	REQUIREMENTS SPECIFICATION	3
2.1.	INTRODUCTION	3
2.2.	HARDWARE AND SOFTWARE SPECIFICATION	4
2.3.	TECHNOLOGIES USED	5
2.3.1.	JAVA	5
2.3.2.	AOACHE TOMCAT SERVER	9
2.3.3.	ANDROID INTRODUCTION	10
2.3.4.	LINUX KERNELS	11
2.3.5.	LIBRARIES	11
2.3.6.	ANDROID RUNTIME	12
2.3.7.	APPLICATION FRAMEWORK	12
2.3.8.	APPLICATIONS AND ADVANTAGES	12
3.	PROJECT PURPOSE AND SCOPE	13
3.1.	PROJECT SCOPE	13
3.2.	PRODUCT PERSPECTIVE	13
3.3.	SYSTEM FEATURES	14
3.4.	DESIGN AND IMPLEMENTATION CONSTRAINTS	14

3.5.	OTHER NON-FUNCTIONAL REQUIREMENTS	15
4.	SYSTEM DESIGN	16
4.1.	ARCHITECTURE DIAGRAM	16
4.2.	SEQUENCE DIAGRAM	16
4.3.	USE CASE DIAGRAM	17
4.4.	ACTIVITY DIAGRAM	17
4.5.	COLLABORATION DIAGRAM	18
4.6.	DATA FLOW DIAGRAM	18
4.7.	CLASS DIAGRAM	19
5.	METHODOLOGY	21
5.1.	MODULES	21
5.2.	MODULE EXPLANATION	21
6.	CODING AND TESTING	23
6.1.	CODING	23
6.2.	CODING STANDARDS	23
6.3.	TEST PROCEDURE	24
6.4.	TEST DATA AND OUTPUT	25
6.5.	TESTING STRATEGIES	26
7.	RESULTS AND DISCUSSION	30
8.	CONCLUSION AND FUTURE WORK	31
	REFERENCES	36
	APPENDICES	39
	A. SAMPLE CODE	39
	B. SCREENSHOTS	52
	C. PUBLICATION WITH PLAGIARISM REPORT	69

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO
2.1	SYSTEM DESIGN	25
4.1	SEQUENCE DIAGRAM	25
4.2	USE CASE DIAGRAM	26
4.3	ACTIVITY DIAGRAM	27
4.4	COLLABORATION DIAGRAM:	27
4.5	DATA FLOW DIAGRAM	28
4.6	CLASS DIAGRAM	29

LIST OF ABBREVIATIONS

S No	ABBREVIATIONS	EXPANSION
1.	JDK	JAVA DEVELOPMENT TOOLKIT
2.	DEX	DALVIK EXECUTABLES
3.	TCP	TRANSMISSION CONTROL PROTOCOL
4.	IP	INTERNET PROTOCOL
5.	HTTP	HYPER TEXT TRANSFER PROTOCOL
6.	ADT	ANDROID DEVELOPMENT TOOL

CHAPTER 1

INTRODUCTION

The main aim of this project is to develop user-friendly mobile applications ensuring data protection and high security. The rapid and continuous deployment of mobile devices around the world has been motivated not only by the high technological evolution that allows the communication and use of social media in real time, the two most prevalent user authentication approaches have been Personal Identification Numbers and One-Time Passwords. In our proposed approach, users draw each digit of the password on the touch screen of the device instead of typing them as usual. The handwritten digits can be first recognized using for example an Optical Character Recognition. After this first authentication stage, the biometric information of the handwritten digits is compared in a second authentication stage to the enrolment data of the claimed user, comparing each digit one by one.

1.1 EXISTING SYSTEM

In existing system handwritten signature is one of the most socially accepted biometrics as it has been used in financial and legal agreements for many years and it also finds applications in mobile scenarios. These approaches are based on the combination of two authentication stages. The security system checks that the claimed user introduces its unique password correctly, and its behavioral biometric information is used for an enhanced final verification. The software for capturing handwritten numerical digits was developed to minimize the variability of the user during the acquisition process. The selection of a password that is robust enough for a specific application is a key factor. The number of digits that comprise the password depends on the scenario and level of security considered in the final application. This effect has proven to be very important for many behavioral biometric traits such as the case of the handwritten signature.

1.2 PROBLEM DEFINITION:

- ✓ The amount of data requested to the user during the enrolment.
- ✓ The security level provided by the biometric system. From the point of view of the security system, it seems clear that the ideal case would be to have as much information of the user as possible.

1.3 PROPOSED SYSTEM

Our proposed system focus on providing user-friendly mobile applications ensuring data protection and high security. User should draw each digit of the password on the touch screen instead of typing them as usual. This way, the traditional authentication systems are enhanced by incorporating dynamic handwritten biometric information. Our system involves two stages of authentication the drawn pin should be like pin entered during registration process. Our second stage of authentication involves multiple options based on user preference where user can set multiple set of combinations. User can set second stage password as stroke, time, screen brightness or sensor-based authentication system. The incorporation of biometric information on traditional password-based systems can improve the security through a second level of user authentication.

1.4 ADVANTAGES:

- These approaches enable active or continuous authentication schemes, in which the user is transparently authenticated.
- Handwritten signature is one of the most socially accepted biometrics.
- The incorporation of biometric information on traditional password-based systems can improve the security through a second level of user authentication.

CHAPTER 2

REQUIREMENT SPECIFICATIONS

2.1 INTRODUCTION

Mobile devices have become an indispensable tool for most people nowadays .The rapid and continuous deployment of mobile devices around the world has been motivated not only by the high technological evolution and new features incorporated but also to the new internet infrastructures like 5G that allows the communication and use of social media in real time, among many other factors. In this way, both public and private sectors are aware of the importance of mobile devices for the society and are trying to deploy their services through user friendly mobile applications ensuring data protection and high security. Traditionally, the two most prevalent user authentication approaches have been Personal Identification Numbers (PIN) and One-Time Passwords (OTP). While PIN-based authentication systems require users to memorize their personal passwords, OTP-based systems avoid users to memorize them as the system oversees selecting and providing to the user a different password each time is required, e.g., sending messages to personal mobile devices or special tokens. Despite the high popularity and deployment of PIN- and OTP-based authentication systems in real scenarios, many studies have highlighted the weaknesses of these approaches .First, it is common to use passwords based on sequential digits, personal information such as birth dates, or simply words such as “password” or “qwerty” that are very easy to guess. Second, passwords that are typed on mobile devices such as tablets or smartphones are susceptible to “smudge attacks”, i.e., the deposition of finger grease traces on the touchscreen can be used for the impostors to guess the password .Finally, password-based authentication is also vulnerable to “shoulder surfing”. This type of attack is produced when the impostor can observe directly or use external recording devices to collect the user information. This attack has attracted the attention of many researchers in recent years due to the increased deployment of handheld recording devices and public surveillance infrastructures . Biometric recognition schemes can cope with these challenges by combining both a high level of security and convenience. This study evaluates the advantages and potential of incorporating biometrics to password-based mobile authentication systems, asking the users to draw each digit of

the password on the touchscreen instead of typing them as usual. This way, the traditional authentication systems are enhanced by incorporating dynamic handwritten biometric information. One example of use that motivates our proposed approach is on internet payments with credit cards. Banks usually send a numerical password (typically between 6 and 8 digits) to the user's mobile device. This numerical password must be inserted by the user in the security platform to complete the payment. Our proposed approach enhances such scenario by including a second authentication factor based on the user biometric information while drawing the digits. Fig. 1 shows a general architecture of our proposed password-based mobile authentication approach. The three following main modules are analyzed in this study: i) enrolment set, ii) password generation, and iii) touch biometric system. Depending on the final application (i.e., PIN or OTP), the handwritten digits can be first recognized using for example an Optical Character Recognition (OCR) system to verify the authenticity of the password. After this first authentication stage, the biometric information of the handwritten digits is compared in a second authentication stage to the enrolment data of the claimed user, comparing each digit one by one. In this study we focus on the second authentication stage based on the behavioral information of the user while performing the handwritten digits as the recognition of numerical digits has already shown to be an almost solved problem with errors close to 0% [8], [9]. Therefore, in this study we assume that impostors pass the first stage of the security system (i.e., they know the password of the user to attack) and thus, the attack would have 100% success rate if our proposed approach was not present.

2.2 HARDWARE AND SOFTWARE SPECIFICATION

HARDWARE REQUIREMENTS

- Hard Disk : 500GB and Above
- RAM : 4 GB and Above
- Processor : I3 and Above

SOFTWARE REQUIREMENTS

- Windows 7 and above
- JDK 1.7

- Android Studio 3.4
- Android Phone

2.3 TECHNOLOGIES USED

- Core java, Android, PHP

2.3.1 JAVA

Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. The language, initially called Oak (named after the oak trees outside Gosling's office), was intended to replace C++, although the feature set better resembles that of Objective C.

The language that the team created was originally called Oak. The first implementation of Oak was in a PDA-type device called Star Seven (*7) that consisted of the Oak language, an operating system called GreenOS, a user interface, and hardware. The name *7 was derived from the telephone sequence that was used in the team's office and that was dialed to answer any ringing telephone from any other phone in the office.

Around the time the First-Person project was floundering in consumer electronics, a new craze was gaining momentum in America; the craze was called "Web surfing." The World Wide Web, a name applied to the Internet's millions of linked HTML documents was suddenly becoming popular for use by the masses. The reason for this was the introduction of a graphical Web browser called Mosaic, developed by ncSA. The browser simplified Web browsing by combining text and graphics into a single interface to eliminate the need for users to learn many confusing UNIX and DOS commands. Navigating around the Web was much easier using Mosaic.

It has only been since 1994 that Oak technology has been applied to the Web. In 1994, two Sun developers created the first version of Hot Java, and then called Web Runner, which is a graphical browser for the Web that exists today. The browser was coded entirely in the Oak language, by this time called Java. Soon after, the Java compiler was rewritten in the Java language from its original C code, thus proving that Java could be used effectively as an application language. Sun introduced Java in May 1995 at the Sun World 95 convention. Web surfing has become an enormously popular practice among millions of computer users. Until Java, however, the content of information on the Internet has been a bland series of HTML documents. Web users are hungry for applications that are interactive, that users can execute no matter what hardware or

software platform they are using, and that travel across heterogeneous networks and do not spread viruses to their computers. Java can create such applications.

WORKING OF JAVA

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods). When the interpreter executes a class, it looks for a particular method by the name of **main**, which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (like the `argv []` of C) and is declared as a static method. To output text from the program, we execute the **println** method of **System.out**, which is java's output stream. UNIX users will appreciate the theory behind such a stream, as it is standard output. For those who are instead used to the Wintel platform, it will write the string passed to it to the user's program.

Java consists of two things :

- Programming language
- Platform

THE JAVA PROGRAMMING LANGUAGE

Java is a high-level programming language that is all the following:

- Simple
- Object-oriented
- Distributed
- Interpreted
- Robust
- Secure
- Architecture-neutral
- Portable
- High-performance
- Multithreaded

➤ Dynamic

The code can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows, Java is unusual in that each Java program is both compiled and interpreted. With a compiler, you translate a Java program into an intermediate language called **Java byte codes** – the platform independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction is parsed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. This figure illustrates how it works:

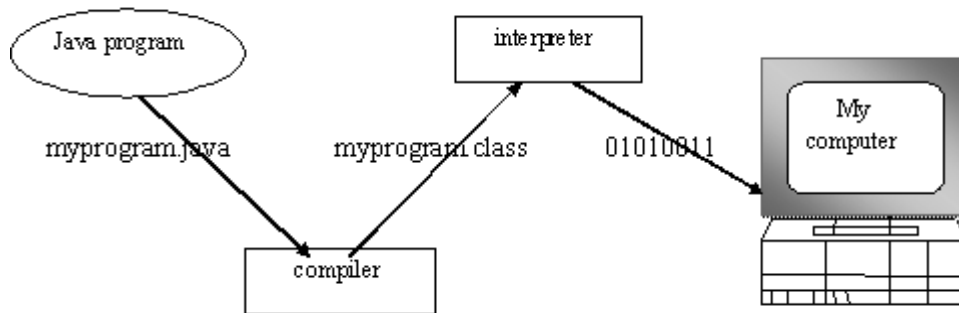
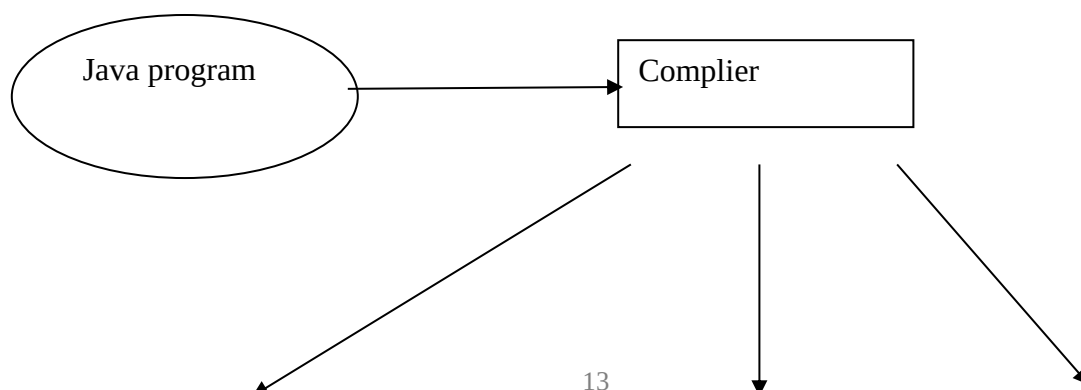
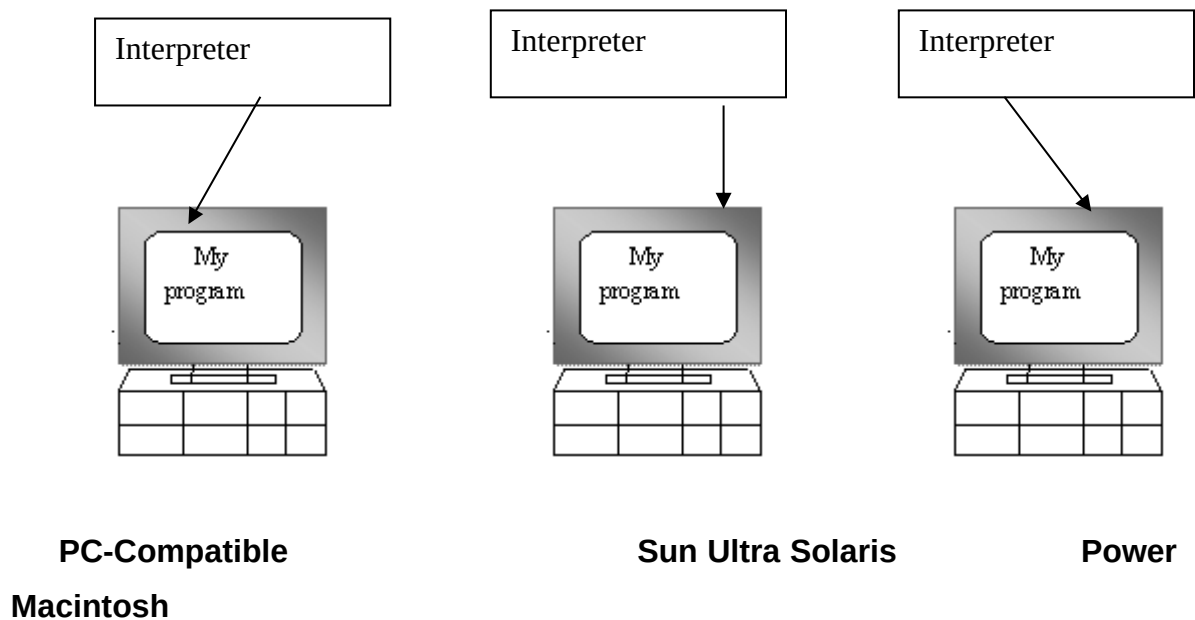


