

Efficient Password Mechanism to Overcome Spyware Attacks

Submitted in partial fulfillment of the
requirements for the award of
Bachelor of Engineering degree in Computer Science and Engineering

By

**MANIKANDAN S (Reg.No - 40110726)
AJITH PETER VIANNEY R (Reg.No – 40110040)**

Under Guidance of

Dr. A. C. SANTHA SHEELA M.E., Ph.D.,



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHISALAI,
CHENNAI - 600119**

NOVEMBER - 2023



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with A++ grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **MANIKANDAN S(40110726)** and **AJITH PETER VIANNEY R(40110040)** who carried out the Project Phase-1 entitled **“EFFICIENT PASSWORD MECHANISMS TO OVERCOME SPYWARE ATTACK”** under my supervision from June 2023 to November 2023.

Internal Guide

Dr. A. C. SANTHA SHEELA M.E., Ph.D

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, **MANIKANDAN S(40110726) Ajith Peter Vianney R(40110040)**, hereby declare that the Project Phase-1 Report entitled **“EFFICIENT PASSWORD MECHANISMS TO OVERCOME SPYWARE ATTACK”** done by me under the guidance of **Dr. A. C. SANTHA SHEELA, M.E.,Ph.D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.

DATE:

PLACE: Chennai

SIGNATURE OF THE 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 my thanks to **Dr. T. Sasikala M.E., Ph. D, Dean**, School of Computing, **Dr. L. Lakshmanan M.E., Ph.D.**, Head of the Department of Computer Science and Engineering for providing me 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. A. C. SANTHA SHEELA M.E., Ph.D**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-1 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were 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 98.01% success rate under the same impostor scenario.

Chapter No	TITLE		Page No.
	ABSTRACT		v
	LIST OF FIGURES		vii
1	INTRODUCTION		1
2	LITERATURE SURVEY		3
	2.1	Inferences from Literature Survey	5
	2.2	Open problems in Existing System	5
3	REQUIREMENTS ANALYSIS		6
	3.1	Software Requirements	6
	3.2	Hardware Requirements	6
	3.3	Technology Used	6
4	DESCRIPTION OF PROPOSED SYSTEM		7
	4.1	Selected Methodology or process model	7
	4.2	Architecture / Overall Design of Proposed System	11
	4.3	Description of Software for Implementation and Testing plan of the Proposed Model/System	12
	4.4	Project Management Plan	13
	4.5	Transition/ Software to Operations Plan (as applicable)	14
	CONCLUSION		15
	REFERENCES		16

LIST OF FIGURES

FIGURE NO	FIGURE NAME	Page No.
4.1	Architecture Diagram	11

LIST OF ABBREVIATIONS

JDK	Java Development Toolkit
DEX	Dalvik Executables
TCP	Transmission Control Protocol
IP	Internet Protocol
HTTP	Hyper Text Transfer Protocol
ADT	Android Development Tool

CHAPTER 1

INTRODUCTION

In an increasingly interconnected digital landscape, the importance of safeguarding our personal and sensitive information cannot be overstated. One of the most fundamental yet frequently overlooked aspects of cybersecurity is the strength and resilience of our passwords. Spyware attacks, a prevalent threat in today's cyber world, exploit vulnerabilities in our online defenses to gain unauthorized access to our data. To counter this menace, implementing effective password mechanisms is paramount. Spyware, a type of malicious software designed to infiltrate and compromise a user's system, often gains entry through weak or easily guessable passwords. In this introductory exploration, we delve into the critical role that robust password mechanisms play in fortifying our digital defenses against spyware attacks. As we proceed, we will unveil the key principles and strategies that individuals, organizations, and institutions should employ to create and maintain secure passwords. By adopting these practices, we can mitigate the risks associated with spyware and significantly enhance our online security posture.

In the ever-expanding realm of e-commerce, security is paramount. With the rise of online shopping, payment gateways, and digital transactions, ensuring the security of customer data, financial information, and personal details is of utmost importance for an e-commerce app. Spyware attacks, among other cyber threats, loom as a significant concern, making the establishment of efficient password mechanisms crucial in safeguarding the app and its users.

Efficient password mechanisms are the backbone of cybersecurity for e-commerce apps. They act as the first layer of defense against unauthorized access and are integral in protecting sensitive customer information, payment data, and personal details from prying eyes and malicious agents. This article explores strategies, techniques, and best practices that are tailored to e-commerce apps, helping to fortify the security of the platform and its users against the ever-present threat of spyware attacks.

E-commerce applications deal with a unique set of security challenges, given their handling of financial transactions and personal information. As such, it is vital to implement strong password creation guidelines, advising customers to create complex and unique passwords for their accounts. This means utilizing a combination of uppercase and lowercase letters, numbers, and special characters while avoiding easily guessable information. Promoting and enforcing these practices during the account creation process is essential.

In the ever-evolving world of cybersecurity, the demand for more secure and user-friendly authentication methods has given rise to innovative solutions. Among these, "Biotouchpass" handwritten passwords stand as a remarkable example of technology bridging the gap between security and user convenience. This cutting-edge biometric authentication technique harnesses the uniqueness of an individual's handwritten style, combining the art of penmanship with the science of security to create a highly personalized and effective means of safeguarding digital access.

