



MOBILE THEFT DETECTION APPLICATION

A PROJECT REPORT

Submitted by

SNEGAA S

211520205143

HEMACHITHRA T R

211520205053

PRIYADHARSHINI T

211520205105

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

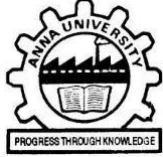
INFORMATION TECHNOLOGY

PANIMALAR INSTITUTE OF TECHNOLOGY

ANNA UNIVERSITY CHENNAI 600 025

MAY 2023

PANIMALAR INSTITUTE OF TECHNOLOGY
ANNA UNIVERSITY CHENNAI 600 025



BONAFIDE CERTIFICATE

Certified that this project report “**MOBILE THEFT DETECTION APPLICATION**” is the bonafide work of “**SNEGAA (211520205143)**
HEMACHITHRA T R (211520205053), PRIYADHARSHINI T (211519205020)” that carried out the project work under my supervision.

SIGNATURE

**Dr. S. SUMA CHRISTAL MARY, M.E, Ph.D.,
HEAD OF THE DEPARTMENT**

Department of Information Technology,
Panimalar Institute of Technology,
Poonamallee, Chennai 600 123

SIGNATURE

**Dr. S. SUMA CHRISTAL MARY,
M.E, Ph.D.,
HEAD OF THE DEPARTMENT**

Department of Information Technology,
Panimalar Institute of Technology,
Poonamallee, Chennai 600 123

**Certified that the candidates were examined in the university project
Viva-voce held on -----¹⁵⁻⁰⁵⁻²⁰²³----- at Panimalar Institute of Technology,
Chennai 600 123.**

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

A project of this magnitude and nature requires kind co-operation and support from many, for successful completion. We wish to express our sincere thanks to all those who were involved in the completion of this project.

We seek the blessing from the **Founder** of our institution **Dr. JEPPIAAR, M.A., Ph.D.**, for having been a role model who has been our source of inspiration behind our success in education in his premier institution.

We would like to express our deep gratitude to our beloved **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 also express our sincere thanks and gratitude to our dynamic **Directors Mrs. C. VIJAYA RAJESHWARI, Dr. C. SAKTHI KUMAR, M.E., Ph.D., and Dr. SARANYA SREE SAKTHI KUMAR, B.E, M.B.A., Ph.D.**, for providing us with necessary facilities for completion of this project.

We also express our appreciation and gratefulness to our respected **Principal Dr. T. JAYANTHY, M.E., Ph.D.**, who helped us in the completion of the project. We wish to convey our thanks and gratitude to our **Head of the Department, Dr. S. SUMA CHRISTAL MARY, M.E, Ph.D.**, for her full support by providing ample time to complete our project. We express our indebtedness and special thanks to our **Supervisor, Mr .VINSTON RAJA.R, M.Tech (Ph.D)** for his expert advice and valuable information and guidance throughout the completion of the project.

Last, we thank our parents and friends for providing their extensive moral support and encouragement during the course of the project.

Table of Contents

SL.NO	DESCRIPTION	PAGE NO.
1	INTRODUCTION 1.1 PROJECT DEFINITION AND DESCRIPTION	1
2	SYSTEM ANALYSIS 2.1 EXISTING SYSTEM 2.2 PROPOSED SYSTEM	2 2
3	SYSTEM DESIGN 3.1 ARCHITECTURAL DESIGN AND DATA MODELS 3.2 DATABASE DESIGN 3.3 DATA FLOW DIAGRAM 3.4 FORM DESIGN 3.5 MODULE DESCRIPTION	3 7 9 15
4	SYSTEM IMPLEMENTATION 4.1 IMPLEMENTATION OF MODULES 4.2 EXPLANATION OF PROCEDURE	16 17
5	SYSTEM TESTING	18
6	CONCLUSION	20
7	FUTURE ENHANCEMENT	21
8	APPENDIX 8.1 SOFTWARE SPECIFICATION 8.2 HARDWARE SPECIFICATION 8.3 ABOUT SOFTWARE 8.4 SAMPLE SOURCE CODE 8.5 SCREEN SHORTS	22 22 23 46 57

Abstract

Smart phones are changing the way we live our lives and have become a very important part of our day-to-day life. It also used to store contact numbers, email, in phone memory which reduces the concept of File-System to store personal contacts of an end user. Company related information are stored and viewed on anywhere through authorization. Due to its attracting features and modernized usage the people around the world are like to hold it on their packets always, if there device is missing or stolen at any situation mean their secret information or personal information going to be on wrong hands. This Project presents a technique to improve Mobile theft detection for android based mobile phones using different services like GPS, Email, and camera. In this project we forward a technique through which the stolen person or theft , who steals any android mobile are caught through installation of this application, and their user can obtain the location and image through email.

1. INTRODUCTION

1.1 Project Definition

Smart phones are changing the way we live our lives and have become a very important part of our day-to-day life and it change the ways of communication using phones, its advantage are communicating with anyone virtually through video-conferencing, email, etc., and it also used to store contact numbers, email, in phone memory which reduces the concept of File-System to store personal contacts of an end user. Now a day, smart phones are acting like a computer and used to store information, documents etc., and can be shared with anyone throughout the world. These latest Technologies are very helpful for doing business. Company related information are stored and viewed on anywhere through authorization. Through the usage of android the on mobiles, it look like Pc on Packet. Due to its attracting features and modernized usage the people around the world are like to hold it on their packets always, if there device is missing or stolen at any situation mean their secret information or personal information going to be on wrong hands so this project forward a technique through which the stolen person or theft, who steals any android mobile are caught through installation of this application, you can receive thief photo, current location. It gives the exact details about the theft and his/her last location of IMSI number changed through email you can get image and through Email you can Receive URL link Google map location.

Today Mobile phones are becoming more techno logically advanced and offer more features. Specially, Smart phones are having more advanced computing capability than a feature phone. Smart phones can run applications and can access the internet directly unlike cell phones rely on a carrier to get that. The reason Smart phones can run applications because these phones have CPU, memory and all other stuff that allows PCs to do the same thing. The mobile operating systems (OS) used by modern smart phones include Android, BlackBerry, iOS and Symbian , which are the world's best mobile operating systems. There are many differences between their features and performance.

In Android 2.3 various changes have been made in the user interface. There are changes in settings and menus which make it easier for the users to navigate and control the features of system and device. This application uses Android OS which demonstrates a system that uses a regular mobile phone equipped with a GPS receptor and connected to a global system for mobile (GSM) network that takes advantage of these technologies in behalf of the user safety. AALTM is a useful mobile application that combines several features which aims at the user's security. Mobile location estimation and tracking technique for wireless communication systems. The location estimation is based on the differences of Downlink signal attenuations.

A mobile tracking technique via piecewise linear optimization using a simple genetic algorithm is applied to improve the locations estimation. With the inclusion of built-in GPS and 802.11 supports in the mobile phone hardware, the mobility information are easily captured and routed to a remote system via opportunistic connections

over Internet. In this, a Web based mobility analysis system which collects location data from mobile phone users via opportunistic Internet. A method to track a mobile device by monitoring the signal powers of the mobile transmitter measured at several base stations. The signal power measurements at several base stations with the power maps (non-linear function of the position of a mobile) were compared to get the likelihood at each position. This method is mainly focused on mobile devices that are mounted in vehicles.

