

**EFFICIENT PASSWORD MECHANISM TO OVERCOME SPYWARE ATTACKS**

**A PROJECT REPORT**

***Submitted by***

**NIVETHITHA.R [REGISTER NO:211417104174]**

**VARSHA.M [REGISTER NO:211417104291]**

***in the partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**PANIMALAR ENGINEERING COLLEGE,CHENNAI 600123**

**ANNA UNIVERSITY:CHENNAI 600025**

**APRIL 2021**

**BONAFIDE CERTIFICATE**

Certified that this project report **“EFFICIENT PASSWORD MECHANISM TO OVERCOME SPYWARE ATTACKS”** is the bonafide work of “**NIVETHITHA.R(REGNO:211417104174)&VARSHA.M(REGNO:211417104291)”**who carried out the project work under my supervision.

**SIGNATURE SIGNATURE**

**Dr.S.MURUGAVALLI,M.E.,Ph.D., Mrs.SANGEETHA KRISHNAN,M.E**

**HEAD OF THE DEPARTMENT ASSISTANT PROFESSOR**

DEPARTMENT OF CSE, DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, PANIMALAR ENGINEERING COLLEGE,

NASARATHPETTAI, NASARATHPETTAI,

POONAMALLEE, POONAMALLEE,

CHENNAI-600 123. CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on...........................

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYA RAJESWARI**, **Thiru.C.SAKTHIKUMAR,M.E.,** and **Tmt. SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Phd** for the support extended throughout the project.

We would like to thank our **Project Guide Mrs SANGEETHAKRISHNAN ,M.E.** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

**NIVETHITHA.R[211417104174]**

**VARSHA.M[211417104291]**

**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 le0vel 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** |
|  | LIST OF FIGURES |  |
|  | LIST OF ABBREVIATIONS |  |
| **1** | **INTRODUCTION** |  |
| 1.1 | Synopsis |  |
| **2** | **SYSTEM ANALYSIS** |  |
| 2.1 | Existing system |  |
| 2.2 | Proposed system |  |
| **3** | **REQUIREMENTS SPECIFICATION** |  |
| 3.1 | Introduction |  |
| 3.2 | Hardware and Software specification |  |
| 3.3 | Technologies Used |  |
| 3.3.1 | Java |  |
| 3.3.1.1 | Introduction to Java |  |
| 3.3.1.2 | Working of Java |  |
| 3.3.1.3 | The Java Programming Language |  |
| 3.3.1.4 | The Java Platform |  |
| 3.3.2 | Apache Tomcat Server |  |
| 3.3.3 | Android Introduction |  |
| **4** | **Project Purpose and Scope** |  |
| 4.1 | Purpose |  |
| 4.2 | Project Scope |  |
| 4.3 | Product Perspective |  |
| 4.4 | System Features |  |
| 4.5 | Design and Implementation Constraints |  |
| 4.6 | Other Nonfunctional Requirements |  |
| **5** | **SYSTEM DESIGN** |  |
| 5.1 | Architecture Diagram |  |
| **6** | **SYSTEM DESIGN – DETAILED** |  |
| 6.1 | Modules |  |
| 6.2 | Module Description |  |
| **7** | **CODING AND TESTING** |  |
| 7.1 | Coding |  |
| 7.2 | Coding standards |  |
| 7.3 | Test procedure |  |
| 7.4 | Test data and output |  |
|  | **SOURCE CODE** |  |
|  | **SNAP SHOTS** |  |
|  | **REFERENCES** |  |

**LIST OF FIGURES**

5.1 System Design

5.2 Sequence Diagram

5.3 Use Case Diagram

5.4 Activity Diagram

5.5 Collaboration Diagram:

5.6 Data Flow Diagram

5.7 Class Diagram

**LIST OF ABBREVIATIONS**

**JDK**  Java Development Toolkit

**DEX** Dalvik Executables

**TCP** Transmission Control Protocol

**IP**  Internet Protocol

**HTTP** HyperText Transfer Protocol

**ADT** Android Development Tool

**CHAPTER 1**

**INTRODUCTION**

**Aim:**

The main aim of this project is to develop user-friendly mobile applications ensuring data protection and high security.

**1.1 Synopsis:**

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.

**CHAPTER 2**

**SYSTEM ANALYSIS**

**2.1 EXISTING SYSTEM**

In the 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 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.

This effect has proven to be very important for many behavioral biometric traits such as the case of the handwritten signature.

**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 about the user as possible.

**2.2 PROPOSED SYSTEM**

Our proposed system focuses on providing user-friendly mobile applications ensuring data protection and high security. Users 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 the pin entered during the registration process.

Our second stage of authentication involves multiple options based on user preference where users can set multiple sets of combinations. Users can set a 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.

**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 3**

**REQUIREMENTS SPECIFICATION**

**3.1 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 is in charge of 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 are able to 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 in order to complete the payment. Our proposed approach enhances such a 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 in order 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 make the assumption 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.

**3.2 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
* Apache Tomcat
* Android Studio 3.4
* Android Phone

**3.3 TECHNOLOGIES USED**

* Core java, Android, PHP

**3.3.1 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.

**3.3.1.1 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 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.

**3.3.1.2 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

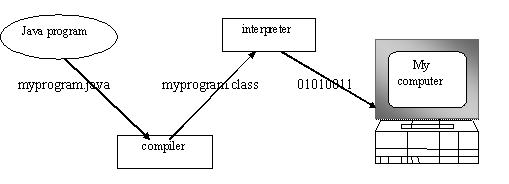
**3.3.1.3 THE JAVA PROGRAMMING LANGUAGE**

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
  + 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.3.1**

You can think of Java bytecodes 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.