In an era where traditional password methods face increasing vulnerabilities due to data breaches and phishing attacks, Biotouchpass introduces a compelling alternative. This biometric solution leverages the distinctive characteristics of a person's handwriting, such as stroke patterns, pressure, and pen angles, to establish a highly personalized digital signature. These characteristics are not easily replicated or forged, offering a level of security that transcends the limitations of traditional alphanumeric passwords.

The essence of Biotouchpass lies in the fusion of biometrics with user convenience. Traditional biometric methods, like fingerprints and facial recognition, are highly secure but often require specialized hardware or can be sensitive to environmental factors. Handwriting, on the other hand, is a universally accessible and natural form of expression, making Biotouchpass a highly adaptable solution for a wide range of applications, from personal devices to secure access in corporate and government settings.

CHAPTER 2

LITERATURE SURVEY

1. Title: "Update Strategies for Hmm-Based dynamic Signature Biometric Systems"

Author Name: Ruben tolosana, Ruben vera- rodriguez, et al.,

One of the most ideal and growing traits is handwriting signature as it was used for economic and valid agreements eventualities for over a century. The main intension of this work is to find out about system disposition update strategies of time function based systems such as hidden Markov model (HMM) and gaussian mixture fashions (GMM). Hence the two distinct instances have been recognised. Initially, the usual case of having an HMM based gadget with constant disposition (i.e., baseline system). Next, HMM based and GMM based systems whose disposition are optimized concerning the number of training signatures on hand to bring about the consumer template.

2. Title: "The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes"

Author name: Joseph Bonneau, Cormac herley, et al.,

The scope of scheme we review is also extensive, including parole management software, federated login protocols, graphical password schemes, cognitive authentication schemes, one-time passwords, hardware tokens, phone aided schemes, and biometrics. In particular, there is a large vary from schemes offering minor safety advantages passed legacy password, to those presenting remarkable safety benefits in return for being more steeply praised to deploy or more difficult to use. We conclude that many educational schemes have failed to reap traction, because researches not often reflect on consideration on a sufficiently both vary of real-world constraints.

3. Title: "Surveying the Development of Biometric User Authentication on Mobile Phones"

Author name: Weizhi meng, Duncan S. Wong, et.al.,

Designing dependable person authentication on mobile devices is turning into an increasingly vital venture to protect user's private statistics, and information. Since biometric strategies can give many benefits over the ordinary authentication methods, they have end up a huge topic for both academic and industry. The important purpose of biometric consumer authentication is to authenticate legitimate consumers, and discover imposters based on physiological, and behavioural characteristics. The current taxonomy of existing efforts concerning biometric authentication on cellular phones, and analyse their feasibility of deployment on touch enabled mobile devices.

4. Title: "Preprocessing and Feature Selection for Improved Sensor Interoperability in On-Line Biometric Signature Verification"

Author name: Ruben tolosana, Ruben vera- rodriguez, et al.,

Due to engineering science evolution, and the growing popularity of clever phones, humans can get right of entry to an application using authentication primarily based on biometric approaches from many exceptional devices. The scheme approach is based on two predominant stages. The first one is the pre-processing stage; the place records acquired from one-of-a-kind gadgets are processed in order to normalize the indicators in similar ranges. The next one is based on feature resolution taking into account; the system interoperability case, in order to select the choose elements which are sturdy in these condition.

2.1 INFERENCES FROM LITREATURE SURVEY

- We propose both Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) systems with a Siamese architecture. In addition, a bidirectional scheme (which is able to access both past and future context) is considered for both LSTM- and GRU-based systems. An exhaustive analysis of the system performance and also the time consumed during the training process for each recurrent Siamese network is carried out in order to compare the advantages and disadvantages for practical applications.
- Literature discusses the robustness of Bio Touch Pass systems against various attacks, such as forgery attempts, replay attacks, and brute-force attacks. It highlights the need for robust countermeasures and continuous monitoring.

2.2 OPEN PROBLEMS IN 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 year 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 in order 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.

CHAPTER 3 REQUIREMENT ANALYSIS

3.1 AIM

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

3.2 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 similar to 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.

3.3 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 in order 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 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.

3.4 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 29 that data through sonic waves to the user's mobile. Thus security mechanism for user drawing patterns in android application.

3.5 DESIGN AND IMPLEMENTATION CONSTRAINTS

3.5.1 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

3.5.2 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

3.5.3 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, 30 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.

3.6 OTHER NONFUNCTIONAL REQUIREMENTS

3.6.1 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.

3.6.2 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

DESCRIPTION OF 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 similar to 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.

- 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.

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

- Core java, Android, PHP

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.

4.1.1 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.

4.1.1.1 Introduction to Java

Java has been around since 1991, developed by a small team of Sun Microsystems developers in a project originally called the Green project. The intent of the project was to develop a platform-independent software technology that would be used in the consumer electronics industry. 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 in order 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 16 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.

4.1.1.2 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 (similar to 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 actually 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

4.1.1.3 The Java Programming Language

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

- Simple
- Object-oriented
- Distributed
- Interpreted
- Robust
- Secure
- Architecture-neutral
- Portable
- High-performance
- Multithreaded 18
- Dynamic

The code and 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.