The two techniques to locate and track cellular phones using digital cellular mobile telephony networks. One is based on time of arrival methods with a minimum of three base stations required, while the other technique uses angle of arrival methods that require only two base-stations. Both methods were examined for a multipath fading environment. There are already several applications in the market that offer tracking systems and anti theft applications like mGuard, detect non-authorized SIM cards. Unlike this application, the AALTM is able to enable the GPS when a non-authorized SIM card is detected in the device by comparing the Integrated Circuit Chip Card Identification (ICC ID). The ICC ID number is unique for each SIM card. Tracking applications such as, Mobile Tracking System, AccuTracking, and PhoneBak (also anti-theft application), are already rooted in the mobile phone market.

Our proposed system presents a novel Mobile-theft Detection application for android based devices. Our application can send email through virtually connecting the mobile hardware to operation process with their automated background activity of android mobile. When our pattern mobile locking system goes to wrong. Camera suddenly ON capture the thief photo and collect the GPS location of mobile send to authority person via mail. The application deploys an enterprise security solution that meets users immediate and long term requirements by providing the images and videos of the thief, which makes easy for the user to identify the thief and make him/her get caught and arrested.

1.2 Project description:

User Registration

This module is used to registering the user details for accessing the benefit of app. The details such as name, address, mobile no, email ID, and username, password, photo of the user. This all details stored in to the database.

Password Matching

If the user wants to open the mobile it ask the password pattern of the mobile. If the user put the password pattern the this app matching the password with already default password if the password was matched the the mobile allow the user for work other than it don't allow the user for work.

Mail Sending:

If the user or any one put the wrong password then this app automatically open the front Camera suddenly captures the photo and collects the GPS location of mobile. If this photo is matched with user photo then it sends the password pattern to the user mail id. Else it sends the capturing photo and current location of mobile send to authority person via mail.

2. SYSTEM ANALYSIS

2.1 Existing System

In existing mobile tracker application system is based on SMS option only. If anybody change the SIM card that card numbers only goes to the authority or relative person. Service providers are also required since they offer different kinds of LBS services to users and are responsible for processing service requests and responds back by sending request results. Servers can calculate locations or position, search for a path or route travelled, or search specific information regarding users position. Usually all the information requested by users are not store and maintain by the Service providers, geographic data are collect and stored by content provider. Information's such as Location-based and other related data are also collect and store by the content provider. Service servers will requests and process these data and then returned to users.

- No live Updates
- Process will start when request is given
- Need service providers help
- We cannot get the exact mobile location and thief picture.

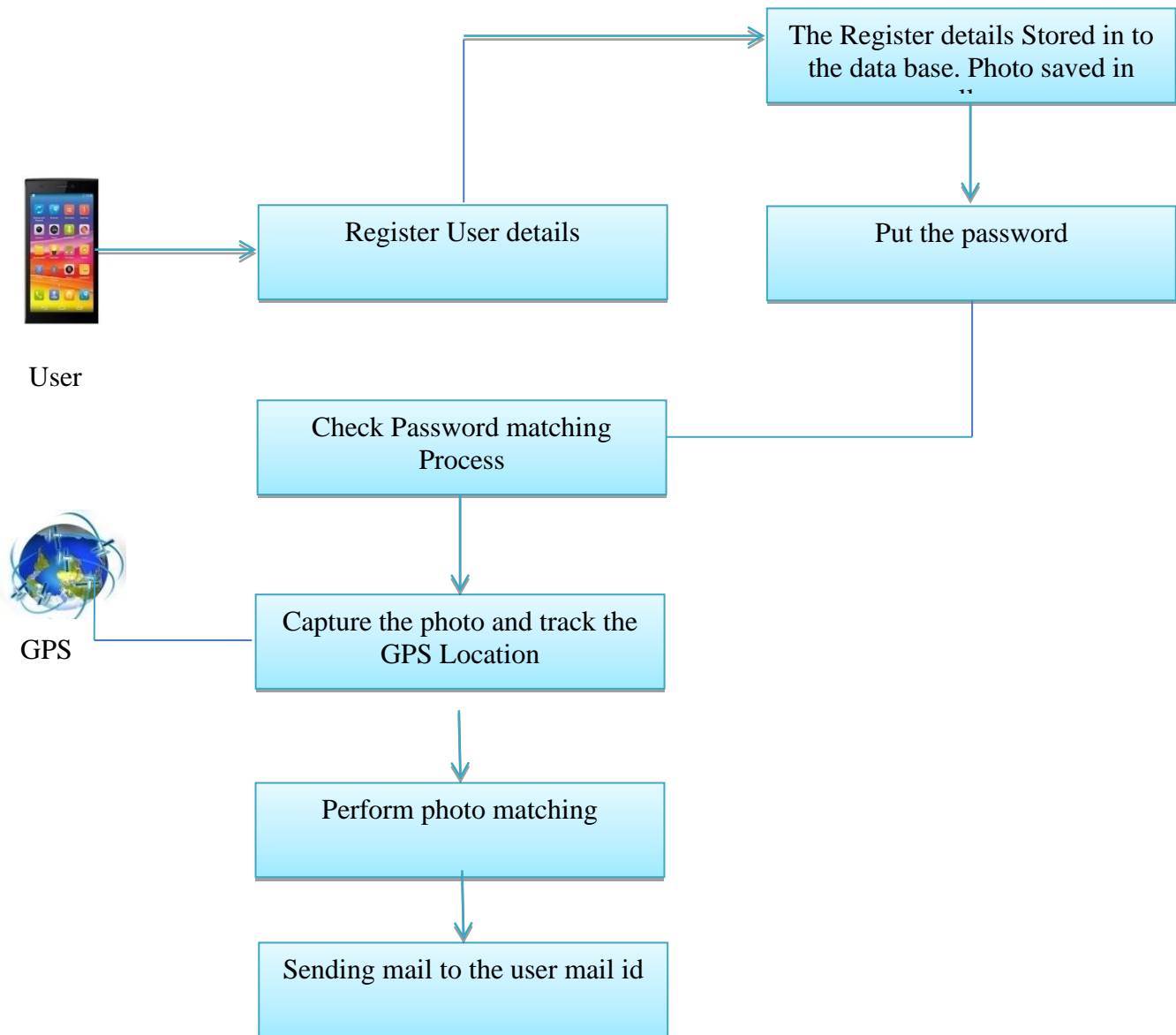
2.2 Proposed system

Our proposed system presents a novel Mobile-theft Detection application for android based devices. Our application can send email through virtually connecting the mobile hardware to operation process with their automated background activity of android mobile. When our pattern mobile locking system goes to wrong. Camera suddenly ON capture the thief photo and collect the GPS location of mobile send to authority person via mail. The application deploys an enterprise security solution that meets users immediate and long term requirements by providing the images and videos of the thief, which makes easy for the user to identify the thief and make him/her get caught and arrested.