Fig.2.1

You can think of Java byte codes as the machine code instructions for the **Java Virtual Machine (JVM)**. Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of JVM. That JVM can also be implemented in hardware. Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM. For example, that same Java program can be run on Windows NT, Solaris and Macintosh





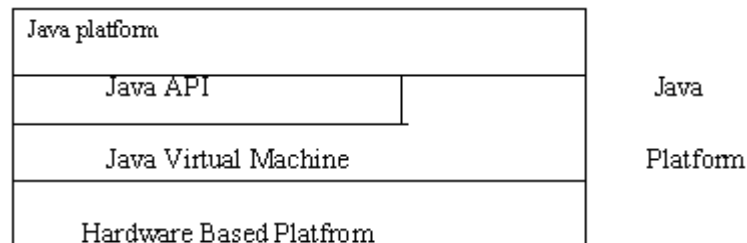
THE JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components :

- The Java Virtual Machine (JVM)
- The Java Application Programming Interface (Java API)

You've already been introduced to the JVM. It's the base for the Java platform and is ported onto various hardware-based platforms. The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries **(packages)** of related components. The following figure depicts a Java program, such as an application or applet, that's running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware



dependencies. As a platform-independent environment, Java can be a bit slower than native code. However, smart compilers, wheel-tuned interpreters, and just-in-time byte compilers can bring Java's performance close to that of native code without threatening portability.

2.3.2 APACHE TOMCAT SERVER

Apache Tomcat (formerly under the Apache Jakarta Project; Tomcat is now a top-level project) is a web container developed at the Apache Software Foundation. Tomcat implements the servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are normally XML-formatted. Because Tomcat includes its own HTTP server internally, it is also considered a standalone web server.

Environment

Tomcat is a web server that supports servlets and JSPs. Tomcat comes with the Jasper compiler that compiles JSPs into servlets. The Tomcat servlet engine is often used in combination with an Apache web server or other web servers. Tomcat can also function as an independent web server. Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other

environments with minimal requirements for speed and transaction handling. However, that perception no longer exists; Tomcat is increasingly used as a standalone web server in high-traffic, high-availability environments. Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM.

Product features

Tomcat 3.x (initial release)

- Implements The Servlet 2.2 And JSP 1.1 Specifications
- Servlet Reloading
- Basic HTTP Functionality Tomcat 4.X
- Implements The Servlet 2.3 And JSP 1.2 Specifications
- Servlet Container Redesigned As Catalina
- JSP Engine Redesigned As Jasper
- Coyote Connector
- Java Management Extensions (JMX), JSP and Struts-based administration
- Tomcat 5.x
- implements the Servlet 2.4 and JSP 2.0 specifications
- reduced garbage collection, improved performance and scalability
- native Windows and Unix wrappers for platform integration
- faster JSP parsing

History

Tomcat started off as a servlet specification implementation by James Duncan Davidson, a software architect at Sun. He later helped make the project open source and played a key role in its donation by Sun to the Apache Software Foundation. Davidson had initially hoped that the project would become open-sourced and, since most open-source projects had O'Reilly books associated with them featuring an animal on the cover, he wanted to name the project after an animal. He came up with Tomcat since he reasoned the animal represented something that could take care of and fend for itself. His wish to see an animal cover eventually came true when O'Reilly published their Tomcat book with a tomcat on the cover.

2.3.3 Android Introduction:

Android is a Linux based operating system it is designed primarily for touch screen mobile devices such as smart phones and tablet computers. The operating system has developed a lot in last 15 years starting from black and white phones to recent smart phones or minicomputers. One of the most widely used mobile OS these days is android. The android is software that was founded in Palo Alto of California in 2003. The android is a powerful operating system, and it supports large number of applications in Smartphones. These applications are more comfortable and advanced for the users. The hardware that supports android software is based on ARM architecture platform. The android is an open-source operating system means that it's free and any one can use it. The android has got millions of apps available that can help you managing your life one or other way and it is available low cost in market at that reasons android is very popular. The android development supports with the full java programming language. Even other packages that are API and JSE are not supported. The first version 1.0 of android development kit (SDK) was released in 2008 and latest updated version is jellybean. The android is an operating system and is a stack of software components which is divided into five sections and four main layers that is



Fig 2.1

2.3.4. Linux kernel:

The android uses the powerful Linux kernel, and it supports wide range of hardware drivers. The kernel is the heart of the operating system that manages input and output requests from software. This provides basic system functionalities like process management, memory management, device management like camera, keypad, display etc. the kernel handles all the things. The Linux is good at networking, and it is not necessary to interface it to the peripheral hardware. The kernel itself does not interact directly with the user but rather interacts with the shell and other programs as well as with the hardware devices on the system.

2.3.5. Libraries:

The on top of a Linux kennel there is a set of libraries including open-source web browser such as web kit, library libc. These libraries are used to play and record audio

and video. The SQLite is a data base which is useful for storage and sharing of application data. The SSL libraries are responsible for internet security etc.

2.3.6. Android Runtime:

The android runtime provides a key component called Dalvik Virtual Machine which is a kind of java virtual machine. It is specially designed and optimized for android. The Dalvik VM is the process virtual machine in the android operating system. It is software that runs apps on android devices. The Dalvik VM makes use of Linux core features like memory management and multithreading which is in a java language. The Dalvik VM enables every android application to run its own process. The Dalvik VM executes the files in the .dex format.

2.3.7. Application framework:

The application framework layer provides many higher-level services to applications such as windows manager, view system, package manager, resource manager etc. The application developers are allowed to make use of these services in their application.

2.3.8. Applications:

You will find all the android applications at the top layer and you will write your application and install on this layer. Examples of such applications are contacts, books, browsers, services etc. Each application performs a different role in the overall applications.

Advantages:

- Android is Linux based open-source operating system ; it can be developed by any one
- Easy access to the android apps
- You can replace the battery and mass storage, disk drive and UDB option
- It supports all Google services
- The operating system can inform you of a new SMS and Emails or latest updates.

- It supports Multitasking
- Android phone can also function as a router to share internet
- Its free to customize
- Can install a modified ROM
- Its supports 2D and 3D graphics

CHAPTER 3

PROJECT PURPOSE AND SCOPE

The main aim of this paper is to secure a mobile application with Password drawing pattern with various ways of implementation.

3.1 Project Scope

Our proposed system focus on providing user-friendly mobile applications ensuring data protection and high security. User should draw each digit of the password on the touch screen instead of typing them as usual. This way, the traditional authentication systems are enhanced by incorporating dynamic handwritten biometric information. Our system involves two stages of authentication the drawn pin should be like pin entered during registration process. Our second stage of authentication involves multiple options based on user preference where user can set multiple set of combinations. User can set second stage password as stroke, time, screen brightness or sensor-based authentication system.

3.2 Product Perspective

In this work we evaluate the potential of touch biometric verification systems based on time functions . Signals captured by the digitizer (i.e., X and Y spatial coordinates) are used to extract a set of 21-time functions for each numerical digit sample . Information related to pressure, pen angular orientations or pen ups broadly used in other biometric traits such as handwriting and handwritten signature is not considered here as this information is not available in all mobile devices when using the finger touch as input. Sequential Forward Floating Search (SFFS) algorithm is used for the DTW algorithm in some of the experiments to select the best subsets of time functions for each handwritten digit and improve the system performance in terms of EER (%). So with the help of gesture what we have drawn in the mobile , all the numbers patters will be stored in the backend. So whenever a user signin to the ecommerce application , this password security mechanism will be enacting with every product purchase.

3.3 System Features

Here we concentrate on password mechanism with user drawing patterns. The user need to register their pin number in the application with proper strokes. All the pin numbers with proper strokes will be getting stored in database server.

We implement random changes in screen brightness for securing the user's confidential passwords from hackers. We use ultrasound for OTP transfer , so that OTP will be sent to a host mobile from there it send that data through sonic waves to the user's mobile. Thus security mechanism for user drawing patterns in android application.

3.4 Design and Implementation Constraints

Constraints in Analysis

- ◆ Constraints as Informal Text
- ◆ Constraints as Operational Restrictions
- ◆ Constraints Integrated in Existing Model Concepts
- ◆ Constraints as a Separate Concept
- ◆ Constraints Implied by the Model Structure

Constraints in Design

- ◆ Determination of the Involved Classes
- ◆ Determination of the Involved Objects
- ◆ Determination of the Involved Actions
- ◆ Determination of the Require Clauses
- ◆ Global actions and Constraint Realization

Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality

associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object-oriented methods.

3.5 Other Nonfunctional Requirements

Performance Requirements

The application at this side controls and communicates with the following three main general components.

- embedded browser in charge of the navigation and accessing to the web service;
- Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.

Web Server, Security Module, Server-Side Capturing Engine, Preprocessing Engine, Database System, Verification Engine, Output Module.

Safety Requirements

1. The software may be safety critical. If so, there are issues associated with its integrity level
2. The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions.
3. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
4. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
5. If a computer system is to run software of a high integrity level, then that system should not at the same time accommodate software of a lower integrity level.
6. Systems with different requirements for safety levels must be separated.
7. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

CHAPTER 4 SYSTEM DESIGN

4.1 ARCHITECTURE DIAGRAM:

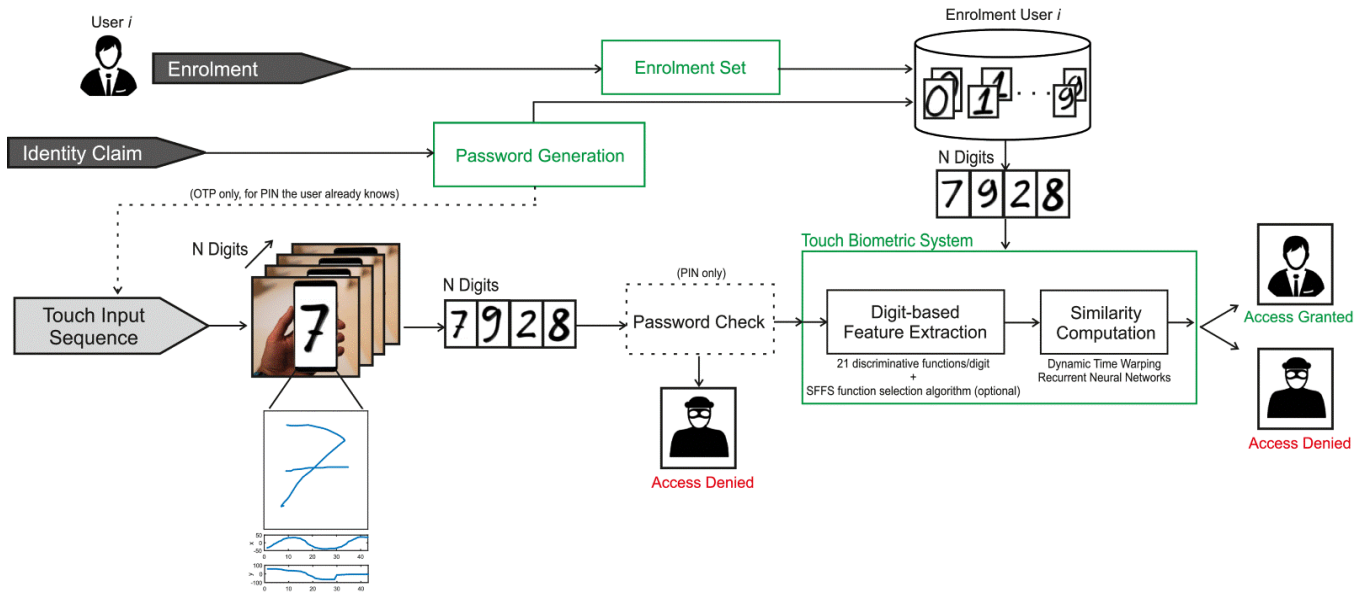


Fig: 4.1

4.2 Sequence Diagram:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

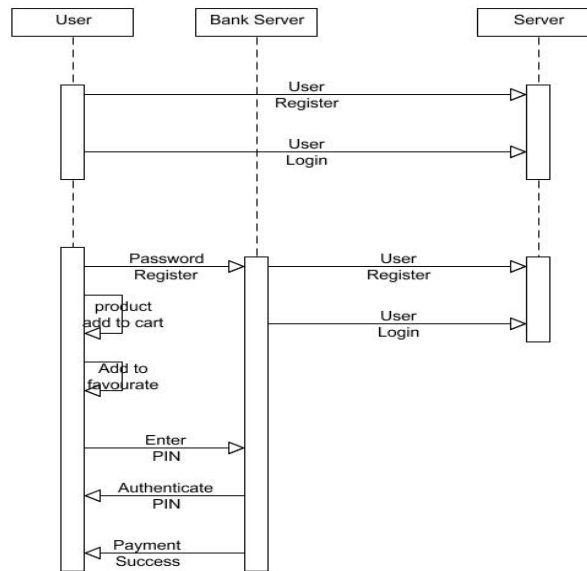


Fig: 4.2

4.3 Use Case Diagram:

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals, and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