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.

4.1.1.4 THE JAVA PLATFORM

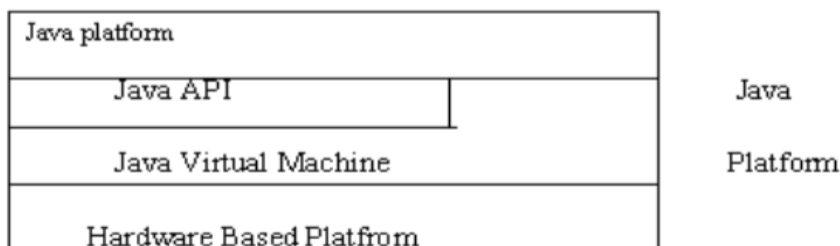
A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

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, weel-tuned interpreters, and just-in-time byte compilers can bring Java's performance close to that of native code without threatening portability.

4.1.2 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.

4.1.2.1 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 22 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.

4.1.2.2 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

4.1.2.3 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.

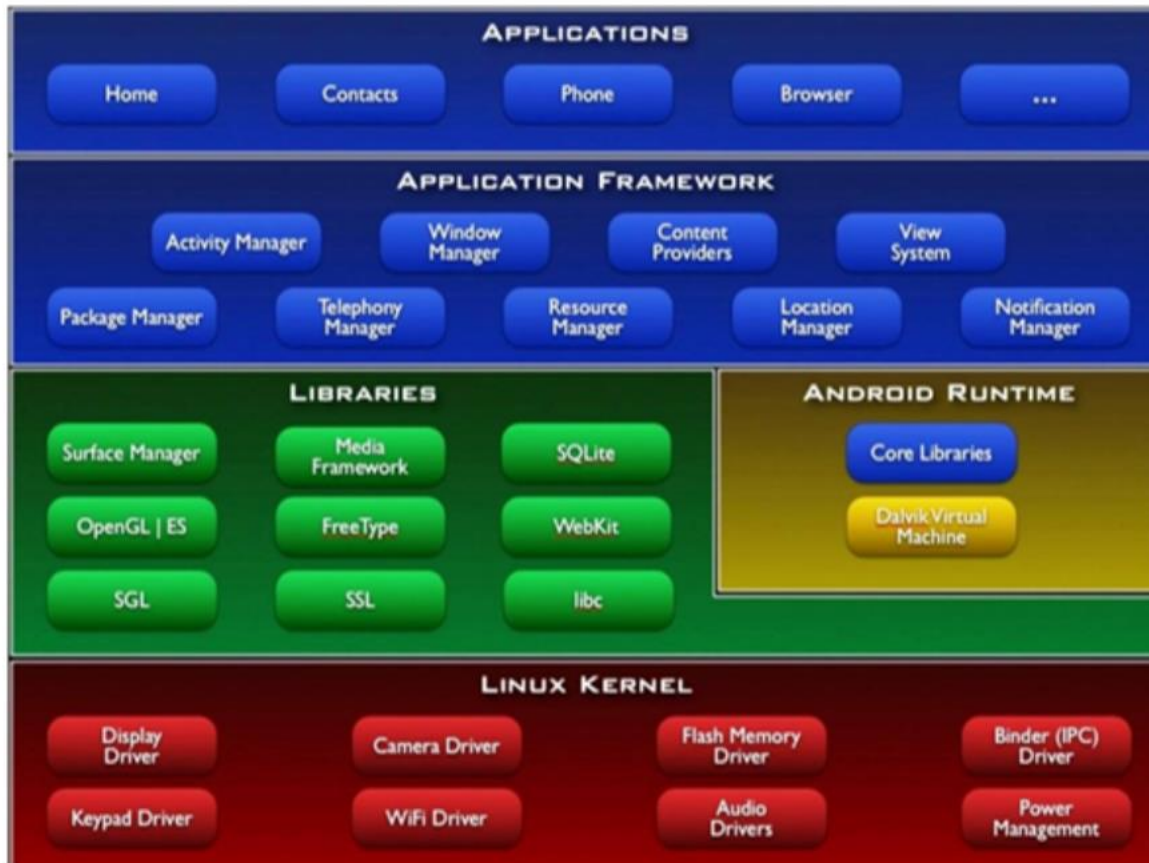
4.1.3 Android Introduction

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 mini computers. 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 jelly bean.

The android is a operating system and is a stack of software components which is divided into five sections and four main layers that is



4.1.3.1 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 really 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 hard ware devices on the system.

4.1.3.2 Libraries:

On top of a Linux kernel there is a set of libraries including open source web browser such as webkit, 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.

4.1.3.3 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.

4.1.3.4 Application frame work:

The application frame work 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.