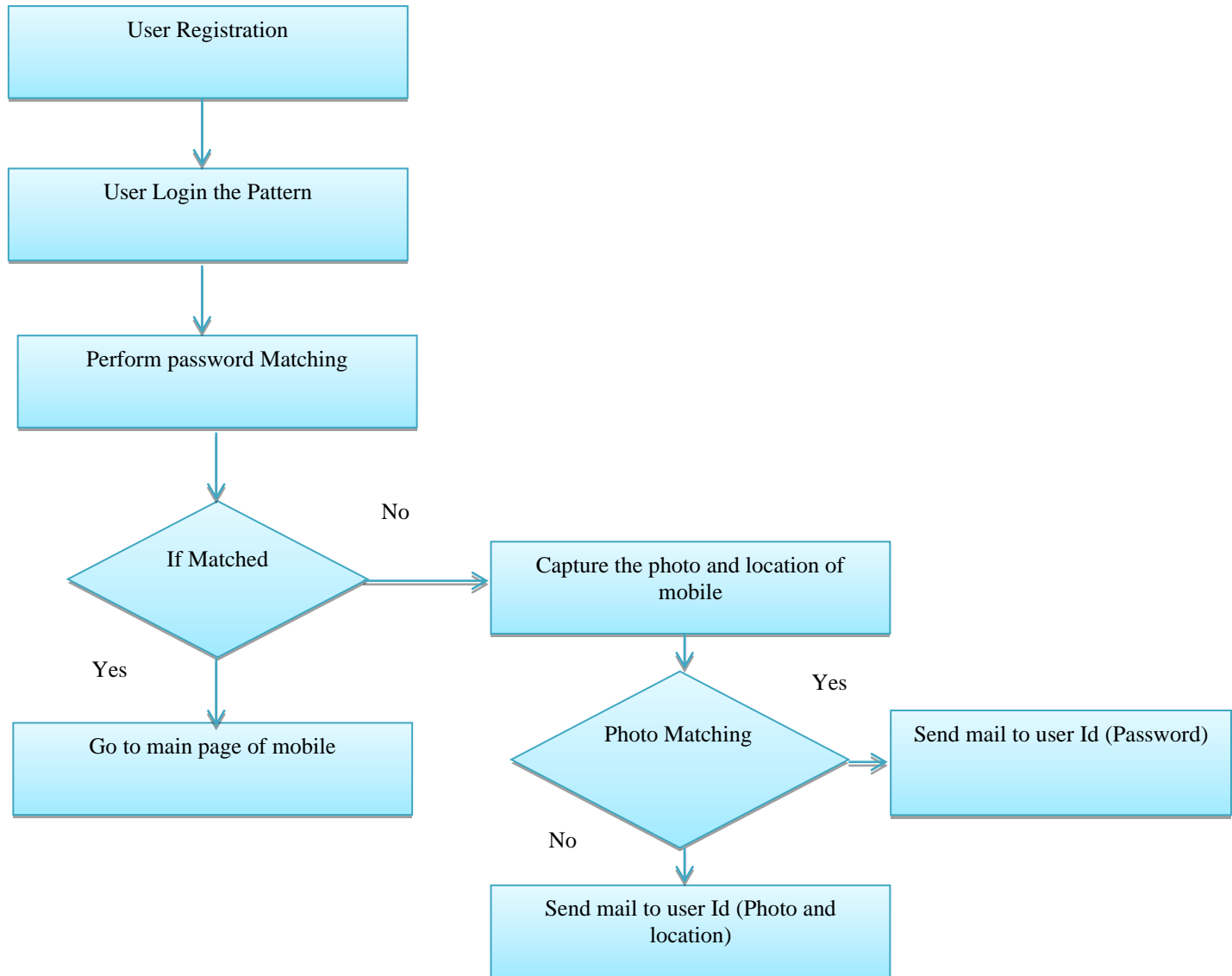
- Live Updates
- Automatically process start without request
- No need service providers help
- We can get the exact mobile location and thief picture.
- High reliability

3. SYSTEM DESIGN

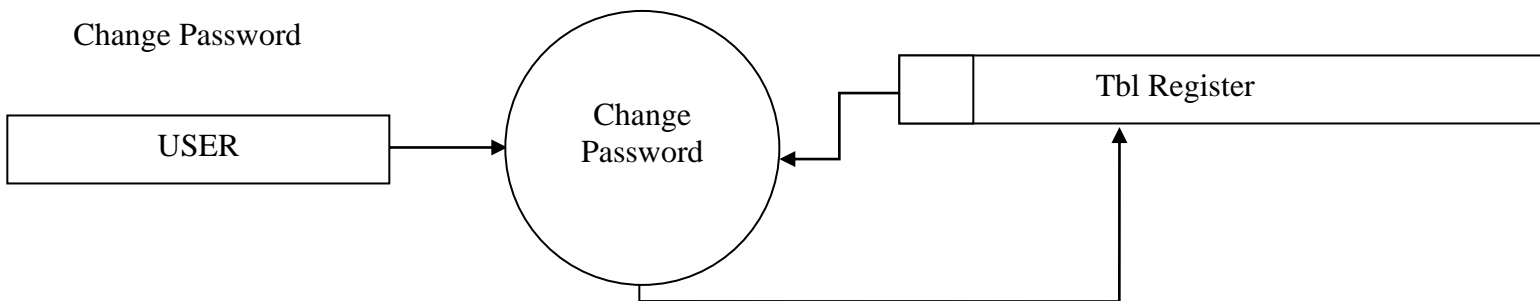
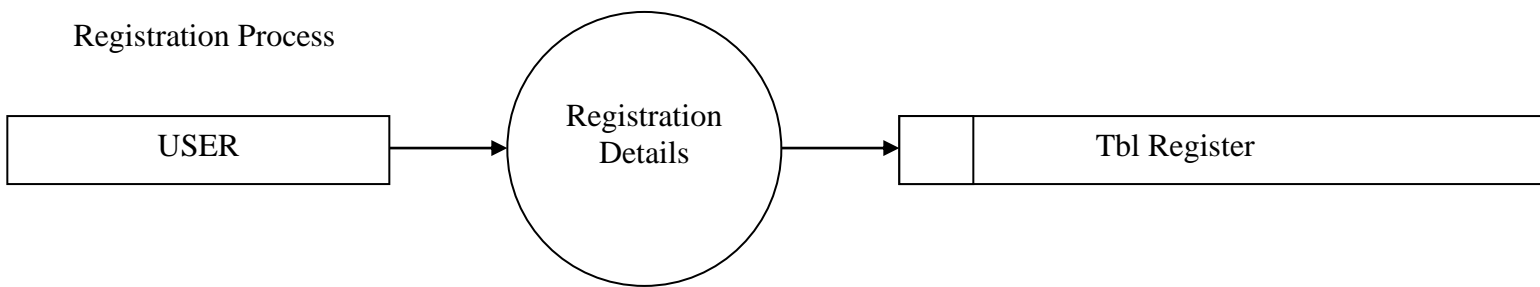
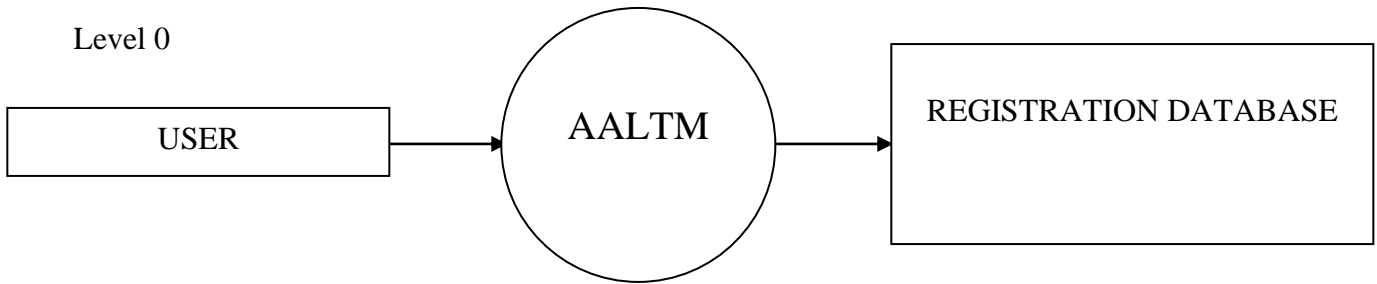
3.1 ARCHITECTUE DESIGN