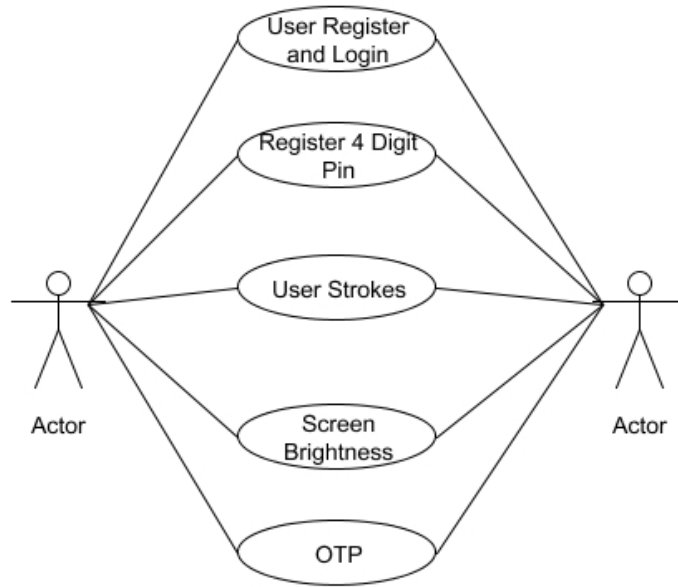


Fig: 4.3

4.4 Activity Diagram:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

- Rounded rectangles represent activities.
- Diamonds represent decisions.
- Bars represent the start or end of concurrent activities.
- A black circle represents the start of the workflow.
- An encircled circle represents the end of the workflow.

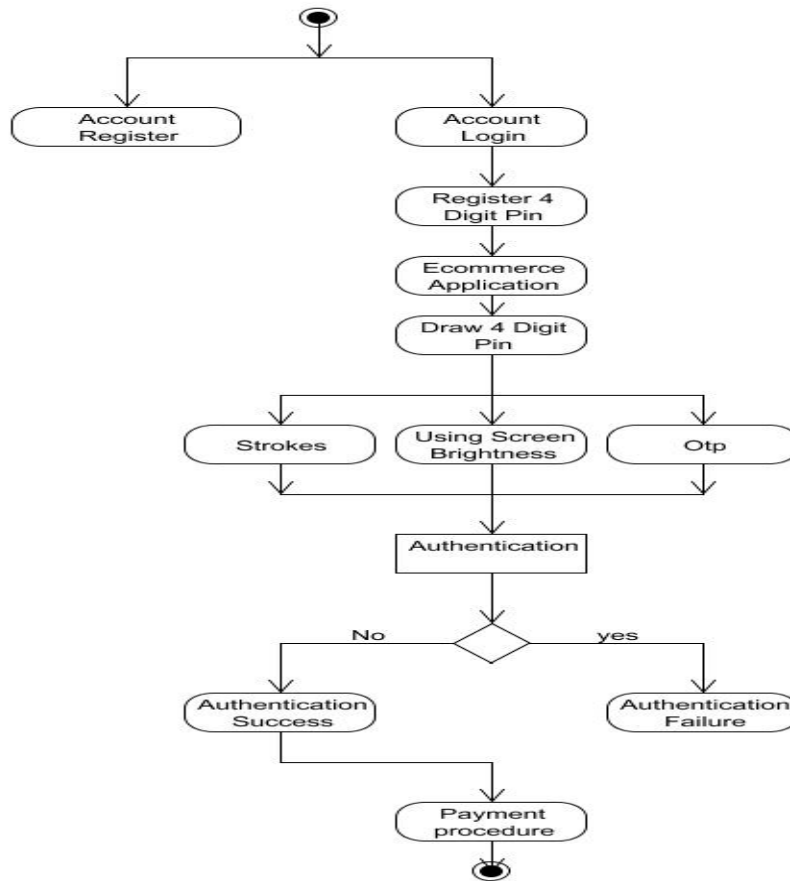


Fig: 4.4

4.5 COLLABORATION DIAGRAM:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

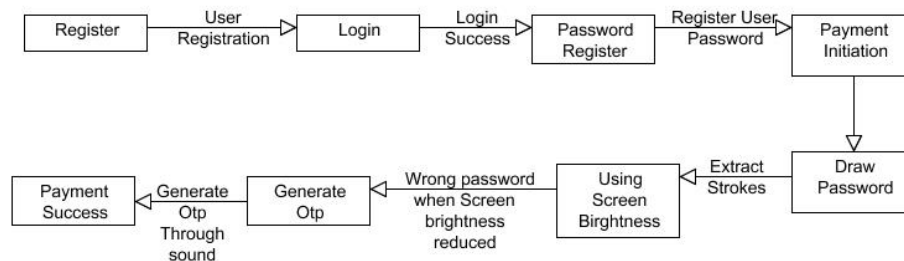
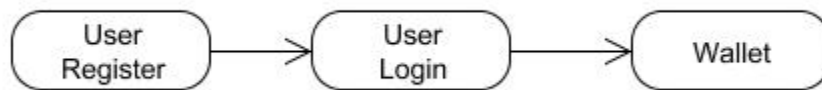


Fig: 4.5

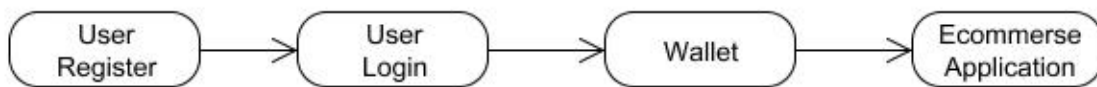
4.6 DATA FLOW DIAGRAM:

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

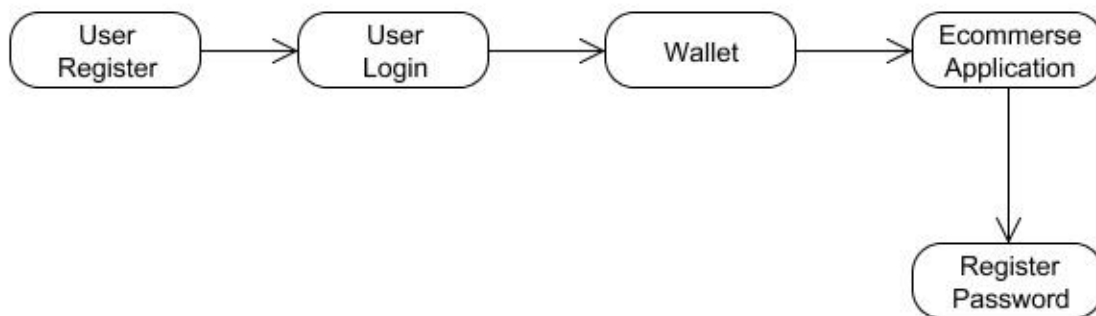
Level 0:



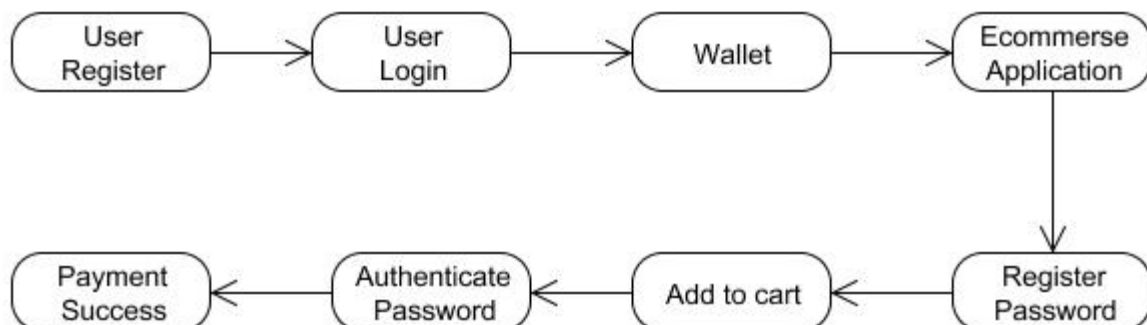
Level 1:



Level 2:



Level 3:



4.7 Class Diagram

A Class diagram in the Unified Modeling Language 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.

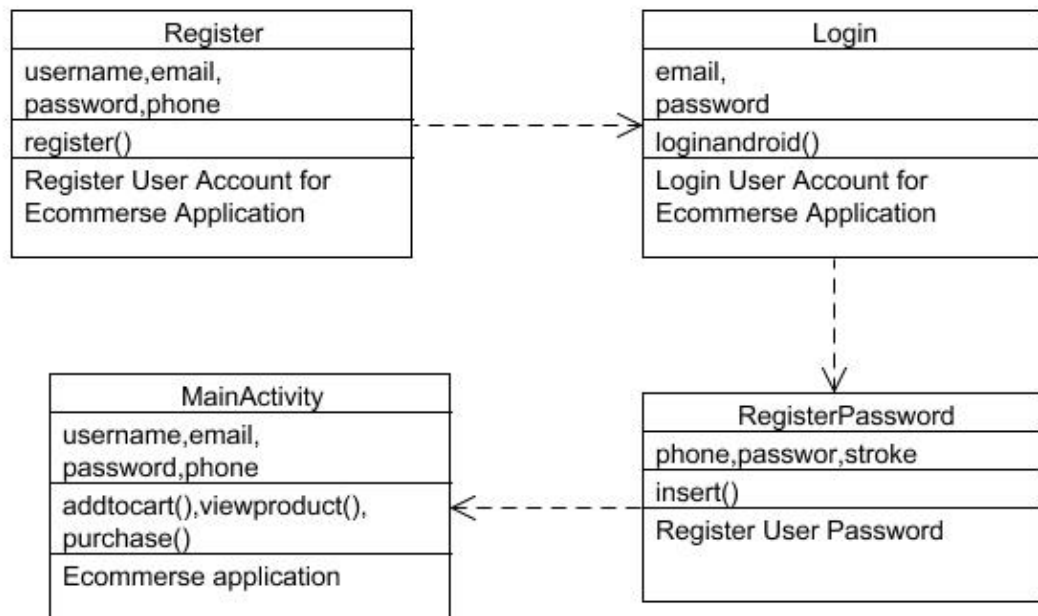


Fig: 4.6

CHAPTER 5

METHODOLOGY

User should draw each digit of the password on the touch screen instead of typing them as usual. This way, the traditional authentication systems are enhanced by incorporating dynamic handwritten biometric information. Our system involves two stages of authentication the drawn pin should be like pin entered during registration process. Our second stage of authentication involves multiple options based on user preference where user can set multiple set of combinations. User can set second stage password as stroke, time, screen brightness or sensor-based authentication system. The incorporation of biometric information on traditional password-based systems can improve the security through a second level of user authentication.

5.1. MODULES

- User Authentication and Ecommerce View Product
- Cart and Payment Using Biometric Handwritten Password
- Biometric Password Using Strokes
- Biometric Password Using Screen Brightness and Time

5.2. MODULES DESCRIPTION:

- User Authentication and Ecommerce View Product

User has an initial level Registration Process. The users provide their own personal information for this process. The server in turn stores the information in its database and user can view a list of products in their page multiple list of products and their details.

- Cart and Payment Using Biometric Handwritten Password

User can select a list of product they wish to purchase the selected product will be listed in a cart page and user can initiate general purchase information has to be filled. Completing general detail user must draw their four-digit pin one by one on screen. The drawn password then converted into an image through optical character recognition numbers from each image fetched and verified with user password.

- Biometric Password Using Strokes

User must register their four-digit password with multiple strokes during their registration process once the process completed during confirm password .User must confirm their password with same password with stroke has to be verified. Strokes for each drawn digits should match with strokes given at time of registration.

- Biometric Password Using Screen Brightness and Time

Spyware attack will be avoided by proposing the idea that uses the screen brightness as an authentication tool. The android secure environment generates the 6-digit binary value. Based on the binary digit the brightness of the screen gets changed to high or low. If the screen brightness is high the user should input the correct PIN digit. Else the user should give the wrong and random PIN number.

The system will remove the digits which inserted while the screen brightness is low and apply the HMac algorithm for the PIN given by user and generate the Signature for the user PIN which is a digestible Value to avoid MAN-IN-MIDDLE attack. The server gets the signature of user generated PIN and generates the signature value for the Original PIN and compare two signatures. If the two Signatures are equal the user can access the Profile of the user. If not, user can't access the profile.

CHAPTER 6

CODING AND TESTING

6.1 CODING

Once the design aspect of the system is finalizing the system enters the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in each programming language. Therefore, a good coding style must be taken whenever changes are required it easily screwed into the system.

6.2 CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

- Program should be simple, clear, and easy to understand.
- Naming conventions
- Value conventions
- Script and comment procedure
- Message box format
- Exception and error handling

NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

Class names

Class names are problem domain equivalence and begin with capital letter and have mixed cases.