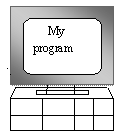
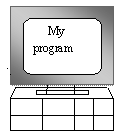
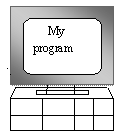
You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM. For example, that same Java program can e run on Windows NT, Solaris and Macintosh









**  **

**PC-Compatible Sun Ultra Solaris Power macintosh**

**Windows NT System 8**

**3.3.1.4 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.

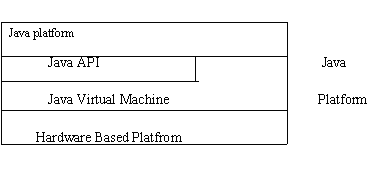
****

Fig.3.3

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

**3.3.2 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 and 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 paring

**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.

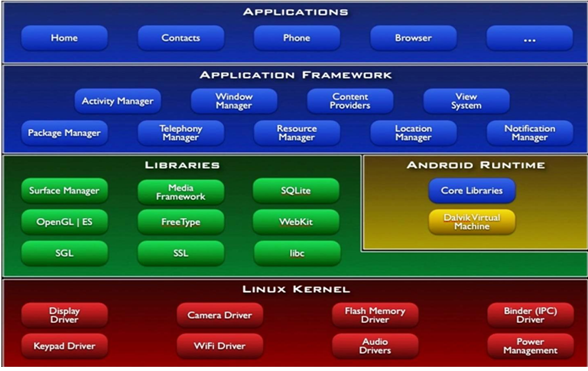
### 3.3.3 Android Introduction:

Android is a Linux based operating system designed primarily for touch screen mobile devices such as smartphones and tablet computers. The operating system has developed a lot in the last 15 years starting from black and white phones to recent smartphones 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 a 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 the ARM architecture platform. The android is an open source operating system means that it’s free and anyone can use it. The android has got millions of apps available that can help you manage your life one or other way and it is available low cost in the market at that time and android is very popular.

Android development supports 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 the 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



### Linux kernel:

The android uses the powerful Linux kernel and it supports a 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. 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 hardware devices on the system.

### Libraries:

On top of a Linux kernel there is a set of libraries including open source web browsers such as webkit, library libc. These libraries are used to play and record audio and video. The SQLite is a database which is useful for storage and sharing of application data. The SSL libraries are responsible for internet security etc.

### 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.

### 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.

### Applications:

You will find all [the android applications](https://www.edgefxkits.com/home-automation-by-android-application-based-remote-control?utm_source=elprocus.com) 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 USB 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

**CHAPTER 4**

**4.1 Project Purpose and Scope**

**4.1 Purpose**

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

~~.~~**4.2 Project Scope**

Our proposed system focuses on providing user-friendly mobile applications ensuring data protection and high security. Users 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 the pin entered during the registration process.

Our second stage of authentication involves multiple options based on user preference where users can set multiple sets of combinations. Users can set a 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.

**4.3 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 drew in the mobile , all the numbers patterns will be stored in the backend. So whenever a user signs in to the e-commerce application , this password security mechanism will be enacted with every product purchase.

* 1. **4.4 SystemFeatures**

Here we concentrate on password mechanisms with user drawing patterns. The user needs to register their pin number in the application with proper strokes. All the pin numbers with proper strokes will be getting stored in the 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 sends that data through sonic waves to the user’s mobile. Thus security mechanism for user drawing patterns in android application.

**4.5 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.

* 1. **Other Nonfunctional Requirements**

**Performance Requirements**

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

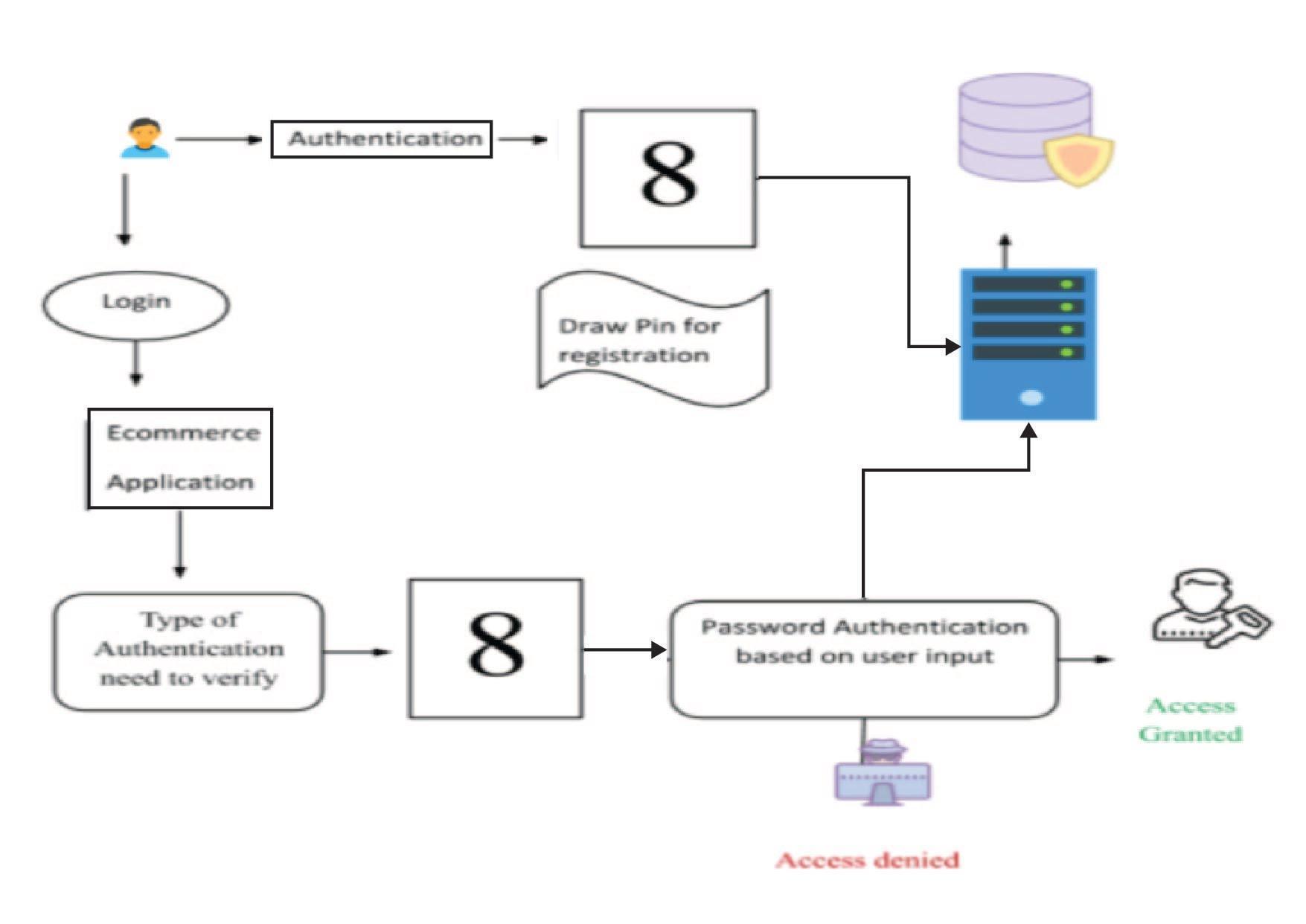
* embedded browser in charge of the navigation and accessing to the web service;
* Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.
* Web Server, Security Module, Server-Side Capturing Engine, Preprocessing 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 the 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.