4.1.3.5 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
- Its supports all Google services
- The operating system is able to 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

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

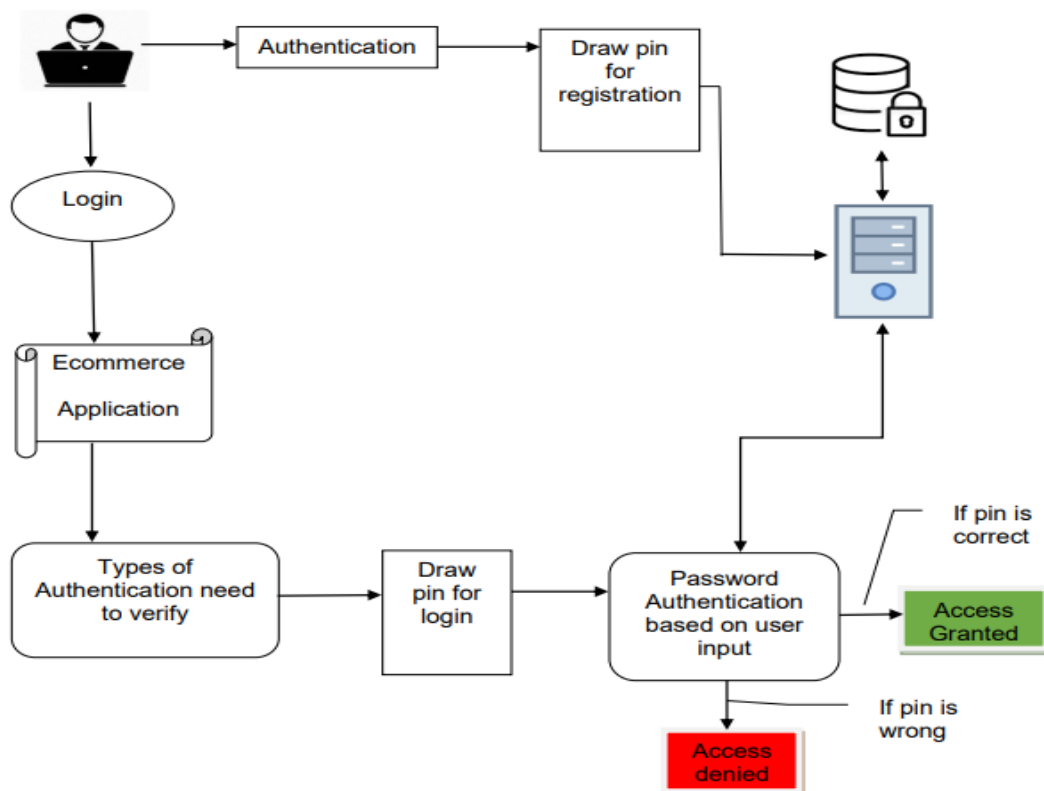


Figure 4.1 Architecture diagram

4.3 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.