Member Function and Data Member name

Member function and data member name begins with a lowercase letter with each subsequent letters of the new words in uppercase and the rest of letters in lowercase.

VALUE CONVENTIONS

Value conventions ensure values for variable at any point of time. This involves the following:

- Proper default values for the variables.
- Proper validation of values in the field.
- Proper documentation of flag values.

SCRIPT WRITING AND COMMENTING STANDARD

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

MESSAGE BOX FORMAT

When something must be prompted to the user, he must be able to understand it properly. To achieve this, a specific format has been adopted in displaying messages to the user. They are as follows:

- X – User has performed illegal operation.
- ! – Information to the user.

6.3 TEST PROCEDURE

SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately

and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

6.4 TEST DATA AND OUTPUT

UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white box testing techniques were heavily employed for unit testing.

FUNCTIONAL TESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

- Performance Test
- Stress Test
- Structure Test

PERFORMANCE TEST

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

STRESS TEST

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the way a programmer in which a program unit breaks.

STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been exercised at least once.

- Exercise all logical decisions on their true or false sides.
- Execute all loops at their boundaries and within their operational bounds.
- Exercise internal data structures to assure their validity.
- Checking attributes for their correctness.
- Handling end of file condition, I/O errors, buffer problems and textual errors in output information

INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules

integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

6.5 TESTING TECHNIQUES / TESTING STRATEGIES

The software testing process commences once the program is created, and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding a yet undiscovered error. A successful test is one that uncovers a yet undiscovered error. Any engineering product can be tested in one of the two ways:

WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been designed to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

- Flow graph notation
- Cyclometric complexity
- Deriving test cases
- Graph matrices Control

BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

- Graph based testing methods
- Equivalence partitioning
- Boundary value analysis
- Comparison testing

SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

- Testing begins at the module level and works “outward” toward the integration of the entire computer-based system.
- Different testing techniques are appropriate at different points in time.
- The developer of the software and an independent test group conducts testing.
- Testing and Debugging are different activities, but debugging must be accommodated in any testing strategy.

INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when we put them together. The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

PROGRAM TESTING:

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that in violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax error. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data

fields, out-of-range items, and invalid combinations. Since the compiler will not deduct logical error, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or on arithmetic expression. Condition testing method focuses on testing each condition in the program the purpose of condition test is to deduct not only errors in the condition of a program but also other an errors in the program.

SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built into a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attack must also be tested for invulnerability from rear attack. During security, the tester places the role of individual who desires to penetrate system.

VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditions exists.

- The function or performance characteristics confirm to specifications and are accepted.
- A validation from specification is uncovered and a deficiency created.

Deviation or errors discovered at this step in this project is corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic

USER ACCEPTANCE TESTING

User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

- Input screen design.
- Output screen design.

CHAPTER 7

RESULTS AND DISCUSSION

The concept is used to identify the numbers drawn and is then sent to the process of HMAC and HMAC (hash-based message authentication code) – secure computing concept is used to convert the pin entered where a hash code which is later clubbed into one and stored in our database.

AWS implementation can be done in either way,

1. Client-side authentication where the authentication service can be requested to any authorization to add them as an additional protection service
2. Backend management where the AWS could host the service and several other organizations could share it to their customers and they can directly access from our instance.

OCR (optical character recognition) – machine learning concept is used to identify the numbers drawn and is then sent to the process of HMAC and HMAC (hash-based message authentication code) – secure computing concept is used to convert the pin entered where a hash code which is later clubbed into one and stored in our database.

The new e-Bio Digit edifying arrangement, which recalls for line made by hand digits from 0 to 9, has been procured recollecting the finger as liability regarding a telephone. This data base is used in the starters reported in this work and it is available nearby benchmark achieves GitHub1. Finally, we talk about unequivocal nuances for the relationship of our proposed approach on current PIN and OTP structures, achieving results with Equal Error Rates (EERs) ca. 4.0% when the aggressor knows the mysterious enunciation. These results support the sending of our proposed approach on the other hand, with standard PIN and OTP systems where the attack would have 100% accomplishment rate under a comparable stunt circumstance

CHAPTER 8

CONCLUSION AND FUTURE WORK

This work assesses the savvy method for verifying the informal communication accounts having a place with them by utilizing the screen brilliance of android mobiles to keep away from the spyware assault, shoulder riding assault, and man in the center assault and to have them in virtual server which would be gainful to individuals anyplace around the globe. To ad lib the equivalent and distribute it in motivation behind each organization administration could add equivalent to a secret phrase system and we can add it to our essential applications which are to be secured. The advantages and plausibility of bringing written by hand contact biometrics into secret word-based portable confirmation frameworks are inspected in this review. The examinations point by point in this paper utilized the new e-Bio Digit information base, which contains manually written mathematical digits from zero to nine, and it is given together benchmark results.

REFERENCES

- [1] M. Salehan and A. Negahban, "Social Networking on Smartphones: When Mobile Phones Become Addictive," *Computers in Human Behavior*, vol. 29, no. 6, pp. 2632–2639, 2013.
- [2] J. Bonneau, C. Herley, P. Oorschot, and F. Stajano, "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes," in *Proc. IEEE Symposium on Security and Privacy*, 2012, pp. 553–567.
- [3] J. Galbally, I. Coisel, and I. Sanchez, "A New Multimodal Approach for Password Strength Estimation Part I: Theory and Algorithms," *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 2829–2844, 2017.
- [4] A. Aviv, K. Gibson, E. Mossop, M. Blaze, and J. Smith, "Smudge Attacks on Smartphone Touch Screens," in *Proc. of the 4th USENIX Conference on Offensive Technologies*, 2010, pp. 1–7.
- [5] D. Shukla, R. Kumar, A. Serwadda, and V. Phoha, "Beware, Your Hands Reveal Your Secrets!" in *Proc. of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.
- [6] Q. Yue, Z. Ling, X. Fu, B. Liu, W. Yu, and W. Zhao, "My Google Glass Sees Your Passwords!" in *Black Hat USA*, 2014.
- [7] W. Meng, D. Wong, S. Furnell, and J. Zhou, "Surveying the Development of Biometric User Authentication on Mobile Phones," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1268–1293, 2015.
- [8] L. Wan, M. Zeiler, S. Zhang, Y. LeCun, and R. Fergus, "Regularization of Neural Networks using DropConnect," in *Proc. of the 30th International Conference on Machine Learning*, 2013, pp. 1058–1066.

- [9] M. Liang and X. Hu, "Recurrent Convolutional Neural Network for Object Recognition," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3367–3375.
- [10] J. Angulo and E. Wastlund, "Exploring Touch-Screen Biometrics for User Identification on Smart Phones," J. Camenisch, B. Crispo, S. Fischer-Hbner, R. Leenes, G. Russello (Eds.), Privacy and Identity Management for Life, Springer, pp. 130–143, 2011.
- [11] P. Lacharme and C. Rosenberger, "Synchronous One Time Biometrics With Pattern Based Authentication," in Proc. 11th Int. Conf. on Availability, Reliability and Security, ARES, 2016.
- [12] E. von Zezschwitz, M. Eiband, D. Buschek, S. Oberhuber, A. D. Luca, F. Alt, and H. Hussmann, "On Quantifying the Effective Password Space of Grid-based Unlock Gestures," in Proc. of the International Conference on Mobile and Ubiquitous Multimedia, 2016, pp. 201–212.
- [13] D. Buschek, A. D. Luca, and F. Alt, "There is more to Typing than Speed: Expressive Mobile Touch Keyboards via Dynamic Font Personalisation," in Proc. of the International Conference on Human- Computer Interaction with Mobile Devices and Services, 2015, pp. 125– 130.
- [14] —, "Improving Accuracy, Applicability and Usability of Keystroke Biometrics on Mobile Touchscreen Devices," in Proc. of the CHI Conference on Human Factors in Computing Systems, 2015, pp. 1393–
- [15] L. Li, X. Zhao, and G. Xue, "Unobservable Reauthentication for Smartphones," in Proc. 20th Network and Distributed System Security Symposium, NDSS, 2013.

- [16] N. Sae-Bae, N. Memon, K. Isbister, and K. Ahmed, "Multitouch Gesture-Based Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 568–582, 2014.
- [17] C. Shen, Y. Zhang, X. Guan, and R. Maxion, "Performance Analysis of Touch-Interaction Behavior for Active Smartphone Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 3, pp. 498–513, 2016.
- [18] J. Fierrez, A. Pozo, M. Martinez-Diaz, J. Galbally, and A. Morales, "Benchmarking Touchscreen Biometrics for Mobile Authentication," *IEEE Trans. on Information Forensics and Security*, vol. 13, 2018.
- [19] N. Sae-Bae and N. Memon, "Online Signature Verification on Mobile Devices," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 933–947, 2014.
- [20] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega- Garcia, "Benchmarking Desktop and Mobile Handwriting across COTS Devices: the e-BioSign Biometric Database," *PLOS ONE*, 2017.
- [21] W. Khan, M. Aalsalem, and Y. Xiang, "A Graphical Password Based System for Small Mobile Devices," *International Journal of Computer Science*, vol. 5, no. 2, pp. 145–154, 2011.
- [22] M. Martinez-Diaz, J. Fierrez, and J. Galbally, "Graphical Passwordbased User Authentication with Free-Form Doodles," *IEEE Trans. On Human-Machine Systems*, vol. 46, no. 4, pp. 607–614, 2016.
- [23] T. Kutzner, F. Ye, I. Bonninger, C. Travieso, M. Dutta, and A. Singh, "User Verification Using Safe Handwritten Passwords on Smartphones," in *Proc. 8th International Conference on Contemporary Computing, IC3*, 2015.

[24] T. Nguyen, N. Sae-Bae, and N. Memon, "DRAW-A-PIN: Authentication using finger-drawn PIN on touch devices," *Computers and Security*, vol. 66, pp. 115–128, 2017.

APPENDICES

A. SOURCE CODE:

IpAddress.java

```
package com.example.handwrittenpassword;

public class IpAddress {
    public static String Ip_Address;
}
```

Login.java

```
import android.content.Intent;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.example.handwrittenpassword.startup.MainActivity;
import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;

import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class Login extends AppCompatActivity {
    public EditText email,password;
    public Button login;
    public TextView register;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        email = findViewById(R.id.etLogGmail);
        password = findViewById(R.id.etLoginPassword);

        login = findViewById(R.id.btnLogin);
```

```

register = findViewById(R.id.tvRegister);

register.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent = new Intent(getApplicationContext(),Register.class);
        startActivity(intent);
        finish();
    }
});

login.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        loginandroid();
    }
});
}

public void loginandroid(){
    final String username = email.getText().toString();
    final String spassword = password.getText().toString();

    //validating inputs
    if (TextUtils.isEmpty(username)) {
        email.setError("Please enter your email");
        email.requestFocus();
        return;
    }

    if (TextUtils.isEmpty(spassword)) {
        password.setError("Please enter your password");
        password.requestFocus();
        return;
    }

    //if everything is fine
    StringRequest = new StringRequest(Request.Method.POST, URLs.URL_LOGIN,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                // progressBar.setVisibility(View.GONE);

                try {
                    //converting response to json object
                    JSONObject obj = new JSONObject(response);

                    //if no error in response
                    if (!obj.getBoolean("error")) {
                        Toast.makeText(getApplicationContext(), obj.getString("message"),
                            Toast.LENGTH_SHORT).show();

                        //getting the user from the response
                        JSONObject userJson = obj.getJSONObject("user");
                        SharedPreferences = getSharedPreferences("mypref",MODE_PRIVATE);

```

```

        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString("phone",userJson.getString("phone"));
        editor.putString("email",userJson.getString("email"));
        Log.e("phone",userJson.getString("phone"));
        editor.apply();
        //creating a new user object
        /* User user = new User(
            userJson.getInt("id"),
            userJson.getString("username"),
            userJson.getString("email"),
            userJson.getString("gender")
        );

        //storing the user in shared preferences
        SharedPrefManager.getInstance(getApplicationContext()).userLogin(user);

        //starting the profile activity*/
        finish();
        startActivity(new Intent(getApplicationContext(), MainActivity.class));
    } else {
        Toast.makeText(getApplicationContext(), obj.getString("message"),
Toast.LENGTH_SHORT).show();
    }
    } catch (JSONException e) {
        e.printStackTrace();
    }
    }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(), error.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    }) {
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String, String> params = new HashMap<>();
            params.put("username", username);
            params.put("password", spassword);
            return params;
        }
    }
};

VolleySingleton.getInstance(this).addToRequestQueue(stringRequest);

}

}

```

Register.java

```

package com.example.handwrittenpassword;

import android.content.Intent;

```

```

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;

import org.json.JSONException;
import org.json.JSONObject;

import java.util.HashMap;
import java.util.Map;

public class Register extends AppCompatActivity {
    public EditText name,phone,email,password;
    public Button register,login;
    // public TextView login;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        name = findViewById(R.id.etRegName);
        phone = findViewById(R.id.etRegPhone);
        email = findViewById(R.id.etRegGmail);
        password = findViewById(R.id.etRegPassword);

        register = findViewById(R.id.btnRegLogin);
        login = findViewById(R.id.btnGotoLogin);

        register.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                register();
            }
        });

        login.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent = new Intent(getApplicationContext(),Login.class);
                startActivity(intent);
                finish();
            }
        });
    }
}