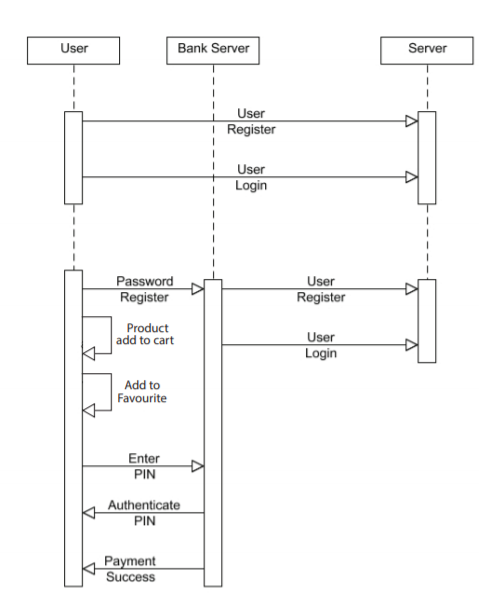
**CHAPTER 5**

**5.1 Architecture Diagram:**

**** **Fig: 5.1**

**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 sometimes called event diagrams, event sceneries and timing diagrams.

****

**UML Diagram:**

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

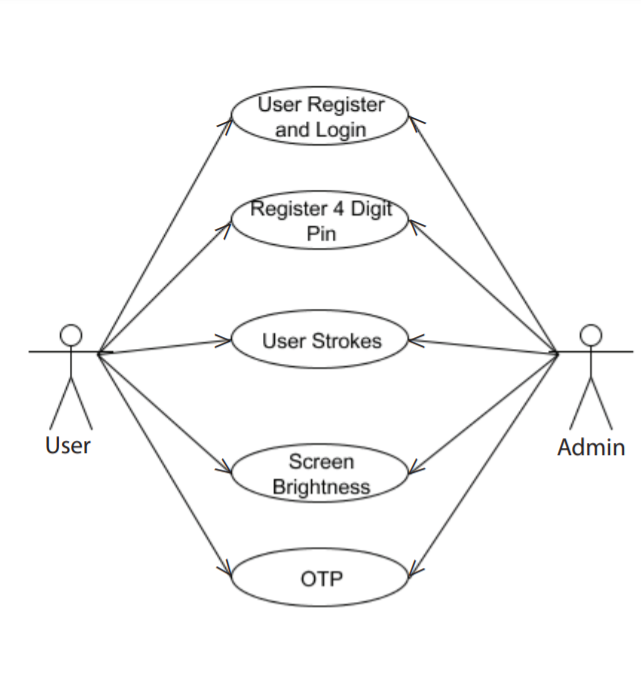
**USE CASE DIAGRAM**

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 provides 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.

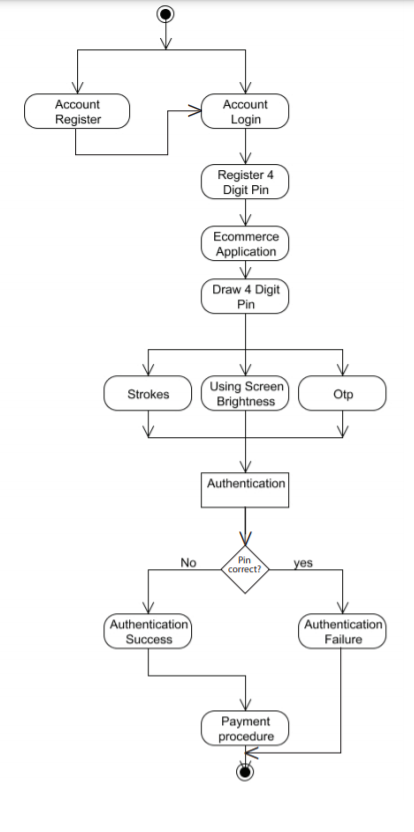
****

**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.

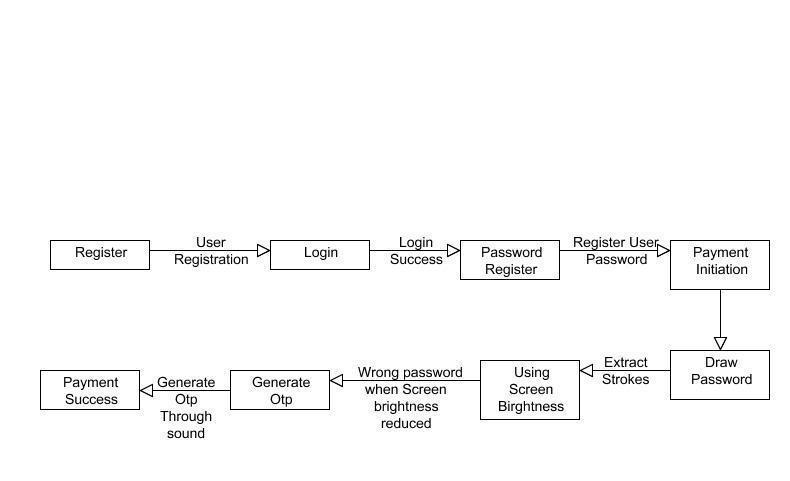
The most important shape types:

* 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.