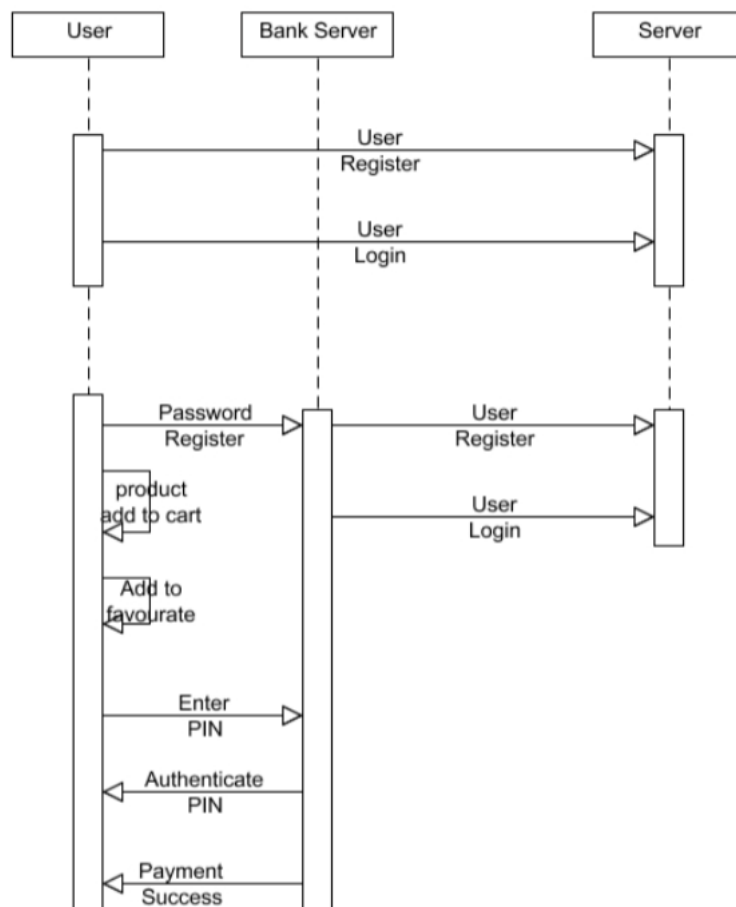


Figure 4.2 Sequence diagram

4.4 COLABORATION 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.

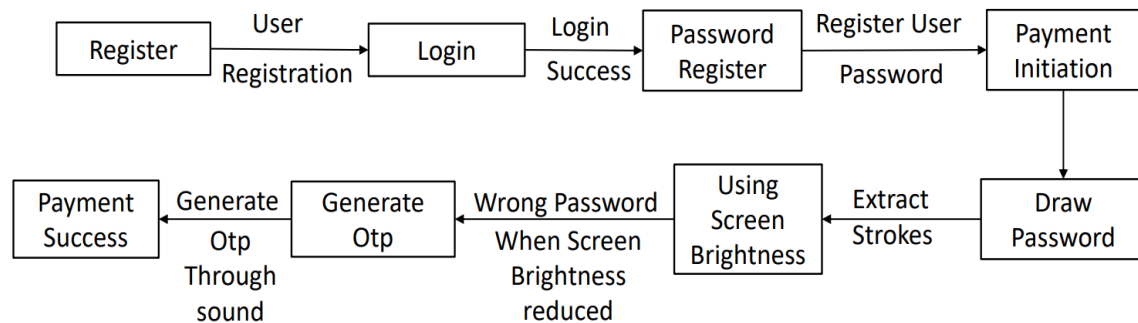


FIGURE 4.3 Colaboration Diagram

4.5 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.

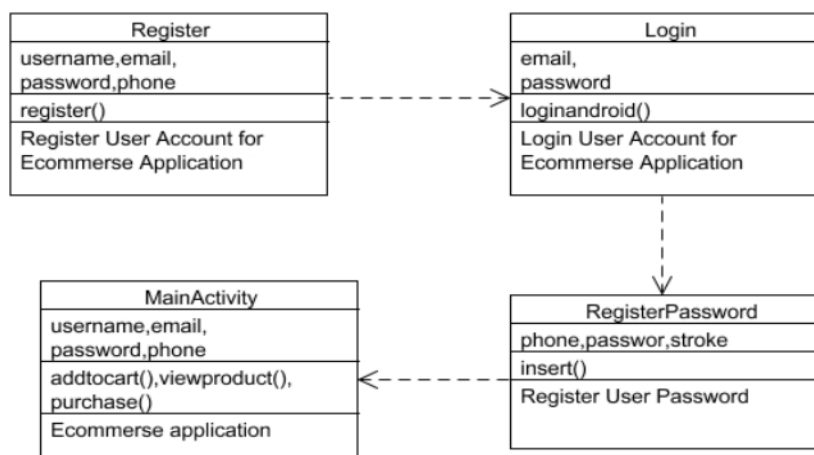


FIGURE 4.4 Class Diagram

4.6 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

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

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 Hand Written 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 has to 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 has to register their four digit password with multiple strokes during their

registration process once the process completed during confirm password .User has to 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 in order 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.

4.7 PROJECT MANAGEMENT PLAN

July: Project Planning and Requirement Gathering

August: Environment Setup (Docker container, LLM, vector database)

September: Document Ingestion and Preprocessing

October: RAG Implementation, Utility tool script development

November: Web Interface Development

December: Fine-tuning and Testing

January: Documentation and Presentation

February: Full working demo

4.8 SOFTWARE TO OPERATIONS PLAN

OCR stands for Optical Character Recognition. It's a technology used to convert different types of documents, such as scanned paper documents, PDF files, or images captured by a digital camera, into editable and searchable data. OCR software or systems analyze the shapes and patterns in the input data and then translate it into text that can be edited, searched, and stored electronically. This technology is particularly useful for digitizing printed or handwritten documents, as it eliminates the need for manual transcription and allows for the efficient extraction of text and data from images. OCR technology has a wide range of applications, including:

Document Digitization: OCR is widely used to convert printed documents, such as books, articles, and historic records, into digital formats.

Data Extraction: In business settings, OCR is used to extract data from invoices, receipts, and forms, making it easier to process and analyze this information.

Text Searching: By converting images with text into searchable text, OCR enables users to search for specific words or phrases within documents.

Accessibility: OCR technology is used to convert printed material into accessible formats for individuals with visual impairments.

Translation: OCR can be used to recognize text in one language and then translate it into another language.

Data Entry Automation: In data entry tasks, OCR reduces manual data entry efforts, thus saving time and reducing errors.

License Plate Recognition: OCR is employed in license plate recognition systems for parking enforcement and security applications.

Business Cards: OCR can be used to extract contact information from business cards and input it into digital contact management systems.

Handwriting Recognition: Some OCR systems are capable of recognizing handwritten text, enabling the conversion of handwritten notes into digital text.

OCR technology has advanced significantly in recent years, with improved accuracy and the ability to handle a wide range of fonts, languages, and document layouts. It

plays a crucial role in digitizing and managing vast amounts of textual data, making it more accessible and useful in various industries and applications.

CODING AND TESTING

7.1 CODING

Once the design aspect of the system is finalized, the system enters into 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 a given programming language. Therefore, a good coding style has to be taken whenever changes are required to easily screw into the system.

7.2 CODING STANDARDS

Coding standards are guidelines to programming that focus on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually 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 standards 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

7.2.1 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.

7.2.2 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.

7.2.3 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.

7.2.4 MESSAGE BOX FORMAT

When something has to 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.

7.3 TEST PROCEDURE

7.3.1 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, as a whole 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.

7.4 TEST DATA AND OUTPUT

7.4.1 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.