Flow Diagram



3.2 Dataflow Diagram



3.3 Database Design

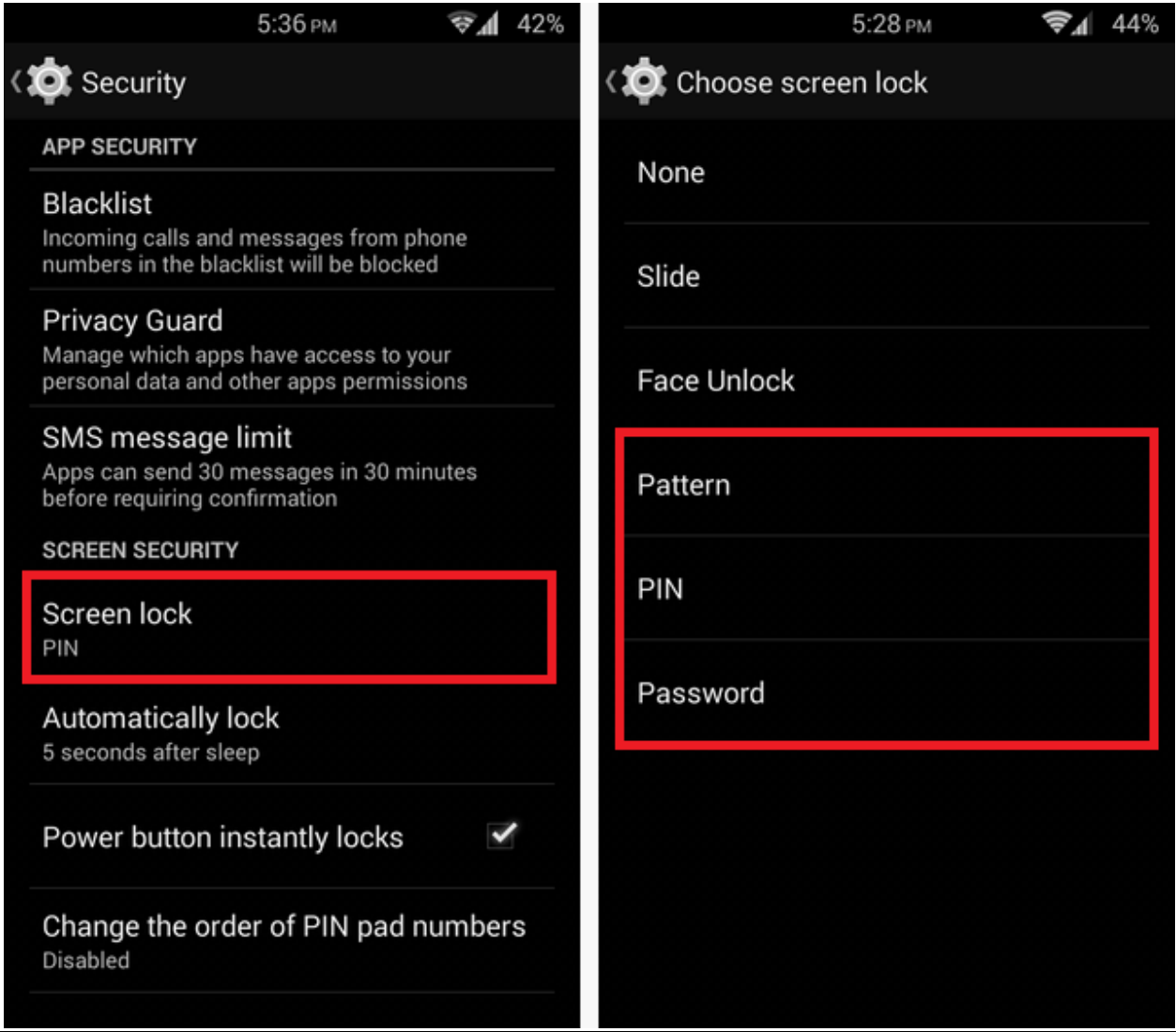
Registration Details

Field	Type	Size	Constrain	Description
Name	Varchar	15	Primary Key	Username
Email ID	Varchar	20	Not Null	Email Address
Phone Num	Varchar	10	Not Null	Mobile Num

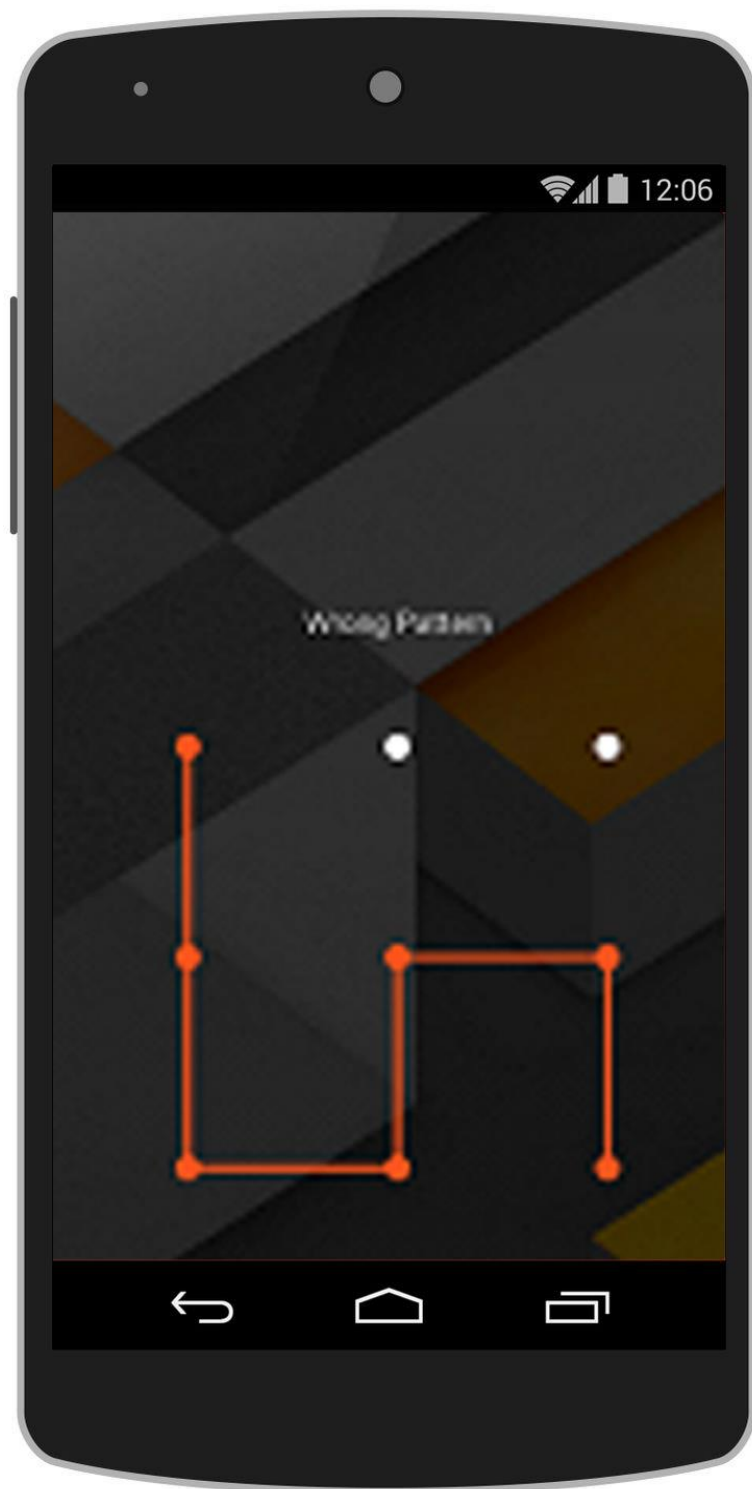
Address	Varchar	20	Not Null	User Address
Password	Varchar	15	Not Null	User Password