****

**Collaboration Diagram:**

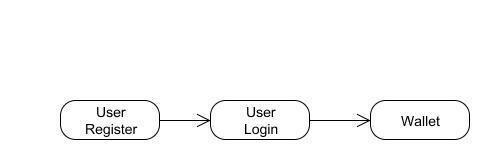
UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain models to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

****

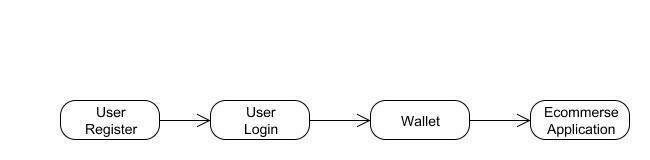
**DATA FLOW DIAGRAM:**

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.

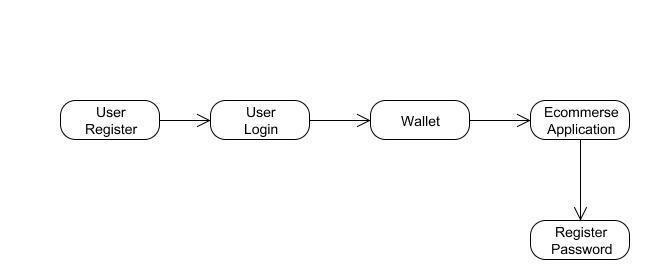
**Level 0:**

****

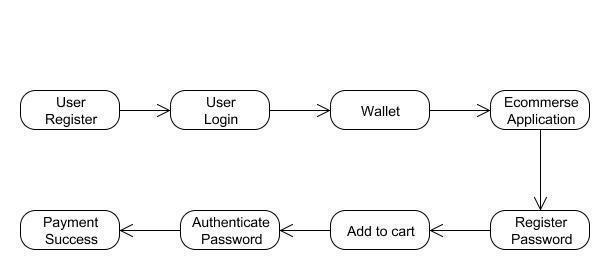
**Level 1:**

****

**Level 2:**

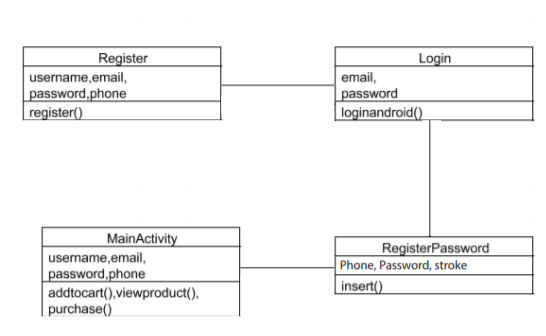
****

**Level 3:**

****

**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.



**CHAPTER 6**

**SYSTEM DESIGN**

Users 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 the pin entered during the registration process.

Our second stage of authentication involves multiple options based on user preference where users can set multiple sets of combinations. Users can set a 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.

**6.1 MODULES**

### User Registration

### Password Creation Using Strokes

### Product Selection And Payment Using Handwritten Password

### Password Analysing

### 6.2 Modules Description:

### User Registration

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 the user can view a list of products in their page multiple lists of products and their details.

### Password Creation Using Strokes

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

### Product Selection And Payment Using Handwritten Password

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

### Password Analysing

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 are inserted while the screen brightness is low and apply the HMac algorithm for the PIN given by the 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 the user generated PIN and generates the signature value for the Original PIN and compares two signatures. If the two Signatures are equal the user can access the Profile of the user. If not, the user can’t access the profile.

**CHAPTER 7**

**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 and easily screwed into the system.

**7.2 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 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 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 letters and have mixed cases.

**Member Function and Data Member name**

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

**VALUE CONVENTIONS**

Value conventions ensure values for variables 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 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 an illegal operation.
* ! – Information to the user.

**7.3 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 be 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**

**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 a 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.

**STRUCTURAL TEST**

Structural 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 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 constructing 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**

**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 consisting of several key activities and steps for running a 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 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 has 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:

**WHITE BOX TESTING**

This testing is also called Glass box testing. In this testing, by knowing the specific functions that a product has been designed to perform, tests 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
* Cyclomatic complexity
* Deriving test cases
* Graph matrices Control

**BLACK BOX TESTING**

In this testing by knowing the internal operation of a product, a 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 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.

**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 violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax errors. 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 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 the condition test is to deduct not only errors in the condition of a program but also other 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 an individual who desires to penetrate the 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 a manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirements. After the 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 a key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective systems and users at the time of developing and making changes whenever required. This is done in regard to the following points.

* Input screen design.
* Output screen design.

**TEST CASE 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED**  **OUTPUT** | **ACTUAL**  **OUTPUT** | **TEST**  **RESULT** | **TEST**  **COMMENTS** |
| 1. | Login | username,password | homepage | homepage | pass | authorized user is logged in system |

**TEST CASE 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED**  **OUTPUT** | **ACTUAL**  **OUTPUT** | **TEST**  **RESULT** | **TEST COMMENTS** |
| 2. | User Registration | E-mail ID,  Password | New user details are added | New user details are added | Pass | Details of new user is added in database |

**TEST CASE 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED**  **OUTPUT** | **ACTUAL**  **OUTPUT** | **TEST**  **RESULT** | **TEST COMMENTS** |
| 3. | Password registration | Drawing 4-digit pin | Pin successfully registered | Pin  successfully registered | pass | 4-digit pin will be stored in the database |

**TEST CASE 4**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED**  **OUTPUT** | **ACTUAL**  **OUTPUT** | **TEST**  **RESULT** | **TEST COMMENTS** |
| 4. | Password  Analysing | 4-digit pin | correct | correct | pass | The user draws the correct pin and the payment is done successfully. |

**Source Code**

**filedao**

package logic;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.List;

public class FileDao

{

public static Connection getConnection()

{

Connection con=null;

try

{

Class.forName("com.mysql.jdbc.Driver"); con=DriverManager.getConnection("jdbc:mysql://localhost/biotouchpass", "root", "root");

//System.out.println("------------CLOUD USER CREATED------------");

}

catch(Exception e){System.out.println(e);}

return con;

}

public static boolean ownerCheck(OwnerPojo op) throws SQLException

{

boolean emailexists=false;

Connection con=FileDao.getConnection();

String query="select \* from users where email = ?";

PreparedStatement st = con.prepareStatement(query);

st.setString(1, op.getEmail());

ResultSet rs=st.executeQuery();

if(rs.next())

{

emailexists = true;

}

return emailexists;