7.4.2 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

7.4.3 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.

7.4.4 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 manner in which a programmer in which a program unit breaks.

7.4.5 STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing particular 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 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

7.4.6 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.

7.5 TESTING TECHNIQUES / TESTING STRATEGIES

7.5.1 TESTING

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

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 an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error. Any engineering product can be tested in one of the two ways:

7.5.1.1 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 design 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

7.5.1.2 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

7.5.2 SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance 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.

7.5.2.1 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.

7.5.2.2 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-off-range items and invalid combinations. Since the compiler s 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 a errors in the program.

7.5.2.3 SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built in to 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.

7.5.2.4 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

7.5.2.5 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.

CONCLUSION:-

In conclusion, establishing robust password mechanisms is imperative for countering spyware attacks and safeguarding digital assets. A multi-faceted strategy, encompassing strong password policies, regular updates, and user education, is crucial. Integrating two-factor or multi-factor authentication adds an additional layer of defense, rendering compromised passwords insufficient for unauthorized access. Continuous monitoring, software updates, and collaborative efforts within the cyber security community are essential for staying ahead of evolving threats. Ultimately, a proactive and comprehensive approach to password security is paramount in mitigating the risks posed by spyware and fortifying overall digital resilience.

REFERENCES:-

- [1] M. Salehan and A. Negahban, "Social Networking on Smartphones: When Mobile Phones Become Addictive," *Computers in Human Behavior*.
- [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*.
- [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*
- [4] W. Meng, D. Wong, S. Furnell, and J. Zhou, "Surveying the Development of Biometric User Authentication on Mobile Phones," *IEEE Communications Surveys Tutorial*.
- [5] J. Angulo and E. Wastlund, "Exploring Touch-Screen Biometrics for User Identification on Smart Phones," J. Camenisch, B. Crispo, S. Fischer-Hbner.
- [6] P. Lacharme and C. Rosenberger, "Synchronous One Time Biometrics with Pattern Based Authentication," in *Proc. 11th Int. Conf. on Availability, Reliability and Security, ARES*.
- [7] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega- Garcia, "Benchmarking Desktop and Mobile Handwriting across COTS Devices: the e-Incorporating Touch Biometrics to Mobile One-Time Passwords: Exploration of Digits," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*.
- [8] 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*.
- [9] N. Sae-Bae and N. Memon, "Online Signature Verification on Mobile Devices," *IEEE Transactions on Information Forensics and Security*.
- [10] "Improving Accuracy, Applicability and Usability of Keystroke Biometrics on Mobile Touchscreen Devices," in *Proc. of the CHI Conference on Human Factors in Computing Systems*.
- [11] 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.

- [12] 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.
- [13] 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.
- [14] Q. Yue, Z. Ling, X. Fu, B. Liu, W. Yu, and W. Zhao, "My Google Glass Sees Your Passwords!" in *Black Hat USA*, 2014.
- [15] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, and J. Ortega-Garcia, "Incorporating Touch Biometrics to Mobile One-Time Passwords: Exploration of Digits," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018.
- [16] 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.
- [17] 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.
- [18] 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.
- [19] 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.
- [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] 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.

- [22] 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.
- [23] L. Li, X. Zhao, and G. Xue, "Unobservable Reauthentication for Smartphones," in *Proc. 20th Network and Distributed System Security Symposium, NDSS*, 2013.
- [24] 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.
- [25] 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.
- [26] 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.

Source code:

```
<?xml version="1.0" encoding="UTF-8"?>
<projectDescription>
  <name>BiotouchPass</name>
  <comment></comment>
  <projects>
  </projects>
  <buildSpec>
    <buildCommand>

      <name>com.genuitec.eclipse.j2eedt.core.WebClasspathBuild
er</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>
        <name>org.eclipse.jdt.core.javabuilder</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>

      <name>com.genuitec.eclipse.j2eedt.core.J2EEProjectValidato
r</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>

      <name>com.genuitec.eclipse.j2eedt.core.DeploymentDescript
orValidator</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>

      <name>org.eclipse.wst.validation.validationbuilder</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>
```

```

    <name>com.genuitec.eclipse.ast.deploy.core.DeploymentBuilder</name>
        <arguments>
        </arguments>
    </buildCommand>
</buildSpec>
<natures>

    <nature>com.genuitec.eclipse.ast.deploy.core.deploymentnature</nature>

    <nature>com.genuitec.eclipse.j2eedt.core.webnature</nature>
    <
        <nature>org.eclipse.jdt.core.javanature</nature>
        <nature>org.eclipse.wst.jsdt.core.jsNature</nature>
    </natures>
</projectDescription>

```

```

<?xml version="1.0" encoding="UTF-8"?>

```

```

<projectDescription>
    <name>BiotouchPassEcomm</name>
    <comment></comment>
    <projects>
    </projects>
    <buildSpec>
        <buildCommand>

```

```

            <name>com.genuitec.eclipse.j2eedt.core.WebClasspathBuilder</name>
                <arguments>
                </arguments>
            </buildCommand>
            <buildCommand>
                <name>org.eclipse.jdt.core.javabuilder</name>
                <arguments>
                </arguments>
            </buildCommand>
            <buildCommand>