```

```

public void register(){
    final String susername = name.getText().toString().trim();
    final String semail = email.getText().toString().trim();
    final String spassword = password.getText().toString().trim();
    final String sphone = phone.getText().toString().trim();

    //final String phone = ((RadioButton)
findViewById(radioGroupGender.getCheckedRadioButtonId())).getText().toString();

    //first we will do the validations

    if (TextUtils.isEmpty(susername)) {
        name.setError("Please enter username");
        name.requestFocus();
        return;
    }

    if (TextUtils.isEmpty(semail)) {
        email.setError("Please enter your email");
        email.requestFocus();
        return;
    }
    if (TextUtils.isEmpty(sphone)) {
        phone.setError("Please enter your email");
        phone.requestFocus();
        return;
    }

    if (!android.util.Patterns.EMAIL_ADDRESS.matcher(semail).matches()) {
        email.setError("Enter a valid email");
        email.requestFocus();
        return;
    }

    if (!Patterns.PHONE.matcher(sphone).matches()) {
        phone.setError("Enter a valid email");
        phone.requestFocus();
        return;
    }

    if (TextUtils.isEmpty(spassword)) {
        password.setError("Enter a password");
        password.requestFocus();
        return;
    }

    StringRequest = new StringRequest(Request.Method.POST, URLs.URL_REGISTER,
        new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                // progressBar.setVisibility(View.GONE);

                try {
                    //converting response to json object
                    JSONObject obj = new JSONObject(response);

```

```

        //if no error in response
        if (!obj.getBoolean("error")) {
            Toast.makeText(getApplicationContext(), obj.getString("message"),
Toast.LENGTH_SHORT).show();

            //getting the user from the response
            JSONObject userJson = obj.getJSONObject("user");

            /* //creating a new user object
            User = new User(
                userJson.getInt("id"),
                userJson.getString("username"),
                userJson.getString("email"),
                userJson.getString("gender")
            );

            //storing the user in shared preferences
            SharedPrefManager.getInstance(getApplicationContext()).userLogin(user);
*/

            //starting the profile activity
            finish();
            startActivity(new Intent(getApplicationContext(), Login.class));
        } else {
            Toast.makeText(getApplicationContext(), obj.getString("message"),
Toast.LENGTH_SHORT).show();
        }
    } catch (JSONException e) {
        e.printStackTrace();
    }
}

},
new Response.ErrorListener() {
    @Override
    public void onErrorResponse(VolleyError error) {
        Toast.makeText(getApplicationContext(), error.getMessage(),
Toast.LENGTH_SHORT).show();
    }
}) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("username", susername);
        params.put("email", semail);
        params.put("password", spassword);
        params.put("phone", sphone);
        return params;
    }
};

VolleySingleton.getInstance(this).addToRequestQueue(stringRequest);

}
}

```

Screenbrightness.java

```
package com.example.handwrittenpassword;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Build;
import android.os.Bundle;
import android.provider.Settings;
import android.support.annotation.NonNull;
import android.support.annotation.RequiresApi;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import com.android.volley.AuthFailureError;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.StringRequest;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.FirebaseApp;
import com.google.firebase.ml.vision.FirebaseVision;
import com.google.firebase.ml.vision.common.FirebaseVisionImage;
import com.google.firebase.ml.vision.text.FirebaseVisionText;
import com.google.firebase.ml.vision.text.FirebaseVisionTextDetector;
import com.williamwww.silkysignature.views.SignaturePad;

import org.json.JSONException;
import org.json.JSONObject;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.LinkedHashSet;
import java.util.List;
import java.util.Map;
import java.util.Random;
import java.util.Set;

public class ScreenBrightness extends AppCompatActivity {
    ArrayList<Integer>numbers;
    Button;
    public int rand_int1;
    Boolean ans;
    Random;
    String[]rannumbers;
    int bright = 0;
```

```

private SignaturePad mSignaturePad;
private Button mClearButton;
private Button mSaveButton;
Bitmap signatureBitmap;

int count = 0;
//public ArrayList<Integer> stroke;
//public ArrayList<String> password;
String[] upassword;
String[] ustroke;

HashMap<Integer,String> password;
HashMap<Integer,Integer> stroke;

HashMap<Integer,Integer> hstroke;
HashMap<Integer,String> hpassword;
int brightcount = 0;
int scout = 0;
ArrayList<String>isnum;
int strokeCount =0;
public String uemail;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_register_password);
    FirebaseApp.initializeApp(this);

    SharedPreferences = getSharedPreferences("mypref",MODE_PRIVATE);
    uemail = sharedPreferences.getString("phone",null);
    insert();

    mSignaturePad = (SignaturePad) findViewById(R.id.signature_pad);

    password = new HashMap<>();
    stroke = new HashMap<>();

    hstroke = new HashMap<>();
    hpassword = new HashMap<>();

    isnum = new ArrayList<>();
    isnum.add("1");
    isnum.add("2");
    isnum.add("3");
    isnum.add("4");
    isnum.add("5");
    isnum.add("6");
    isnum.add("7");
    isnum.add("8");
    isnum.add("9");
    isnum.add("0");

    //----- for random numbers-----

```

```

numbers = new ArrayList<>();
random = new Random();
rannumbers = test().split("@");
Log.e("tets",rannumbers[bright]);
//-----
settingPermission();

if(rannumbers[bright].equalsIgnoreCase("2")){
    changeScreenBrightness(getApplicationContext(),10);
}

else if(rannumbers[bright].equalsIgnoreCase("4")){
    changeScreenBrightness(getApplicationContext(),10);
}
else {
    changeScreenBrightness(getApplicationContext(),120);
}

/*
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Log.e("test",rannumbers[bright]);

        if(rannumbers[bright].equalsIgnoreCase("2")){
            changeScreenBrightness(getApplicationContext(),10);
        }

        else if(rannumbers[bright].equalsIgnoreCase("4")){
            changeScreenBrightness(getApplicationContext(),10);
        }
        else {
            changeScreenBrightness(getApplicationContext(),120);
        }
        bright++;
    }
});
*/

mSignaturePad.setOnSignedListener(new SignaturePad.OnSignedListener() {
    @Override
    public void onStartSigning() {

        strokeCount++;

        // Toast.makeText(MainActivity.this, "OnStartSigning", Toast.LENGTH_SHORT).show();
    }
    @Override
    public void onSigned() {
        mSaveButton.setEnabled(true);
        mClearButton.setEnabled(true);
    }
}

```

```

        @Override
        public void onClear() {
            mSaveButton.setEnabled(false);
            mClearButton.setEnabled(false);
        }
    });
    mClearButton = (Button) findViewById(R.id.clear_button);
    mSaveButton = (Button) findViewById(R.id.save_button);

    mClearButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            mSignaturePad.clear();
            strokeCount=0;

        }
    });

    mSaveButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            signatureBitmap = mSignaturePad.getSignatureBitmap();
            if (signatureBitmap != null) {
                runTextRecognition();
                mSignaturePad.clear();
                for (Map.Entry<Integer, String> entry : hpassword.entrySet()) {
                    Log.e("test2", "Key = " + entry.getKey() +
                        ", Value = " + entry.getValue());
                }

            } else {
                showToast("Choose a proper image");
            }

        }
    });
}

private void changeScreenBrightness(Context context, int screenBrightnessValue)
{
    Settings.System.putInt(context.getContentResolver(),
Settings.System.SCREEN_BRIGHTNESS_MODE,
Settings.System.SCREEN_BRIGHTNESS_MODE_MANUAL); // Apply the screen brightness value to
the system, this will change the value in Settings ---> Display ---> Brightness level.
    Settings.System.putInt(context.getContentResolver(), Settings.System.SCREEN_BRIGHTNESS,
screenBrightnessValue);
}

public void settingPermission() {

    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        if (!Settings.System.canWrite(getApplicationContext())) {
            Intent = new Intent(Settings.ACTION_MANAGE_WRITE_SETTINGS, Uri.parse("package:" +
getPackageName()));
            startActivityForResult(intent, 200);
        }
    }
}

```

```

    }
}

private void runTextRecognition() {
    FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(signatureBitmap);
    FirebaseVisionTextDetector detector = FirebaseVision.getInstance()
        .getVisionTextDetector();
    detector.detectInImage(image)
        .addOnSuccessListener(
            new OnSuccessListener<FirebaseVisionText>() {
                @Override
                public void onSuccess(FirebaseVisionText texts) {

                    processTextRecognitionResult(texts);
                }
            })
        .addOnFailureListener(
            new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    // Task failed with an exception

                    e.printStackTrace();
                }
            });
}

private void processTextRecognitionResult(FirebaseVisionText texts) {

    StringBuilder t = new StringBuilder();

    List<FirebaseVisionText.Block> blocks = texts.getBlocks();
    if (blocks.size() == 0) {
        showToast("No text found");
        strokeCount=0;

        return;
    }
    for (int i = 0; i < blocks.size(); i++) {
        t.append(" ").append(blocks.get(i).getText());
    }
    showToast(t.toString());
    showToast(String.valueOf(strokeCount));
    Log.e("brightcount", String.valueOf(brightcount));
    Log.e("err1", rannumbers[bright]);
    Log.e("err2", t.toString().trim());
    Log.e("err2", hpassword.get(brightcount));

    if(!isnum.contains(t.toString().trim())){
        Toast.makeText(getApplicationContext(),"Please enter number from 0-9",Toast.LENGTH_SHORT).show();

        return;
    }
}

```

```

if(rannumbers[bright].equalsIgnoreCase("2")){
    // password.put(count,t.toString());
    Log.e("err",hpassword.get(brightcount));

    if(t.toString().trim().equalsIgnoreCase(hpassword.get(brightcount))) {
        Log.e("err1", hpassword.get(brightcount));
        password.put(count, t.toString().trim());
        stroke.put(count, strokeCount);
        //strokeCount=0;
        count++;
    }
}
else if(rannumbers[bright].equalsIgnoreCase("4")){
    if(t.toString().trim().equalsIgnoreCase(hpassword.get(brightcount))) {
        //password.put(count,t.toString());
        Log.e("err1", hpassword.get(brightcount));
        Log.e("err1", hpassword.get(brightcount));
        password.put(count, t.toString().trim());
        stroke.put(count, strokeCount);
        //strokeCount=0;
        count++;
    }
}
else {
    password.put(count,t.toString().trim());
    stroke.put(count,strokeCount);
    if(brightcount<3) {
        brightcount++;
    }
    count++;
    Log.e("err","hi");
}
scount++;
strokeCount=0;

if(bright<=4) {
    bright++;
    if (rannumbers[bright].equalsIgnoreCase("2")) {
        changeScreenBrightness(getApplicationContext(), 10);
    } else if (rannumbers[bright].equalsIgnoreCase("4")) {
        changeScreenBrightness(getApplicationContext(), 10);
    } else {
        changeScreenBrightness(getApplicationContext(), 120);
    }
}
if(scount == 6) {
    for (Map.Entry<Integer, String> entry : password.entrySet()) {
        Log.e("test1", "Key = " + entry.getKey() +
            ", Value = " + entry.getValue());
    }
    for (Map.Entry<Integer, String> entry : hpassword.entrySet()) {
        Log.e("test2", "Key = " + entry.getKey() +
            ", Value = " + entry.getValue());
    }
}

```

```

        for (Map.Entry<Integer, Integer> entry : hstroke.entrySet()) {
            Log.e("test3", "Key = " + entry.getKey() +
                ", Value = " + entry.getValue());
        }
        for (Map.Entry<Integer, Integer> entry : stroke.entrySet()) {
            Log.e("test4", "Key = " + entry.getKey() +
                ", Value = " + entry.getValue());
        }
        //Toast.makeText(getApplicationContext(),"finish",Toast.LENGTH_SHORT).show();
        Boolean test1 = hpassword.equals(password);
        Boolean test2 = hstroke.equals(stroke);
        if(test1&&test2){
            Toast.makeText(getApplicationContext(),"success",Toast.LENGTH_SHORT).show();
            changeScreenBrightness(getApplicationContext(),120);
            Intent = new Intent(getApplicationContext(),Success.class);
            startActivity(intent);
            finish();
        }
        else {
            Toast.makeText(getApplicationContext(),"Incorrect
Password",Toast.LENGTH_SHORT).show();
            finish();
        }
    }
}
private void showToast(String message) {
    Toast.makeText(getApplicationContext(), message, Toast.LENGTH_SHORT).show();
}
public String test(){
    int[] randomSequence = new int[6];
    Random randomNumbers = new Random();

    for (int i = 0; i < randomSequence.length; i++ ) {
        if (i == 0) { // seed first entry in array with item 0
            randomSequence[i] = 0;
        } else { // for all other items...
            // choose a random pointer to the segment of the
            // array already containing items
            int pointer = randomNumbers.nextInt(i + 1);
            randomSequence[i] = randomSequence[pointer];
            randomSequence[pointer] = i;
            // note that if pointer & i are equal
            // the new value will just go into location i and possibly stay there
            // this is VERY IMPORTANT to ensure the sequence is really random
            // and not biased
        } // end if...else
    } // end for
    String ss = "";

    for (int number: randomSequence) {
        Log.e("test", String.valueOf(number));
        ss=ss+number+"@";
    } // end for
    ss = ss.substring(0, ss.length() - 1);
    return ss;
}

```

```

    }
    public void insert(){
        StringRequest = new StringRequest(Request.Method.POST, URLs.VERIFY_PASSWORD,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {

                    try {
                        //converting response to json object
                        JSONObject obj = new JSONObject(response);
                        Log.e("test",response);

                        //if no error in response
                        if (!obj.getBoolean("error")) {
                            Toast.makeText(getApplicationContext(), obj.getString("message"),
                                Toast.LENGTH_SHORT).show();

                                JSONObject userJson = obj.getJSONObject("user");

                                upassword = userJson.getString("password").split("@");
                                ustroke = userJson.getString("stroke").split("@");

                                int i=0;
                                int j=0;

                                for (String te:upassword){
                                    hpassword.put(i,te);
                                    i++;
                                }

                                for (String in:ustroke){
                                    hstroke.put(j, Integer.valueOf(in));
                                    Log.e("ted", String.valueOf(hstroke));
                                    j++;
                                }

                                /*upassword= new
                                ArrayList(Arrays.asList(userJson.getString("password").split("@")));
                                ustroke= new ArrayList(Arrays.asList(userJson.getString("stroke").split("@")));*/

                                } else {
                                    Toast.makeText(getApplicationContext(), obj.getString("message"),
                                        Toast.LENGTH_SHORT).show();
                                }
                            } catch (JSONException e) {
                                e.printStackTrace();
                            }
                        }
                    },
                    new Response.ErrorListener() {
                        @Override
                        public void onErrorResponse(VolleyError error) {

```

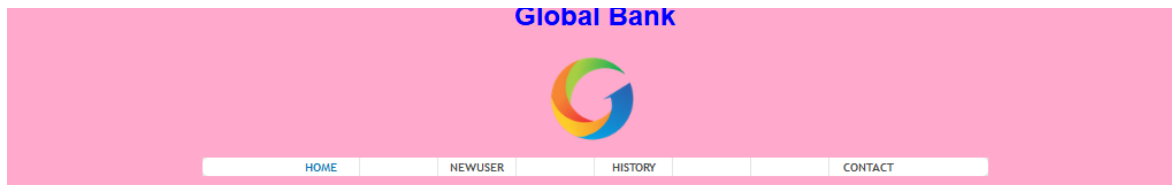
```

        Toast.makeText(getApplicationContext(), error.getMessage(),
Toast.LENGTH_SHORT).show();
    }
    }) {
    @Override
    protected Map<String, String> getParams() throws AuthFailureError {
        Map<String, String> params = new HashMap<>();
        params.put("uid", uemail);
        return params;
    }
};

VolleySingleton.getInstance(this).addToRequestQueue(stringRequest);
}
}

```

B. SCREENSHOTS:



Admin Login

Username:

Password:



Account Information

Account No:

Account Holder:

Balance Amount:

Mobile Number:

E Mail:

Address:

City:

PinCode:

State:

[HOME](#)[NEWUSER](#)[ADDAMOUNT](#)[LOGOUT](#)

New Account Holder

Account Holder Name:

Mobile Number:

E-mail:

Address:

City:

PinCode:

State:

Global Bank



Add Amount

Account No:

Account Holder:

Amount:

[HOME](#)[CONTACT](#)

Global Bank



HOMENEWUSERADDAMOUNTLOGOUT

Account Information

Added Successfully.