}

public static int ownerSave(OwnerPojo op){

int status=0;

try

{

Connection con=FileDao.getConnection();

String query="insert into users(username,email,password,phone) values (?,?,?,?)";

PreparedStatement ps=con.prepareStatement(query);

ps.setString(1, op.getUsername());

ps.setString(2, op.getEmail());

ps.setString(3, op.getPassword());

ps.setString(4, op.getPhone());

status=ps.executeUpdate();

con.close();

}

catch(Exception ex){ex.printStackTrace();

}

return status;

}

public static String ownerLogin(OwnerPojo olog)

{

String status="";

try

{

Connection con=FileDao.getConnection();

String email="";

String password="";

String name="";

String query="select \* from users where email='"+olog.getEmail()+"'";

PreparedStatement st = con.prepareStatement(query);

ResultSet rs=st.executeQuery();

while(rs.next())

{

// name=rs.getString("name");

email=rs.getString("email");

password=rs.getString("password");

// phone=rs.getString("password");

System.out.println(email+" "+password);

System.out.println(olog.getEmail()+" "+olog.getPassword());

}

if (email.equals(olog.getEmail())&& password.equals(olog.getPassword()))

{

status=name;

}

else

{

status="no";

}

}

catch (Exception e)

{

e.printStackTrace();

}

return status;

}

public static String LoginDetails(String email) throws SQLException

{

// System.err.println("fdfdfsdf");

String username="";

String uemail="";

String password="";

String phone="";

Connection con=FileDao.getConnection();

String query="select \* from users where email =?";

PreparedStatement st = con.prepareStatement(query);

st.setString(1, email);

ResultSet rs=st.executeQuery();

if(rs.next()) {

username=rs.getString("username");

uemail=rs.getString("email");

password=rs.getString("password");

phone=rs.getString("phone");

}

return username+","+uemail+","+password+","+phone;

}

}

**loginandroid**

package logic;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.json.simple.JSONObject;

public class LoginAndroid extends HttpServlet

{

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String email=request.getParameter("username");

String password=request.getParameter("password");

String[] detail;

JSONObject jsonObject = new JSONObject();

JSONObject jsonObject2 = new JSONObject();

OwnerPojo olog=new OwnerPojo();

olog.setEmail(email);

olog.setPassword(password);

try

{

String status=FileDao.ownerLogin(olog);

if(!status.equals("no"))

{

String userdetails=FileDao.LoginDetails(email);

detail = userdetails.split(",");

jsonObject2.put("username", detail[0]);

jsonObject2.put("email", detail[1]);

jsonObject2.put("password", detail[2]);

jsonObject2.put("phone", detail[3]);

jsonObject.put("error", "false");

jsonObject.put("message", "Login Successfull");

jsonObject.put("user", jsonObject2);

}

else

{

jsonObject.put("error", "true");

jsonObject.put("message", "Login failed");

}

out.print(jsonObject);

}

catch (Exception e)

{

// TODO: handle exception

}

}

}

**ownerpojo**

package logic;

public class OwnerPojo

{

private String username;

private String email;

private String password;

private String phone;

public String getUsername()

{

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public String getPassword()

{

return password;

}

public void setPassword(String password)

{

this.password = password;

}

public String getPhone()

{

return phone;

}

public void setPhone(String phone)

{

this.phone = phone;

}

}

**registerandroid**

package logic;

import java.io.IOException;

import java.io.PrintWriter;

import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.json.simple.JSONObject;

public class RegisterAndroid extends HttpServlet

{

public void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException

{

response.setContentType("text/html");

PrintWriter out = response.getWriter();

String username = request.getParameter("username");

String email = request.getParameter("email");

String password = request.getParameter("password");

String phone = request.getParameter("phone");

String[] detail;

JSONObject jsonObject = new JSONObject();

JSONObject jsonObject2 = new JSONObject();

OwnerPojo op=new OwnerPojo();

op.setUsername(username);

op.setEmail(email);

op.setPassword(password);

op.setPhone(phone);

try {

boolean emailexists=FileDao.ownerCheck(op);

if(emailexists==true)

{

jsonObject.put("error", "false");

jsonObject.put("message", "email already exist");

}else{

int status=FileDao.ownerSave(op);

if(status>0){

String userdetails=FileDao.LoginDetails(email);

detail = userdetails.split(",");

jsonObject2.put("username", detail[0]);

jsonObject2.put("email", detail[1]);

jsonObject2.put("password", detail[2]);

jsonObject2.put("phone", detail[3]);

jsonObject.put("error", "false");

jsonObject.put("message", "Registration Successful");

jsonObject.put("user", jsonObject2);

}else {

jsonObject.put("error", "true");

jsonObject.put("message", "Registration failed");

}

}

out.print(jsonObject);

}catch (Exception e)

{ // TODO: handle exception }

}

**SQL**

**bankadmindb**

-- phpMyAdmin SQL Dump

-- version 4.7.0

-- https://www.phpmyadmin.net/

--

-- Host: 127.0.0.1

-- Generation Time: Oct 23, 2019 at 09:19 AM

-- Server version: 10.1.26-MariaDB

-- PHP Version: 7.1.8

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET AUTOCOMMIT = 0;

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `bankadmindb`

--

-- --------------------------------------------------------

--

-- Table structure for table `password`

--

CREATE TABLE `password` (

`id` int(11) NOT NULL,

`phone` varchar(100) NOT NULL,

`password` varchar(100) NOT NULL,

`stroke` varchar(100) NOT NULL,

`upiid` varchar(100) NOT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `password`

--

INSERT INTO `password` (`id`, `phone`, `password`, `stroke`, `upiid`) VALUES

(5, '9840512532', '2@2@6@5', '1@1@1@1', '9840512532@okbank');

-- --------------------------------------------------------

--

-- Table structure for table `tokengeneration`

--

CREATE TABLE `tokengeneration` (

`accountno` varchar(255) NOT NULL DEFAULT '',

`accholdername` varchar(255) DEFAULT NULL,

`time` varchar(255) DEFAULT NULL,

`productamt` varchar(255) DEFAULT NULL,

`tokenid` varchar(255) DEFAULT NULL,

`status` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `tokengeneration`

--

INSERT INTO `tokengeneration` (`accountno`, `accholdername`, `time`, `productamt`, `tokenid`, `status`) VALUES