```

```

            <name>com.genuitec.eclipse.j2eedt.core.J2EEProjectValidator</name>

```

```

        <arguments>
        </arguments>
    </buildCommand>
    <buildCommand>

<name>com.genuitec.eclipse.j2eedt.core.DeploymentDescriptorValidator</name>
        <arguments>
        </arguments>
    </buildCommand>
    <buildCommand>

<name>org.eclipse.wst.validation.validationbuilder</name>
        <arguments>
        </arguments>
    </buildCommand>
    <buildCommand>

<name>com.genuitec.eclipse.ast.deploy.core.DeploymentBuilder</name>
        <arguments>
        </arguments>
    </buildCommand>
</buildSpec>
<natures>

<nature>com.genuitec.eclipse.ast.deploy.core.deploymentnature</nature>

<nature>com.genuitec.eclipse.j2eedt.core.webnature</nature>
>
    <nature>org.eclipse.jdt.core.javanature</nature>
    <nature>org.eclipse.wst.jsdt.core.jsNature</nature>
</natures>
</projectDescription>

```

```
package com.example.handwrittenpassword.cache;
```

```

/**
 * Created by 06peng on 2015-07-03.
 */

```

```

import android.content.Context;

import java.io.File;

public class FileCache {

    private File cacheDir;

    public FileCache(Context context) {
        //Find the dir to save cached images
        if
(android.os.Environment.getExternalStorageState().equals(an
droid.os.Environment.MEDIA_MOUNTED)) {
            cacheDir = new
File(android.os.Environment.getExternalStorageDirectory(),
"fresco_cache");
        } else {
            cacheDir = context.getCacheDir();
        }
        if (!cacheDir.exists()) {
            cacheDir.mkdirs();
        }
    }

    public File getCacheDir() {
        return cacheDir;
    }

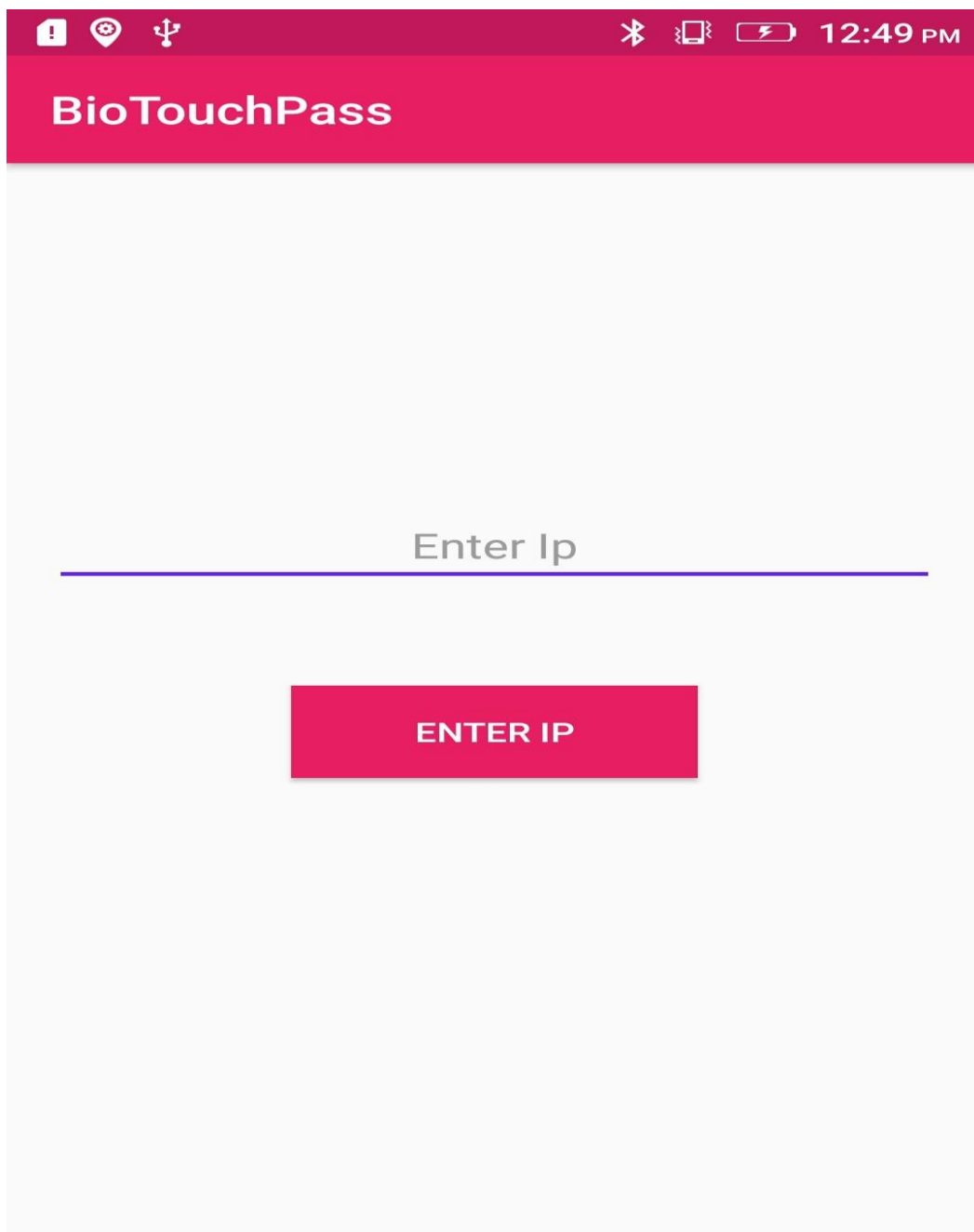
    public File getFile(String url) {
        String filename = String.valueOf(url.hashCode());
        File f = new File(cacheDir, filename);
        if (f.exists()) {
            return f;
        }
        return null;
    }

    public void clear() {
        File[] files = cacheDir.listFiles();
        if (files == null) {
            return;
        }
    }
}