Account No:

52155003

Account Holder:

Asha

Balance Amount:

10729

Mobile Number:

9384675735

E Mail:

asha@gmail.com

Address:

Thoraipakkam

City:

Karaikurichi

PinCode:

627417

State:

TAMIL NADU

Edit

63

11:15 AM

🕒 📶 📶 📶 🔋 82%

EcomSample



Denim Shirt

Rs. 1,979

FREE Delivery

4.3 *

50 ratings & 15 reviews



Share



Similar



Wishlist

Details

- Regular fit, full sleeve
- Fabric: Cotton
- Pattern: printed

ADD TO CART

BUY NOW

11:16 AM

🕒 📶 📶 📶 🔋 82%



EcomSample



OFFERS

ELECTRONICS

LIFESTYLE

HOME APP



Item Name

Item descriptions

Rs. 8,999



Item Name

Item descriptions

Rs. 8,999



Item Name



Item Name

11:17 AM



My Cart



Denim Shirt

Slim fit, cotton

Rs. 8,999

Delivery charge: FREE

Qty: 1 Size: 41

✕ Remove

 Edit



Denim Shirt

Slim fit, cotton

Rs. 8,999

Delivery charge: FREE

Qty: 1 Size: 41

✕ Remove

 Edit



Denim Shirt

Slim fit, cotton

Rs. 8,999

Delivery charge: FREE

Qty: 1 Size: 41

✕ Remove

 Edit

Rs 8,750

PAYMENT

11:17 AM



Mukesh Mikku



HOME APP



Offers



Electronics



Lifestyle



Home Appliances



Books & More



More

My Profile



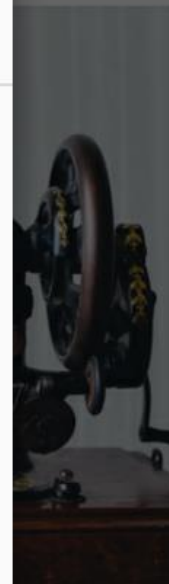
My Orders



My Cart



My Wishlist



11:18 AM



My Wishlist



Denim Shirt
Slim fit, cotton
Rs. 8,999



Denim Shirt
Slim fit, cotton
Rs. 8,999



Denim Shirt
Slim fit, cotton
Rs. 8,999



Denim Shirt
Slim fit, cotton
Rs. 8,999



Denim Shirt
Slim fit, cotton
Rs. 8,999



11:20 AM

🕒 📶 📶 📶 🔋 81%



Great Discounts

Best discounts on every single service
we offer!

SKIP



NEXT



BioTouchPass

Enter Ip

ENTER IP



BioTouchPass

 Email

 Password

Login

New User? Register



12:49 PM

BioTouchPass



Name



Phone number



Gmail



Password

Register

Login



Fastrack 3166KM01
Fastrack by Titan, was lai

Rs. 2362



Fastrack 68007PP03
Fastrack by Titan, was lai

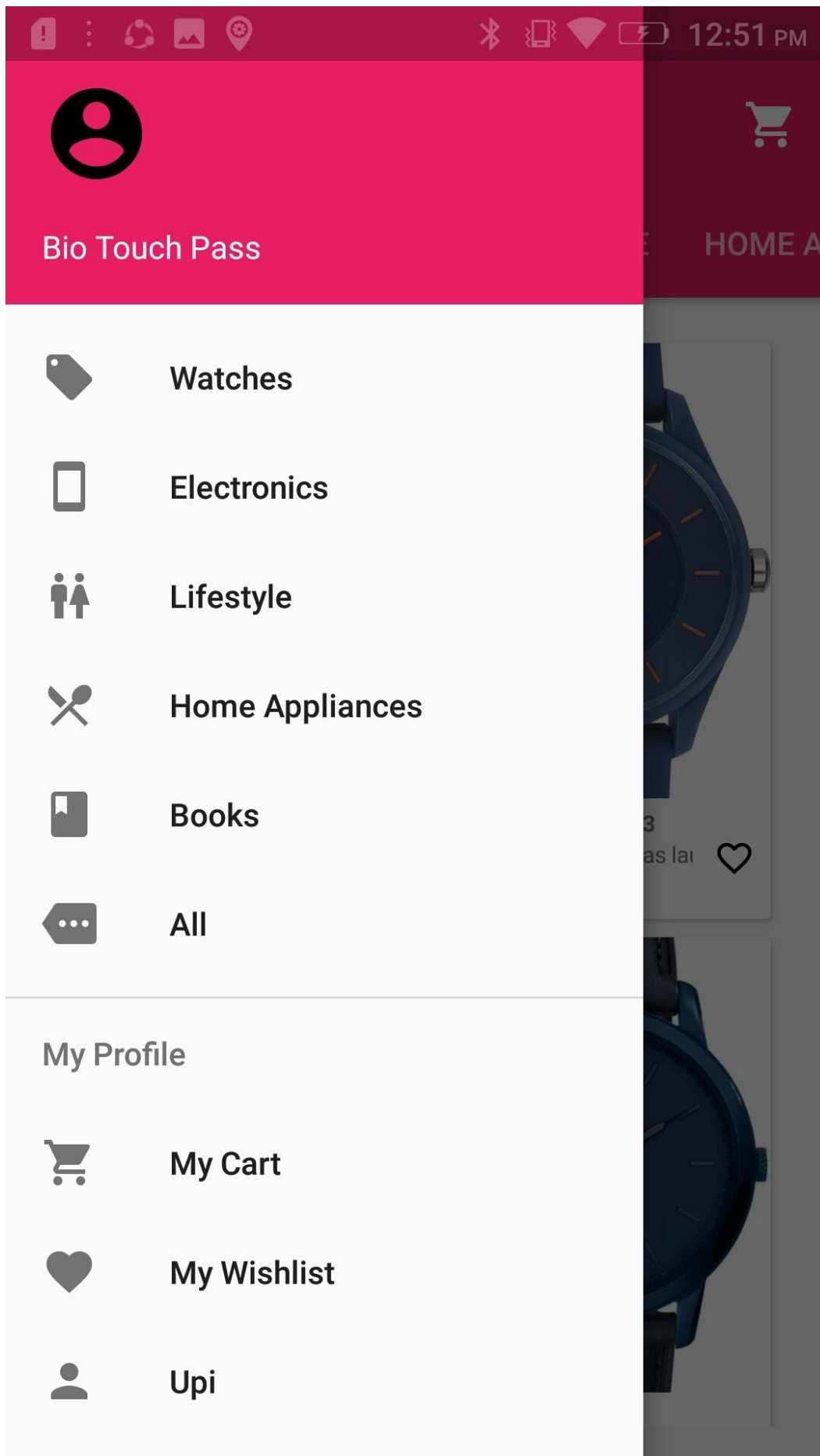
Rs. 1271



Fossil FS4662



Fossil FS5448





BioTouchPass



Fastrack 68007PP03

Rs. 1271

FREE Delivery

4.3 *

50 ratings & 15 reviews



Share



Similar



Wishlist

Details

Fastrack by Titan, was launched in 1998 and became an independent urban youth brand in 2005. Since then, it has carved a niche for itself with watches and sunglasses that are both fashionable and affordable.

ADD TO CART

BUY NOW



My Cart



Fastrack 68007PP03

Fastrack by Titan, was launched

Rs. 1271

Delivery charge: FREE

Qty: 1

✕ Remove

1271

PAYMENT



My Wishlist



Fastrack 68007PP03

Fastrack by Titan, was launched in 1998 and became an independent urban youth brand in 2005. Since then, it has carved a niche for itself with watches

Rs. 1271



C. PUBLICATION WITH PLAGIARISM REPORT

A CONTEMPORARY CLOUD BASED DYNAMIC AUTHENTICATION SYSTEM FOR MOBILE APPLICATIONS

Sarah Philomin V, Senduru Srinivasalu, Abirami M

Department of IT - School of Computing

Sathyabama Institute of Science and Technology – Chennai - 600119

sarahphilomin@gmail.com, sendurusrinivas@gmail.com, mabirami1211@gmail.com

Abstract -- For a variety of reasons, this identification strategy planned several compensations over outdated techniques including ID cards (tokens) or PIN statistics (passwords):

(i) The person who is to be recognized is compulsory to physically exist at the point of identification.

(ii) Identification based on biometric methods eliminates the need to remember a password or carry an encrypted token. We are attempting to understand the significance of mobile devices for society. Using user-friendly safe mobile applications, they can distribute their services ensuring data security and protection and can be hosted on cloud service both as client-side service and backend service.

Keywords -- Biometrics, Cryptography, Secure Computing, Machine Learning, Android, Optical Recognition System, Amazon Web Service Cloud.

1. INTRODUCTION

This research paper surveys the advantages and capability of incorporating biometrics into secret word-based versatile confirmation frameworks by expecting everybody to draw the example of each digit of the secret word on the touch screen rather than composing it. Instead of typing, our system instructs users to draw them. Users can use stroke count, time limit, screen brightness, or sensor-based authentication system as a second-stage password. This research assesses the merits and possibilities of incorporating biometrics into password-based mobile protection systems by considering the users to draw every digit of the password on the touch screen rather than entering it. Instead of typing, our system urges

users to doodle. Users can utilize a stroke count, time limit, screen brightness, or a sensor-based authentication technique as a second-stage password.

Cell phones have evolved into a necessary tool for many people. The constant and rapid global deployment of mobile devices has been a success not only by the technological advancements but new features that have been added, as well as new internet infrastructures such as 5G, which allows for communication and use among many other factors. Hence, both the sector people are careful of the situation. We are attempting to understand the significance of mobile devices for society. Using user-friendly safe mobile applications, they can distribute their services ensuring data security and protection. Traditional authentication methods related to (PIN) Personal Identification Number including (OTP) One-Time Passwords are improved by using biometric data as the next level of authentication.

Banks frequently send a numerical password to the user's mobile device (usually between 6 and 8 digits). To complete the payment, the user must enter this numerical password into the security platform. While drawing the digits, our suggested solution improves this scenario by introducing an improvised protection system related to the person who uses biometric information.

The common architecture of this proposed authentication-based mobile authentication solution is shown in Fig. 1. This research looks at the three primary modules listed below:

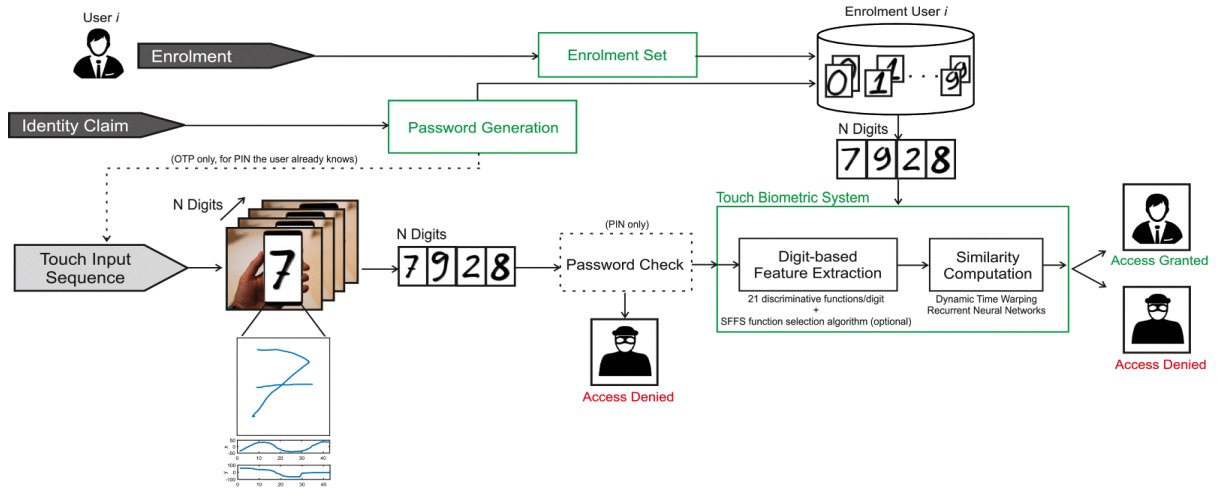
1. Ecommerce and User Authentication Look at the product
2. Biometric Handwritten Password for Cart and Payment
3. Using Strokes to Create a Biometric Password
4. Using Screen Brightness and Time to Create a Biometric Password

The hand-drawn patterned digits could be identified by an (OCR) Optical Character Recognition methodology to validate the authenticity of the password, based on the ultimate web service. Following this initial authentication stage, the biometric information of the handwritten digits as compared to the claimed user's enrollment data in a second authentication stage, comparing each digit one by one. In this study, we concentrate on the next level authentication stage, which will be considered on the person's behavioral data while executing handwritten digits, as numerical digit recognition has been proven to be a nearly solved problem with mistakes of less than 1%.