('17033225', 'webservera', '2018-09-06 16:03:44', '383', 'Tk29096', 'success'),

('17033225', 'webserverAdmin', '2018-09-06 16:17:21', '2323', 'Tk89703', 'success'),

('17033225', 'webservera', '2018-09-06 16:18:34', '77', 'Tk23081', 'success'),

('17033225', 'webservera', '2018-09-06 16:27:51', '123', 'Tk48594', 'success'),

('17033225', 'webservera', '2018-09-06 16:29:09', '77', 'Tk58377', 'success'),

('17033225', 'webservera', '2018-09-06 16:31:50', '400', 'Tk55446', 'success'),

('17033225', 'webservera', '2018-09-06 16:38:56', '100', 'Tk70355', 'success'),

('17033225', 'webservera', '2018-09-08 18:45:27', '123', 'Tk51870', 'success');

-- --------------------------------------------------------

--

-- Table structure for table `useraccountinfo`

--

CREATE TABLE `useraccountinfo` (

`accountno` varchar(500) NOT NULL DEFAULT '',

`pinpass` varchar(10) DEFAULT NULL,

`amount` varchar(50) DEFAULT NULL,

`opass` varchar(10) DEFAULT NULL,

`transferamount` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `useraccountinfo`

--

INSERT INTO `useraccountinfo` (`accountno`, `pinpass`, `amount`, `opass`, `transferamount`) VALUES

('68859502', '5616', '97738', 'null', 'null');

-- --------------------------------------------------------

--

-- Table structure for table `userinfo`

--

CREATE TABLE `userinfo` (

`accountno` varchar(500) NOT NULL DEFAULT '',

`accountholdername` varchar(500) DEFAULT NULL,

`mobilenumber` varchar(255) DEFAULT NULL,

`mailid` varchar(255) DEFAULT NULL,

`Address` varchar(255) DEFAULT NULL,

`city` varchar(255) DEFAULT NULL,

`pincode` varchar(10) DEFAULT NULL,

`state` varchar(255) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--

-- Dumping data for table `userinfo`

--

INSERT INTO `userinfo` (`accountno`, `accountholdername`, `mobilenumber`, `mailid`, `Address`, `city`, `pincode`, `state`) VALUES

('68859502', 'thiru', '9840512532', 'thirusp42@gmail.com', 'chennai', 'Chennai Race Course', '600032', 'TAMIL NADU');

--

-- Indexes for dumped tables

--

--

-- Indexes for table `password`

--

ALTER TABLE `password`

ADD PRIMARY KEY (`id`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `password`

--

ALTER TABLE `password`

MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=6;COMMIT;

/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

**Biotouchpass**

-- phpMyAdmin SQL Dump

-- version 4.7.0

-- https://www.phpmyadmin.net/

--

-- Host: 127.0.0.1

-- Generation Time: Oct 23, 2019 at 09:21 AM

-- Server version: 10.1.26-MariaDB

-- PHP Version: 7.1.8

SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";

SET AUTOCOMMIT = 0;

START TRANSACTION;

SET time\_zone = "+00:00";

/\*!40101 SET @OLD\_CHARACTER\_SET\_CLIENT=@@CHARACTER\_SET\_CLIENT \*/;

/\*!40101 SET @OLD\_CHARACTER\_SET\_RESULTS=@@CHARACTER\_SET\_RESULTS \*/;

/\*!40101 SET @OLD\_COLLATION\_CONNECTION=@@COLLATION\_CONNECTION \*/;

/\*!40101 SET NAMES utf8mb4 \*/;

--

-- Database: `biotouchpass`

--

-- --------------------------------------------------------

--

-- Table structure for table `users`

--

CREATE TABLE `users` (

`id` int(11) NOT NULL,

`username` varchar(200) NOT NULL,

`email` varchar(200) NOT NULL,

`password` text NOT NULL,

`phone` varchar(15) NOT NULL

) ENGINE=MyISAM DEFAULT CHARSET=latin1;

--

-- Dumping data for table `users`

--

INSERT INTO `users` (`id`, `username`, `email`, `password`, `phone`) VALUES

(3, 'thiru', 'thirusp42@gmail.com', '6206be0a5bf1cdcf07821bb35090505c', '9840512532');

--

-- Indexes for dumped tables

--

--

-- Indexes for table `users`

--

ALTER TABLE `users`

ADD PRIMARY KEY (`id`);

--

-- AUTO\_INCREMENT for dumped tables

--

--

-- AUTO\_INCREMENT for table `users`

--

ALTER TABLE `users`

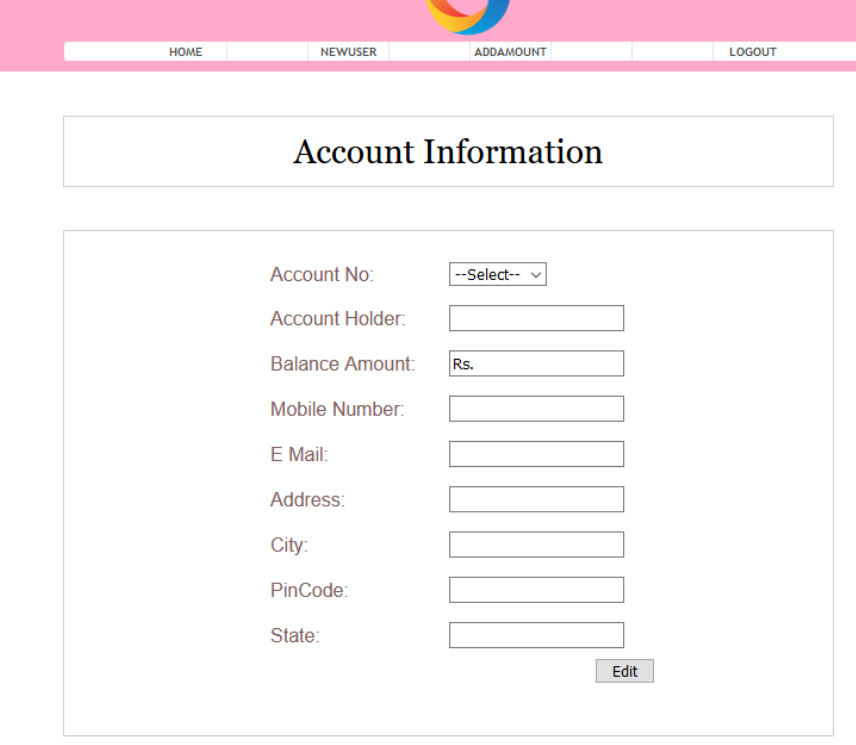
MODIFY `id` int(11) NOT NULL AUTO\_INCREMENT, AUTO\_INCREMENT=4;COMMIT;