3.4 Form Design:

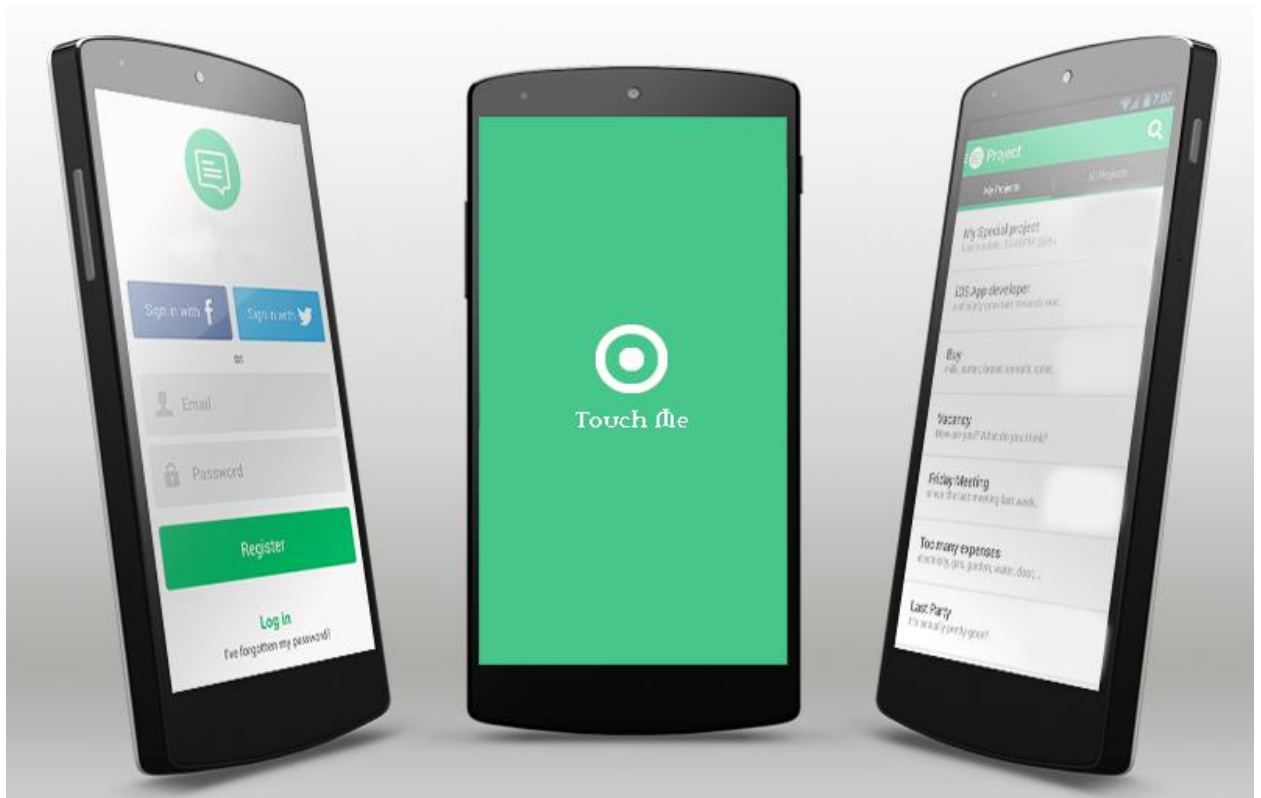
Settings



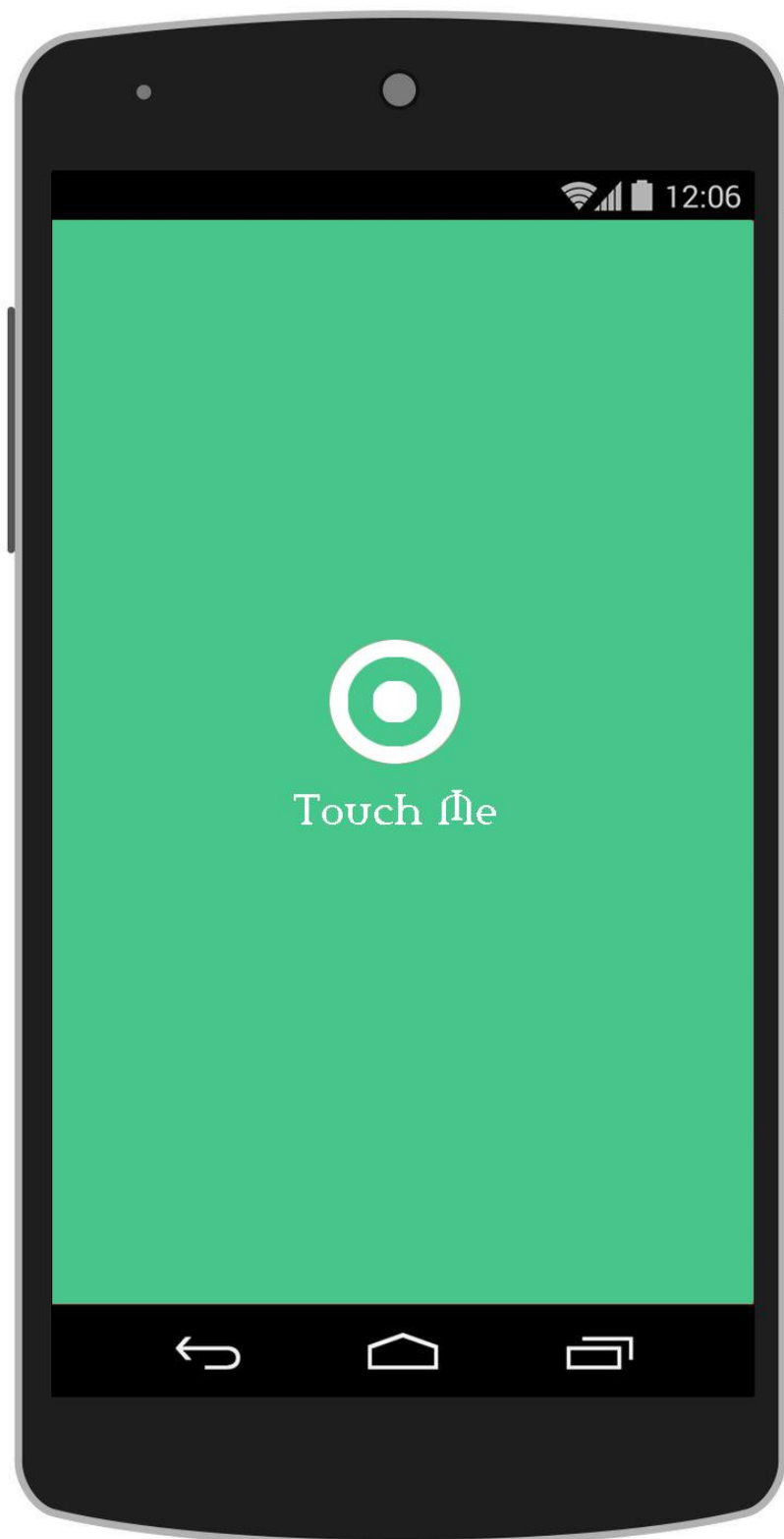
Pattern



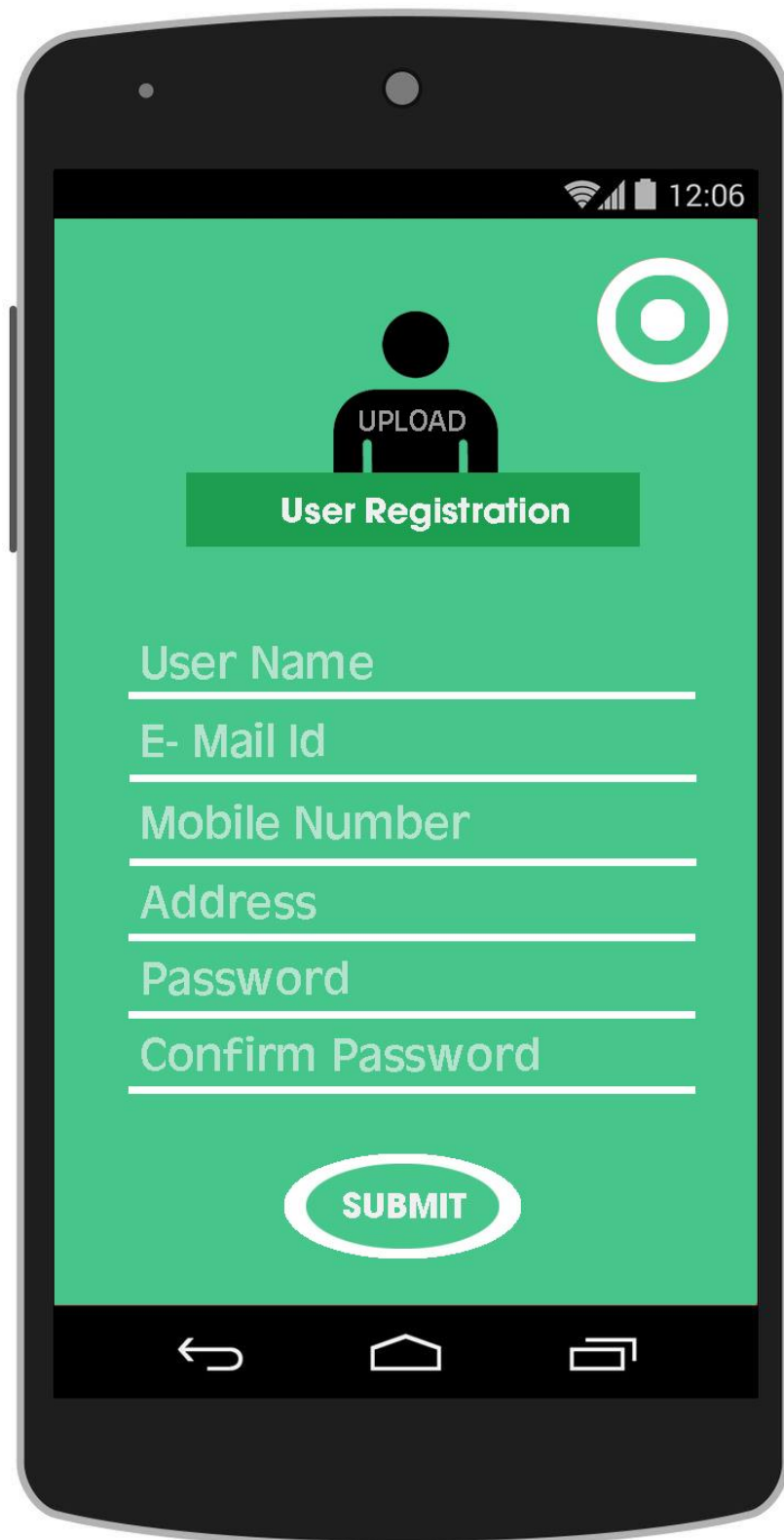
Application Overview



Front Page



First Time Registration



The image shows a smartphone screen with a green background for a user registration form. At the top, there is a status bar with a Wi-Fi icon, a battery icon, and the time 12:06. Below the status bar, there is a black silhouette of a person with the word "UPLOAD" written on it, and a white target icon in the top right corner. A green rectangular button with the text "User Registration" is centered below the upload icon. The form consists of six white input fields with labels: "User Name", "E- Mail Id", "Mobile Number", "Address", "Password", and "Confirm Password". At the bottom of the form is a white oval button with the text "SUBMIT". The smartphone has a black bezel and a home button at the bottom.