2. LITERATURE SURVEY

The mobile applications are simple to use. The output is excellent. Public verification methods are improved by integrating changeable handwritten biometric data. The disadvantage was that it took a long time to complete.[1] The sequential advance drifting selection procedure is used to investigate the most discriminant. Because the mobile device is delicate, pull the bin slowly and cautiously. It has a good rate of accuracy.[1] Could make one-to-many decisions. Affordably priced equipment. simple to put into effect. Only well and earliest biometric knowledge. The disadvantage is that genuine finger scans cannot be generated by starting with a stenciled picture. Users link fingerprint recognition with illegal activity.[2] Biometrics gadget is readily identifiable from human behavior and health state standpoint. Biometric systems and digitizing tablets are required. [6] It takes less time. The cost of a biometric system is significant. This is particularly important given the hazards associated with exploiting these behavioral signals. The fundamental concept is to overlay the new sensor's feature dispersion onto that of the "old" sensor.[3] Under the premise of having a small number of fresh examples accessible, this technique enables the separation of "real" and "fake" classes even without the necessity for training a novel classifier. When using competent forgeries, our suggested technique delivers exceptional results with EERs of around 5.0 percent, beating other classic biometric recognition qualities, for example, transcribed marks or graphical passwords on cell phone conditions. This idea is difficult to understand if simply standard One-Time Password security

instruments given PINs are utilized.[4] In such mobile settings, traditionally grown entry point authentication techniques can be enhanced. However, everybody keeps pertinent data on their phones, such as bank information, emails, or credentials, which might be hacked. The authenticity of unrestricted sketching is investigated in this study.[5] Based on dynamic signature verification methodologies, verification systems are based on dynamic temporal warping. The top worth frameworks on the contrary of randomized and master falsifications were adjusted independently for each case and converging of the two frameworks offered by and large extraordinary outcome in all conditions.[8] The research reveals that the usage of SNS smartphone platforms is a strong predictor of technology addiction, although the results were not conclusive. Progressing verification frameworks were recommended, in which clients are continually seen after accessing the cell phone interestingly.[7] Attackers might quickly get access to many accounts by attempting the most used credentials. Furthermore, when hidden patterns are used to acquire initial entry on portable devices, users prefer to repeat the same pattern. At the point when the amount of planning marks available to make the client design rises, the congruity of propelling the system arrangement conversely, with a legitimate framework structure creates. [10] Signature variability is one of the most difficult difficulties in signature verification. Another significant issue relating to intrinsic or based on inter variability of signatures is referred to as aging. [9] When contrasted with standard frameworks, this article offered a normal relative increase in the execution of 61.3 percent and 28.5 percent, separately, as far as Equal Error Rate. Device interoperability is a difficult topic in biometrics. Signature variability is one of the most difficult difficulties in signature verification. In many computer vision tasks, the (CNN) convolutional neural network has achieved remarkable success.[11] The precise roles of these synapses are unknown. The purpose of this article is to look at touchscreen gestures for single signs on touch-sensitive devices. It has been discovered that executing a user-defined gesture over modified picture results in improved verification performance. [13]The use of biometrics in cellular contexts is rapidly expanding. Several studies on fingerprint identification have been performed in western and the far countries, but they provide information.[12] The internet employs a diverse mix of thirty usability, rapid deployment, and safety advantages



that a perfect system may supply. Our thorough method yields important insights into the difficulties of resetting passwords. Those credentials are afflicted by security issues and are publicly available. [14]

3. PROPOSED WORK

Handwritten signatures have been utilized in legal and financial contracts for too many years in earlier established systems, and they also have applicability in mobile circumstances. The product for gathering manually written number digits was essentially evolved to diminish the client's unusualness all through the acquisition technique. However, a crucial aspect is that the password wasn't strong enough for a certain application. Our second time of endorsement recalls different choices for the light of client inclination where the client can create different game-plan of blends. The client can set second stage secret words such as stroke, screen wonder, or sensor-based underwriting structure, time. The joining of biometric information on standard mystery express gathered systems can work concerning the security through a second level of client statement.

Problem Definition:

How much information was asked by the client all through the election cycle? The level of wellbeing is given by the cautious structure. As per the perspective of the security system, the best decision is to have as much data about the person as viable.

Proposed System:

Our proposed structure turnaround gives clear versatile applications ensuring data protection

and high security. A client should draw each digit of the mystery key on the touch screen rather than outline them unbelievably. Consequently, the standard endorsement structures are improved by joining dynamics created by hand biometric data. Our construction fuses two times of attestation the pulled-in pin should be like the pin entered during the enrollment process. Including the same, it focuses on offering consumer mobile applications while protecting data and maintaining strict security. Instead of inputting the password, the user should sketch each digit on the touch screen. Traditional cryptographic methods are reinforced in this way by the inclusion of dynamic handwritten biometric data and sensor information on traditional passcode systems, which can enhance safety by requiring a second level of user verification.

Traditional cryptographic methods are reinforced in this way by the inclusion of dynamic handwritten biometric data and sensor readings on traditional passcode systems, which can enhance safety by demanding an additional level of user verification. This exploration evaluates the benefits and potential outcomes of fusing biometrics into pin code portable cryptographic strategies by requesting that clients draw every digit of the secret key on the touchscreen rather than entering it. Instead of typing, our technology recommends that users sketch. Users can utilize a stroke count, time limit, illumination variation, or a camera authentication technique as a second-stage password. Variation or stumbles found at this development in this task is adjusted going before finishing the undertaking with the assistance of the client by haggling to fan out a methodology for settling needs. Thusly the proposed framework achievable has been endeavored by utilizing underwriting testing and considered working enough. However, there were lacks in the framework they were

FIGURE 1: SYSTEM ARCHITECTURE

not devastating. Traditional password-based systems can be made more secure by storing biometric information in a cloud or local database and retrieving it after safe authentication. To make it available as a cloud service, so that users may simply use the application without having to install it on their local computer.

ADVANTAGES COMPARING TO EXISTING SOLUTIONS:

- These approaches enable active or continuous authentication schemes, in which the user is transparently authenticated.
- Handwritten signature is one of the most social accepted biometric.
- The incorporation of biometric information on traditional password-based systems can improve the security through a second level of user authentication.

4. ARCHITECTURE DIAGRAM

Rather than typing out the pin or passwords, Traditional authentication methods are improved in this way by adding dynamic handwritten biometric data. Authentication takes place in two steps in our system. The pin drawn should match the pin that's drawn while new registration. Our next level of authentication allows customers to choose from a variety of possibilities based on their preferences, allowing them to create various sets of pairs. According to Fig.1, Users can measure the time, use a variety of screen brightness, stroke count, or sensor-based authentication mechanism as a second stage password. By adding another level of authentication to regular password-based systems, biometric information can develop and improvise security. After entering the PIN, each digit is recognized using OCR (Optical Character Recognition) machine learning algorithm as a google API.

5. ALGORITHMS AND CONCEPTS USED

Algorithmically Calculated:

- 1.MD5/HMAC
- 2.AES

Steps:

User Registration:

1. While enrolling client needs to draw PINs.
2. Utilizing OpenCV from Firebase Cloud the motion will be perceived.
3. Keystrokes are followed at the same time
4. Recognized number and strokes count information will be encoded and scrambled
5. Utilizing MD5 the signal information is encoded into hash
6. Utilizing AES symmetric key encryption the encoded information gets scrambled
7. Secret word Cipher text will be put away in the data set.

MD5 – Message-Digest:

To choose cryptographic encrypting regarding the Java, Message Digest Class was used, considering the get-together java. Message Digest Class enables along with cryptographic hash to see the hash worth of a message, such as:

1. MD5
2. SHA-1
3. SHA-256

Algorithms are in the state in static strategy whose get Instance(). Only resulting in choosing the computation it works the outline worth and returns the results in byte show. BigInteger class is used, which changes over the resultant byte give into its indication degree depiction. This portrayal changes over into hexagonal arrangement to obtain the Message Digest.

AES – Advanced Encryption Standard:

The symmetric-key square code assumes a significant part in information encryption. It implies that a similar key is utilized for both encryption and decoding. The Advanced Encryption Standard (AES) is a generally utilized symmetric-key encryption calculation. The AES calculation is an iterative, symmetric-key square code that upholds cryptographic keys.

Encryption and Decryption:

We first need to create the mystery key and IV as indicated by the past area. In the following stage, we make an example from the Cipher class by utilizing the getInstance() technique. Furthermore, we design a code

case utilizing the unit(a) strategy with a mystery key, IV, and encryption mode. This technique gets bytes of information and returns ciphertext in bytes.

6. PROJECT IMPLEMENTATION

The client would draw every digit of the secret articulation on the touch screen rather than making them amazingly. Thusly, the conventional check frameworks are improved by entwining dynamic interpreted biometric data. Our design consolidates two times of confirmation the drawn pin should be like the pin entered during the enlistment process. Reference to the below figure.2., Client can set second stage secret articulation as stroke, time, screen miracle, or sensor-based confirmation structure. The circuit of biometric data on standard secret word setup frameworks can work for security through a second degree of client confirmation.

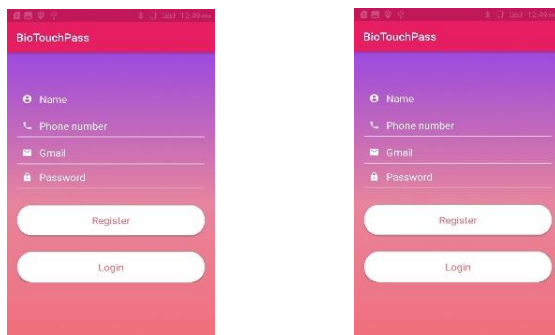


FIGURE 2:

DASHBOARD OF THE APPLICATION

a. User Authentication and Ecommerce View Product

The user goes through a basic application or registration process. People should provide their secured information to use this method. That information is then stored in the server-side database, and the user may witness a set of items on their website, as well as their details.

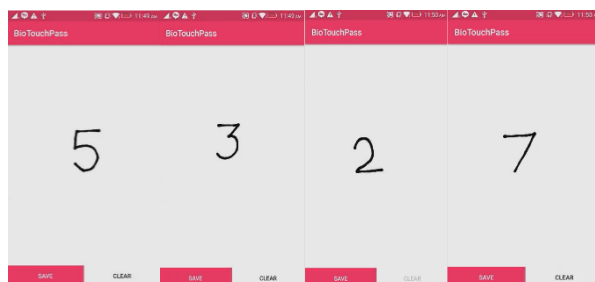


FIGURE 3:

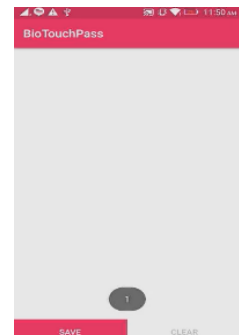
BIOMETRIC PASSWORD AUTHENTICATION

b. Registration Using Biometric Hand-Written Password

In figure 3, The user may choose from a list of products that they want to buy. The chosen products will appear on a cart page, where the user can complete the usual purchasing information. While finishing up broad data, the client should portray each four-digit nail individually to the screen. The client's secret phrase was then affirmed by changing over the brought secret word into a picture utilizing optical person acknowledgment numbers from each image got.

c. Biometric Password Using Strokes

After finishing the affirm secret key strategy, the client should enter their 4-digit secret word commonly during the enrollment cycle. From figure 4 we can see the stroke count and the Client should affirm their secret key by utilizing a similar secret key with a similar



stroke. The strokes of each drawn digit should match the strokes given at the hour of enlistment

FIG4:

STROKE COUNT DISPLAY

d. Biometric Password Using Screen Brightness and Time

By presenting the thought of involving screen splendor as a confirmation instrument, spyware assaults will be turned away. The 6-digit twofold worth is produced by the Android safe climate. The brilliance of the display is adjusted to low and high contingent upon twofold digits. The client ought to enter the legitimate PIN digit on the off chance that the screen splendor is high. If not, the client ought to give a wrong and irregular PIN. To stay away from a MAN-IN-MIDDLE assault, the framework will erase the digits that were input when the screen splendor was low, apply the HMAC calculation on the secret phrase provided by the client, and produce an edible Signature for the client PIN. The server gets the mark for the client-created PIN, ascertains the mark as an incentive for the Original PIN, and thinks about the two marks. The client can get to the client's profile on the off chance that the two marks were equivalent or, likely access is denied.

7. RESULTS AND DISCUSSION

OCR (optical character recognition) – machine learning concept is used to identify the numbers drawn and is then sent to the process of HMAC and HMAC (hash-based message authentication code) – secure computing concept is used to convert the pin entered where a hash code which is later clubbed into one and stored in our database.

The new e-Bio Digit edifying arrangement, which recalls for line made by hand digits from 0 to 9, has been procured recollecting the finger as liability regarding a telephone. This database is used in the starters reported in this work and it is available nearby benchmark achieves GitHub1. Finally, we talk about unequivocal nuances for the relationship of our proposed approach on current PIN and OTP structures, achieving results with Equal Error Rates (EERs) ca. 4.0% when the aggressor knows the mysterious enunciation. These results support the sending of our proposed approach on the other hand, with standard PIN and OTP systems where the attack

would have a 100% accomplishment rate under a comparable stunt circumstance

AWS implementation can be done in either way,

3. Client-side authentication where the authentication service can be requested to any authorization to add them as an additional protection service
4. Backend management where the AWS could host the service and several other organizations could share it with their customers and they can directly access from our instance.