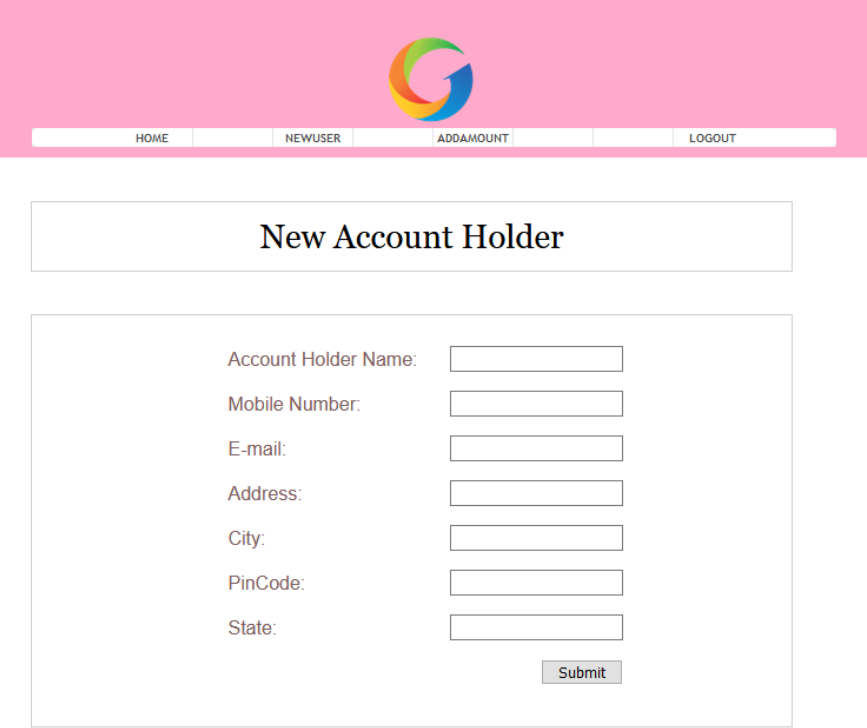
/\*!40101 SET CHARACTER\_SET\_CLIENT=@OLD\_CHARACTER\_SET\_CLIENT \*/;

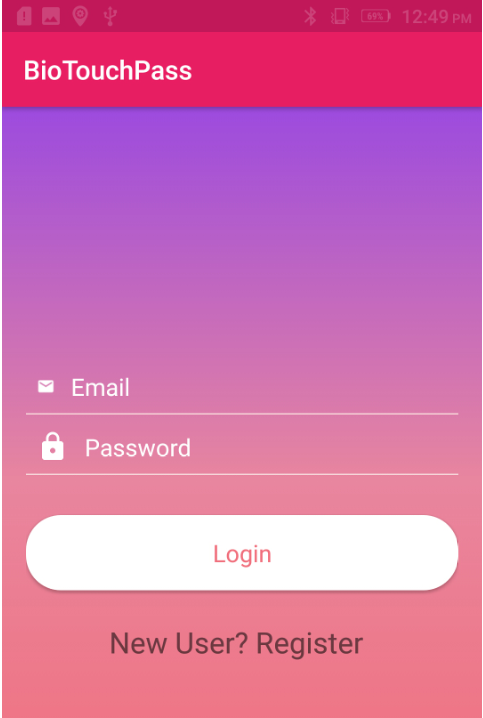
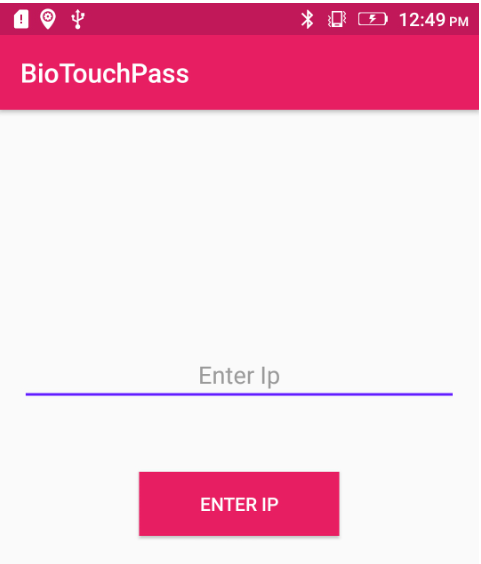
/\*!40101 SET CHARACTER\_SET\_RESULTS=@OLD\_CHARACTER\_SET\_RESULTS \*/;

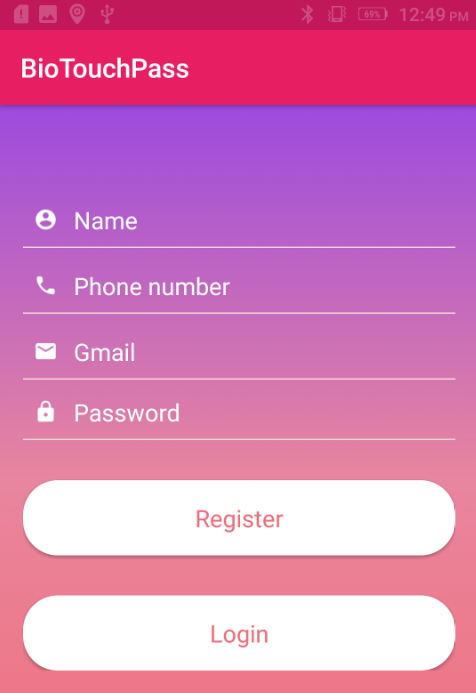
/\*!40101 SET COLLATION\_CONNECTION=@OLD\_COLLATION\_CONNECTION \*/;