12:06

UPLOAD

User Registration

User Name

E- Mail Id

Mobile Number

Address

Password

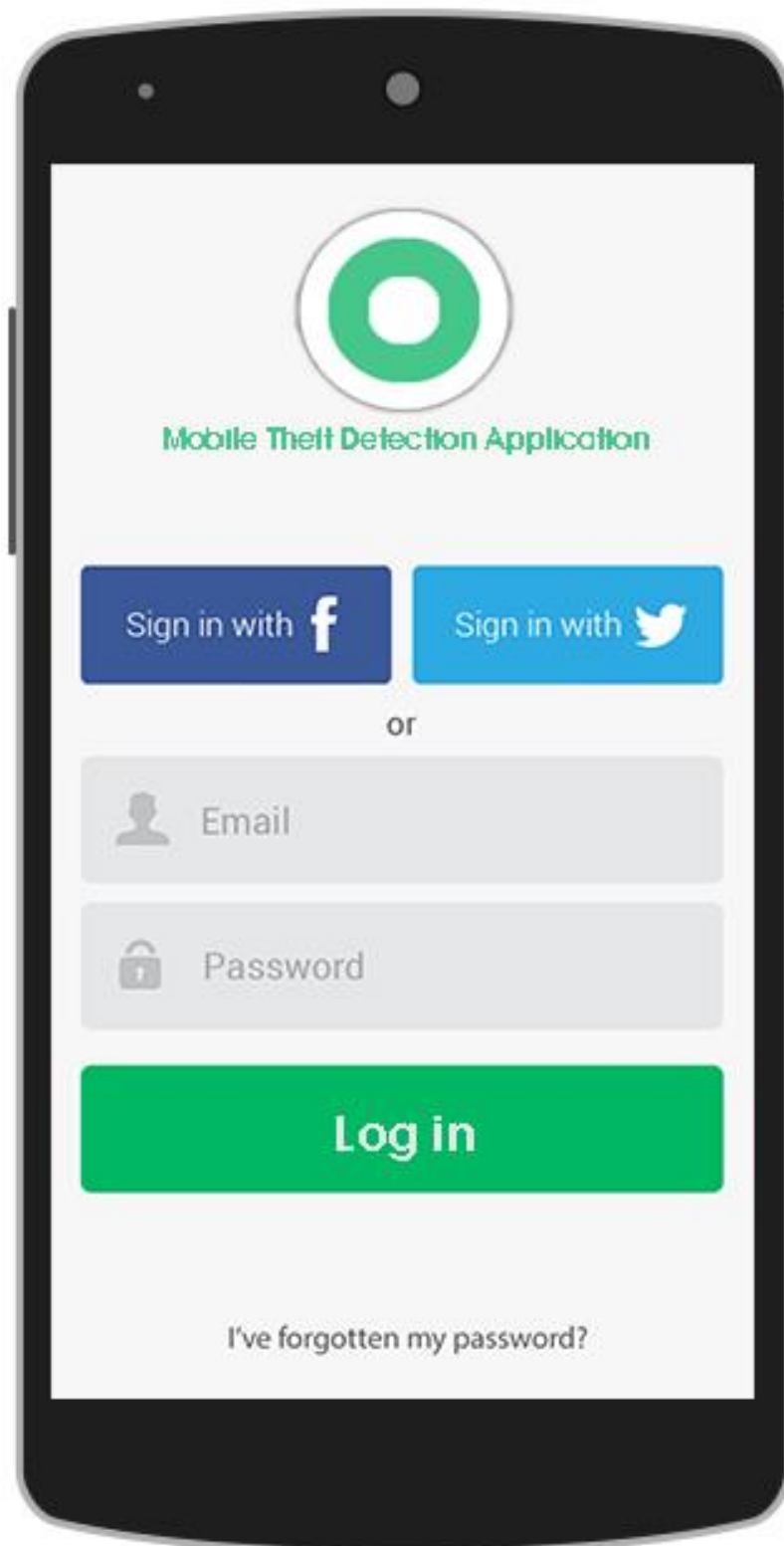
Confirm Password

SUBMIT

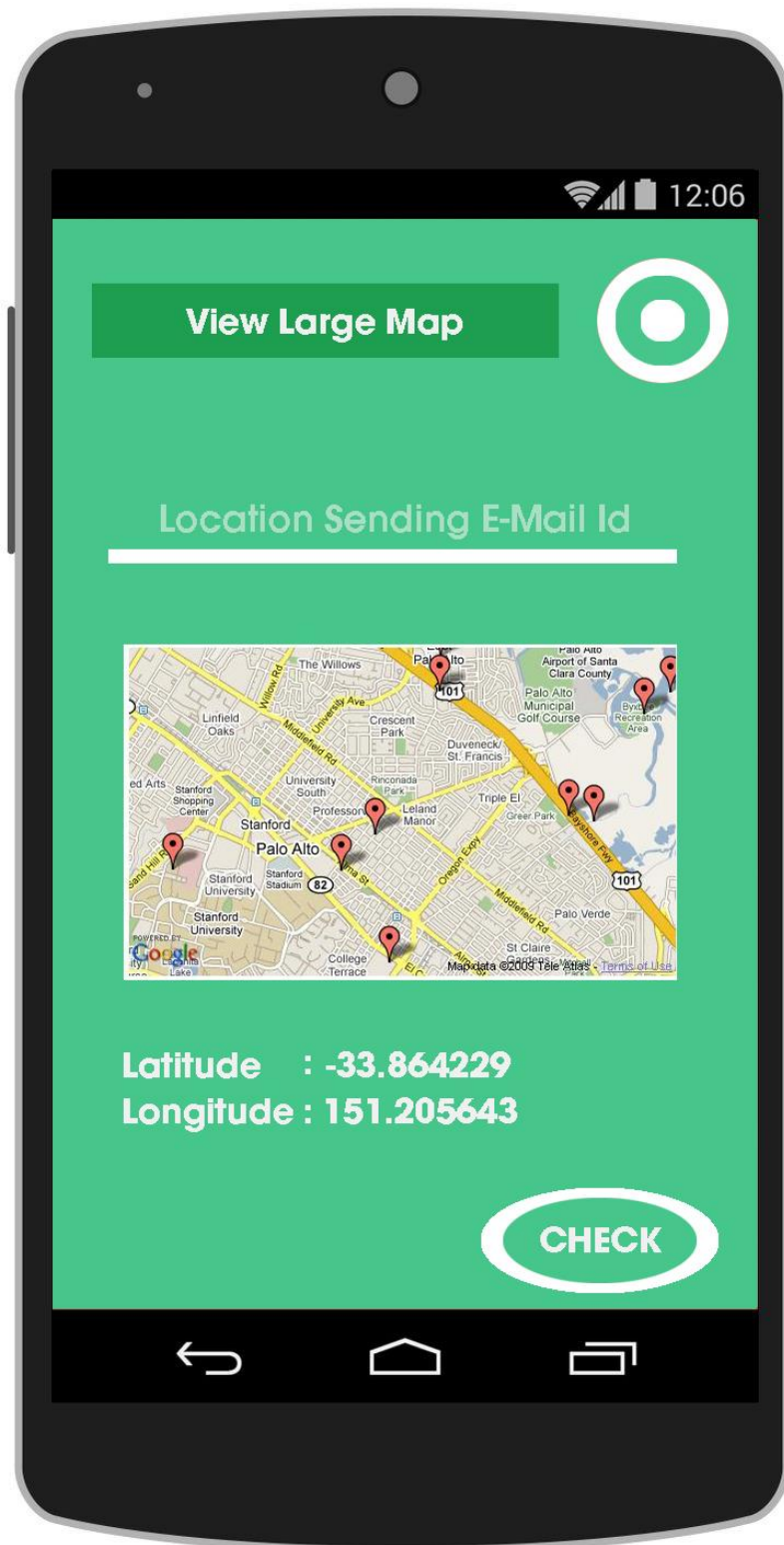
Home Page for Enable / Disable Application & Customer care



First Time Registration



GPS Tracker



3.5 Module description:

User Registration

This module is used to registering the user details for accessing the benefit of app. The details such as name, address, mobile no, email ID, and username, password, photo of the user. This all details stored in to the database.