8. TEST CASE AND PERFORMANCE CHART

Functionality Testing:-

Supportive examinations included practicing the code with clear information values for which the common outcomes are suggested, as well as limit values and uncommon attributes, as truly related inputs, records of indistinct parts, and void reports.

Three kinds of tests in Functional test:

- Execution rate test
- Stress rate Test
- Structure rate Test

The significant mistake that was looked during the task is connecting blunder. Whenever every one of the modules is consolidated the connection isn't set as expected with all help documents. Then, we took a gander at interconnection and the associations. Bumbles are confined to the new module and its intercommunications. The thing progression can be coordinated, and modules joined in as they complete unit testing. Testing is done when the last module is fused and attempted, and the results are tabulated below.

**TABLE 1:
SYSTEM FUNCTIONALITY TEST RESULTS:**

DESCRIPTI ON	TEST INPUTS	EXPECTE D OUTPUT	STATUS

Gesture Identification	0 – 9 numbers are drawn and checked	0-9 numbers Recognised	Pass
Keystrokes Count	Numbers are drawn with different strokes	Stroke counts Comparison	Pass
Numbers 6 and 9 Draw frequently	6 & 9 are drawn	6 & 9 to be recognized	Fail
Is Encoded encrypted Cipher stored in DB	Do API call	Ciphertext stored or not	Pass

Performance Chart:-

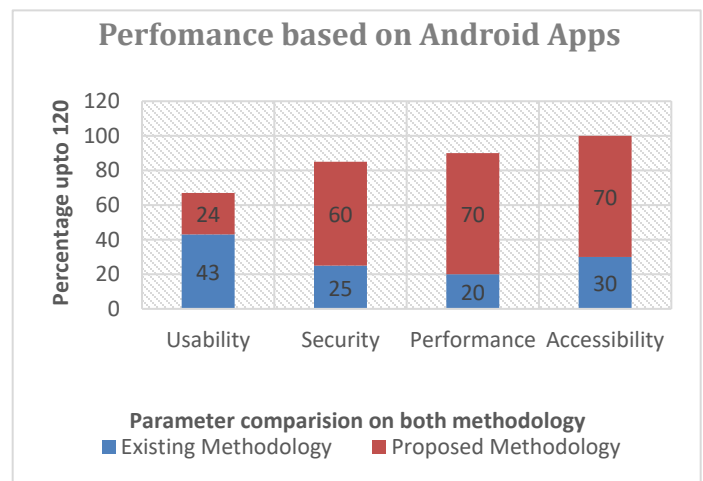
In the existing password approach, the passwords will be put away in an encoded design at information base while every single time when an action is set off it will be contrasted and the put away secret phrase, yet in our proposed strategy we worry about defeating digital assaults like shoulder riding assault and man in the center assault by putting away and looking at a regular secret phrase procedure utilizing PINs draw design and dealing with the vibe sensors for dynamic security improvement. Every one of these useful and client testing reports has produced the exhibition diagram.

9. CONCLUSION

This work assesses the savvy method for verifying the informal communication accounts having a place with them by utilizing the screen brilliance of android mobiles to keep away from the spyware assault, shoulder riding assault, and man in the center assault and to have them in a virtual server which would be gainful to individuals anyplace around the globe. To ad-lib the equivalent and distribute it in the

motivation behind each organization administration could add equivalent to a secret phrase system and we can add it to our essential applications which are to be secured. The advantages and plausibility of bringing written by hand contact biometrics into secret word-based portable confirmation frameworks are inspected in this review.

The examinations point by point in this paper utilized the new e-Bio Digit information base, which contains manually written mathematical digits from zero to nine, and it is given together benchmark results. We propose the brilliant method for validating the person-to-person communication accounts having a place with them by involving the screen splendor of android mobiles to keep away from the spyware assault, shoulder riding assault,



and man in the center assault.

**FIG 5:
COMPARATIVE ANALYSIS**

10. REFERENCES

- [1]. M. Saehan and A. Megabank, "Person to person communication on Smartphones: When Mobile Phones Become Addictive," Computers in Human Behavior, vol. 29, no. 6, pp. 2632-2639, 2013.
- [2]. J. Bonneau, C. Harley, P. Oorschot, and F. Stajan, "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes," in Proc. IEEE

- Symposium on Security and Privacy, 2012, pp. 553-567.
- [3]. J. Globally, I. Chisel, and I. Sanchez, "A New Multimodal Approach for Password Strength Estimation Part I: Theory and Algorithms," *IEEE Transactions on Information Forensics and Security*, vol. 12, pp. 2829-2844, 2017.
- [4]. Aviv, K. Gibson, E. Mossop, M. Burst, and J. Smith, "Smirch Attacks on Smartphone Touch Screens," in *Proc. of the fourth USENIX Conference on Offensive Technologies*, 2010, pp. 1-7.
- [5]. D. Shukla, R. Kumar, A. Sarada, and V. Phocas, "Be careful, Your Hands Reveal Your Secrets!" in *Proc. of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014.
- [6]. Q. Yue, Z. Ling, X. Fu, B. Liu, W. Yu, and W. Zhao, "My Google Glass Sees Your Passwords!" in *Black Hat USA*, 2014.
- [7]. W. Meng, D. Wong, S. Furnell, and J. Zhou, "Reviewing the Development of Biometric User Authentication on Mobile Phones," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1268-1293, 2015.
- [8]. L. Wan, M. Zeigler, S. Zhang, Y. Lacuna, and R. Fergus, "Regularization of Neural Networks utilizing Drop Connect," in *Proc. of the 30th International Conference on Machine Learning*, 2013, pp. 1058-1066.
- [9]. M. Liang and X. Hu, "Intermittent Convolutional Neural Network for Object Recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3367-3375.
- [10]. J. Angulo and E. Eastlund, "Investigating Touch-Screen Biometrics for User Identification on Smart Phones," J. Camelish, B. Crespo, S. Fischer-Here, R. Lens, G. Rossello (Eds.), *Privacy and Identity Management forever*, Springer, pp. 130-143, 2011.
- [11]. P. Liachames and C. Rosenberger, "Coordinated One Time Biometrics With Pattern Based Authentication," in *Proc. eleventh Int. Conf. on Availability, Reliability and Security, ARES*, 2016.
- [12]. E. von Dershowitz, M. Eiland, D. Bushel, S. Ober Huber, A. D. Luca, F. Alt, and H. Hussmann, "On Quantifying the Effective Password Space of Grid-based Unlock Gestures," in *Proc. of the International Conference on Mobile and Ubiquitous Multimedia*, 2016, pp. 201-212.
- [13]. D. Bushel, A. D. Luca, and F. Alt, "There is something else to Typing besides Speed: Expressive Mobile Touch Keyboards through Dynamic Font Personalization," in *Proc. of the International Conference on Human-Computer Interaction with Mobile Devices and Services*, 2015, pp. 125-130.
- [14]. "Further developing Accuracy, Applicability and Usability of Keystroke Biometrics on Mobile Touchscreen Devices," in *Proc. of the CHI Conference on Human Factors in Computing Systems*, 2015, pp. 1393-
- [15]. L. Li, X. Zhao, and G. Xu, "Inconspicuous Reauthentication for Smartphones," in *Proc. twentieth Network and Distributed System Security Symposium, NDSS*, 2013.

- [16]. N. Sea-Bae, N. Memon, K. Is bister, and K. Ahmed, "Multitouch Gesture-Based Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 568-582, 2014.
- [17]. C. Shen, Y. Zhang, X. Guan, and R. Maxibon, "Execution Analysis of Touch-Interaction Behavior for Active Smartphone Authentication," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 3, pp. 498-513, 2016.
- [18]. J. Fierro, A. Pozzo, M. Martinez-Diaz, J. Globally, and A. Spirits, "Benchmarking Touchscreen Biometrics for Mobile Authentication," *IEEE Trans. on Information Forensics and Security*, vol. 13, 2018.
- [19]. N. Sea-Bae and N. Memon, "Online Signature Verification on Mobile Devices," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 6, pp. 933-947, 2014.
- [20]. R. Toscana, R. Vera-Rodriguez, J. Fierro, A. Spirits, and J. Ortega-Garcia, "Benchmarking Desktop and Mobile Handwriting across COTS Devices: the e-Bio Sign Biometric Database," *PLOS ONE*, 2017.
- [21]. W. Khan, M. Amsalem, and Y. Xiang, "A Graphical Password-Based System for Small Mobile Devices," *International Journal of Computer Science*, vol. 5, no. 2, pp. 145-154, 2011.
- [22]. M. Martinez-Diaz, J. Fierro, and J. Globally, "Graphical Password-based User Authentication with Free-Form Doodles," *IEEE Trans. On Human-Machine Systems*, vol. 46, no. 4, pp. 607-614, 2016.
- [23]. T. Ketzner, F. Ye, I. Benninger, C. Treviso, M. Dutta, and A. Singh, "Client Verification Using Safe Handwritten Passwords on Smartphones," in *Proc. eighth International Conference on Contemporary Computing, IC3*, 2015.
- [24]. T. Nguyen, N. Sea-Bae, and N. Memon, "DRAW-A-PIN: Authentication utilizing finger-drawn PIN on touch gadgets," *Computers and Security*, vol. 66, pp. 115-128, 2017.
- [25]. R. Toscana, R. Vera-Rodriguez, J. Fierro, and J. Ortega-Garcia, "Consolidating Touch Biometrics to Mobile One-Time Passwords: Exploration of Digits," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018.
- [26]. L. Wan, M. Zeigler, S. Zhang, Y. Lacuna, and R. Fergus, "Regularization of Neural Networks using Drop Connect," in *Proc. of the 30th International Conference on Machine Learning*, 2013, pp. 1058-1066.
- [27]. M. Liang and X. Hu, "Dull Convolutional Neural Network for Object Recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3367-3375.
- [28]. J. Angulo and E. Eastlund, "Exploring Touch-screen Biometrics for User Identification on Smart Phones," J. Camelish, B. Crespo, S. Fischer-Here, R. Focal point, G. Rossello (Eds.), *Privacy and Identity Management everlastingly*, Springer, pp. 130-143, 2011
- [29]. Fierro-Aguilar J, Alonso-Hermia N, Moreno-Marquez G, Ortega-Garcia J (2004) A

- separated imprint affirmation structure considering the blend of neighborhood and overall information. In: Proceedings of European gathering on PC vision, studio on the biometric check, BIOAW, LNCS, vol. 3087. Springer, Berlin, pp 295-306
- [30]. Toscana R, Vera-Rodriguez R, Ortega-Garcia J, Fierro J (2015) Update strategies for Homebased dynamic imprint biometric structures. In: Proceedings of seventh IEEE worldwide studio on information legitimate sciences and security, WIFS
- [31]. Fierro J, Globally J, Ortega-Garcia J, et al. (2010) Biosecurity: a multimodal biometric database. *Plan Anal Appl* 13(2):235-246
- [32]. All around the world J, Diaz-Cabrera M, Ferrer MA, Gomez-Barrera M, Morales A, Fierro J (2015) Online imprint affirmation through the mix of authentic remarkable data and falsely delivered static data. *Plan Recognin* 48:2921-2934
- [33]. Martinez-Diaz M, Fierro J, Hangar S (2015) Signature features. In: Li SZ, Jain A (eds) *Encyclopedia of biometrics*. Springer, Berlin, pp 1375-1382
- [34]. Toscana R, Vera-Rodriguez R, Fierro J, Ortega-Garcia J (2018) Exploring discontinuous neural associations for online physically composed signature biometrics. *IEEE Access* 6:5128-5138
- [35]. Liu Y, Yang Z, Yang L (2014) Online imprint check considering DCT and pitiful depiction. *IEEE Trans Cybernet* 45:2498-2511
- [36]. Diaz M, Fischer A, Ferrer MA, Plamodel R (2016) Dynamic imprint check structure thinking about one authentic imprint. *IEEE Trans Cybernet* 48:228-239
- [37]. Martinez-Diaz M, Fierro J, Krrish RP, Globally J (2014) Mobile imprint affirmation: incorporate power and execution connection. *IET Biome* 3(4):267-277
- [38]. Tempe N, Merle A, Bringer J, Gomez-Barrera M, Fierro J, Globally J (2016) Toward Common Criteria appraisals of biometric structures. BEAT public deliverable. https://www.beat-eu.org/project/expectations_public/d6-5-toward-normal-standards-assessments-of-biometric-frameworks

Sarah-Abirami Plagiarism Report

ORIGINALITY REPORT

6%	5%	3%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.geeksforgeeks.org Internet Source	2%
2	atvs.iu.uam.es Internet Source	1%
3	www.ijres.org Internet Source	1%
4	Submitted to University of Glasgow Student Paper	<1%
5	Submitted to Study Group Australia Student Paper	<1%
6	hdl.handle.net Internet Source	<1%
7	Ruben Tolosana, Ruben Vera-Rodriguez, Javier Ortega-Garcia, Julian Fierrez. "Preprocessing and Feature Selection for Improved Sensor Interoperability in Online Biometric Signature Verification", IEEE Access, 2015 Publication	<1%



Certificate of Presentation

This is to certify that

Sara Philomin V

has successfully presented the paper entitled

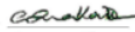
A Contemporary Cloud Based Dynamic Authentication System for Mobile Applications

at the

International Conference on Sustainable Computing and Data Communication Systems (ICSCDS 2022)

organized by Shree Venkateshwara Hi-Tech Engineering College, Erode, India

held during 7-9, April 2022.


Session Chair


Conference Chair
Dr. P. Karuppusamy


Principal
Dr. P. Thangavel