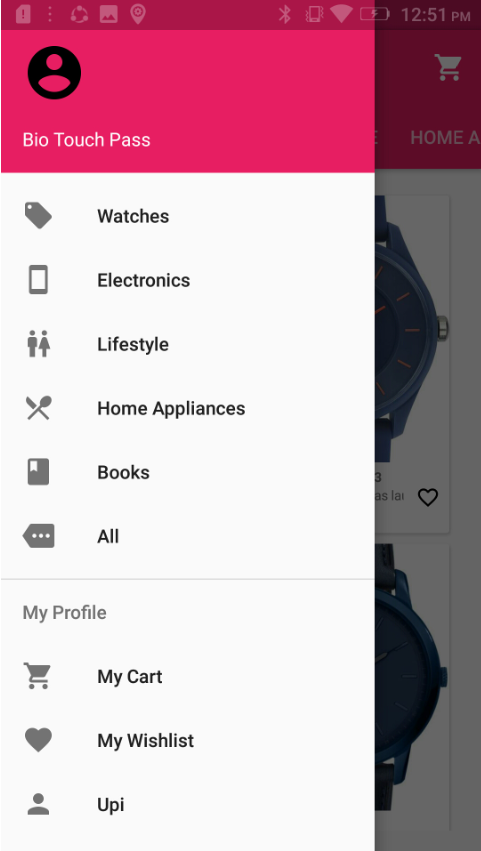
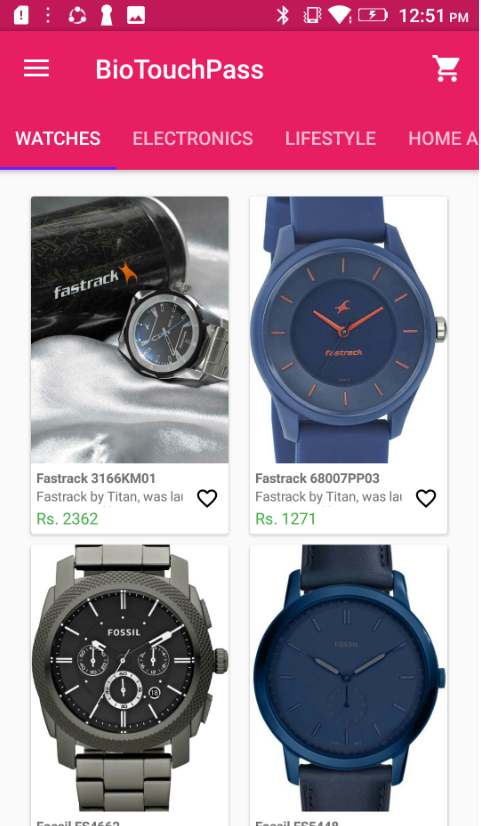
**Screenshots:**

****

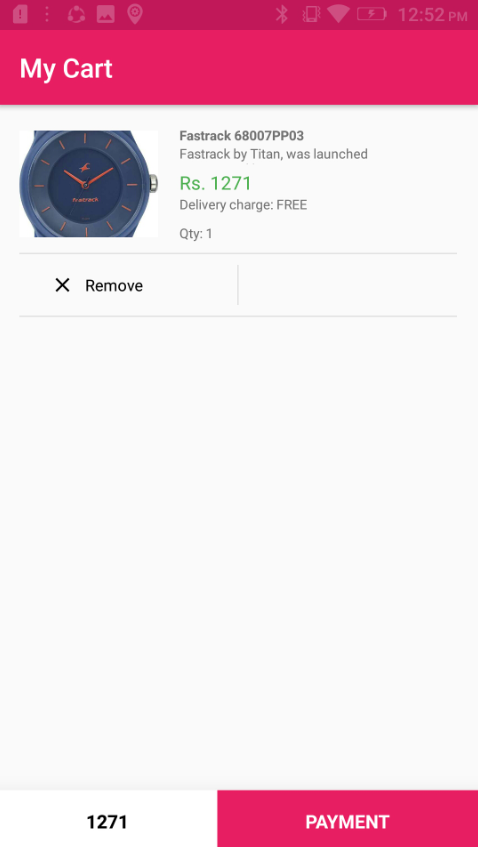
****

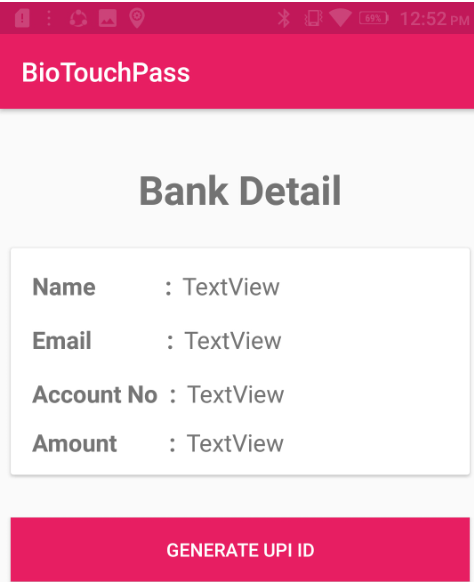
****

****

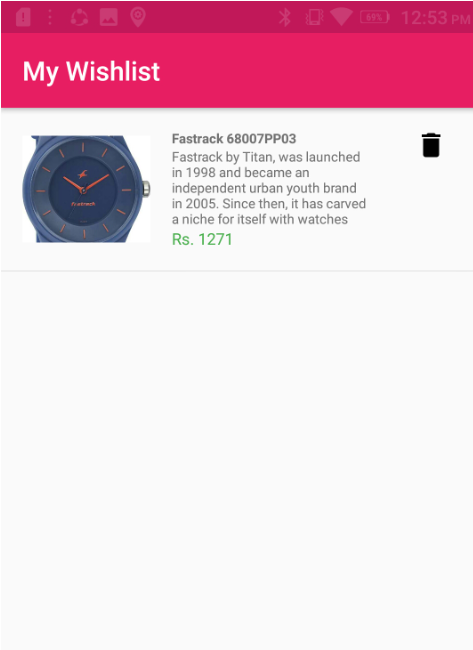
****

****

****

****

****

****

**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 Smartphones,” 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 Password Based 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.

[25] 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.