```

```
    }  
    for (File f : files) {  
        f.delete();  
    }  
}  
}
```

SCREENSHOTS:





BioTouchPass

 Email

 Password

Login

New User? Register



BioTouchPass

 Name

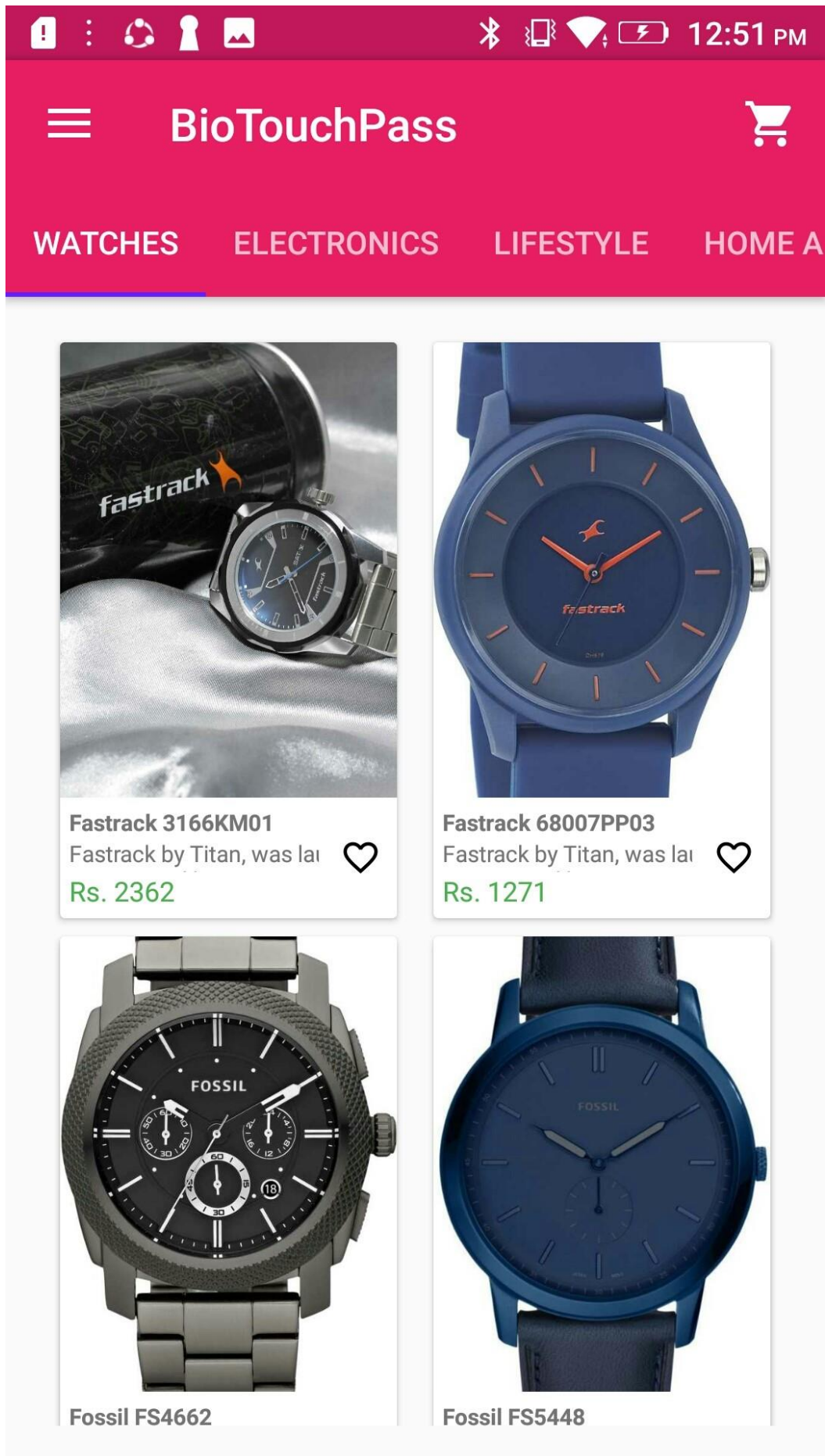
 Phone number

 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

BioTouchPass

Bank Detail

Name : TextView

Email : TextView

Account No : TextView

Amount : TextView

GENERATE UPI ID





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