Password Matching

If the user wants to open the mobile it asks the password pattern of the mobile. If the user put the password pattern the this app matching the password with already default password if the password was matched the the mobile allow the user for work other than it don't allow the user for work.

Mail Sending:

If the user or any one put the wrong password then this app automatically open the front Camera suddenly captures the photo and collects the GPS location of mobile. If this photo is matched with user photo then it sends the password pattern to the user mail id. Else it sends the capturing

4. SYSTEM IMPLEMENTATION

Installation Procedure

- Download the Mobile theft detection application in the play store.
- Then install the application.
- Accept the terms & conditions.
- Then click run button to open the application

After running the application the home page was displayed now the user can fill the first registration form then the application always works in your mobile. The user can allow enabling and disabling the application.

4.1. IMPLEMENTATION OF MODULE

User Registration

- After completing the installation the user must fill the user registration form.
- Enter all the details like name, Email Id, Phone num, Address, Password.
- After all the fields are validating, click submit.
- Now the application automatically runs in our mobile background.

Disable / Enable the application

- If the users need to disable the application click the App Icon.
- Then the click the disable / enable button on the home screen.

5. SYSTEM TESTING

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

TYPES OF TESTS

Unit testing

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

Integration testing

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

Functional test

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

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6. CONCLUSION

This App presents a novel Mobile-theft Detection application for android based devices. The application deploys an enterprise security solution that meets users immediate and long term requirements by providing the images of the thief, which makes easy for the user to identify the thief and make him/her get caught and arrested. We are enhancing this application by providing the information about the location of the android based Smartphone with the help of text messages. With the advent of time, technology is evolving every day.

7. FUTURE ENHANCEMENT

The application is developed to overcome the difficulties in the existing system. Some changes in the system can save the time and increase the performance of the system in the upcoming days.

- Develop the same app in all other platforms like IOS, WINDOWS.
- Reduce the size of the app.
- Create theft alarm in this app.

8. APPENDICES

8.1. HARDWARE REQUIREMENT

CPU type	: Intel i5
Clock speed	: 3.0 GHz
Ram size	: 4 GB
Hard disk capacity	: 1 TB
Monitor type	: 21 Inch color monitor
Keyboard type	: internet keyboard
Mobile	: ANDROID MOBILE (Galaxy Ace)

8.2. SOFTWARE REQUIREMENT

Operating System : Android

Language : JAVA, PHP

8.3 ABOUT SOFTWARE

Java Technology

Java technology is both a programming language and a platform.

The Java Programming Language

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

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

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.

The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. 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.

The Java platform has two components:

- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. 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 of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.

Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

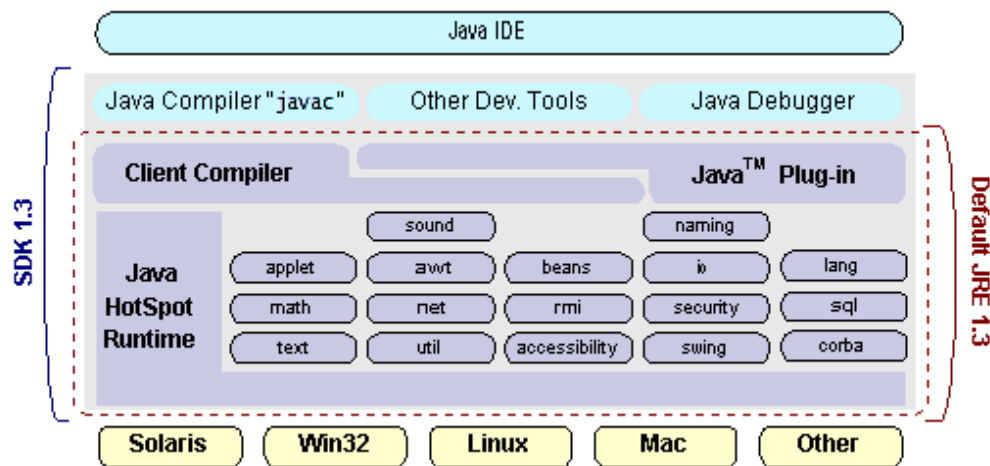
An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality.

Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.

- **Software components:** Known as JavaBeans™, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™):** Provides uniform access to a wide range of relational databases.
- The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages.

We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.

- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded “on the fly,” without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a de facto standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE.

There is a 16-bit and a 32-bit version of this program and each maintains a separate list of ODBC data sources. From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of "plug-in" database connectivity modules, or drivers. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC's framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90 day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. SQL Level API

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to "generate" JDBC code and to hide many of JDBC's complexities from the end user.

2. **SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. **JDBC must be implemental on top of common database interfaces**

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT’s, INSERT’s, DELETE’s and UPDATE’s, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible. Finally we decided to proceed the implementation using Java Networking. And for dynamically updating the cache table we go for MS Access database. Java ha two things: a programming language and a platform.

8.4 SOURCE CODE

```
//Declaration of all variables, vectors and haarcascades
```



```

Image<bgr,> currentFrame;
Capture grabber;
HaarCascade face;
HaarCascade eye;
MCvFont font = new MCvFont(FONT.CV_FONT_HERSHEY_TRIPLEX, 0.5d, 0.5d);
Image<gray,> result, TrainedFace = null;
Image<gray,> gray = null;
List<image<gray,>> trainingImages = new List<image<gray,>>();
List<string> labels= new List<string>();
List<string> NamePersons = new List<string>();
int ContTrain, NumLabels, t;
string name, names = null;

```

Then load the haarcascades for face detection, then I do a little “procedure” to load of previous trained faces and labels for each image stored previously:

Hide Copy Code

```

//Load haarcascades for face detection
face = new HaarCascade("haarcascade_frontalface_alt_tree.xml");
eye = new HaarCascade("haarcascade_eye.xml");
try
{
    //Load of previus trainned faces and labels for each image
    string Labelsinfo = File.ReadAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt");
    string[] Labels = Labelsinfo.Split('%');
    NumLabels = Convert.ToInt16(Labels[0]);
    ContTrain = NumLabels;
    string LoadFaces;

    for (int tf = 1; tf < NumLabels+1; tf++)
    {
        LoadFaces = "face" + tf + ".bmp";
        trainingImages.Add(new Image<gray,>(Application.StartupPath + "/TrainedFaces/" + LoadFaces));
        labels.Add(Labels[tf]);
    }
}

```

```

    }
}
catch(Exception e)
{
    //MessageBox.Show(e.ToString());
    MessageBox.Show("Nothing in binary database, please add at least a face", "Trained faces load",
    MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}

```

Initialize the capture device, and FrameGrabber event that performs the detection and process of images for each frame captured:

Hide Copy Code

```

grabber = new Capture();
grabber.QueryFrame();
//Initialize the FrameGrabber event
Application.Idle += new EventHandler(FrameGrabber);
button1.Enabled = false;

```

Passing to FrameGrabber event (main part of prototype) we use the most important methods and objects: DetectHaarCascade And EigenObjectRecognizer and perform operations For each face detected in one frame:

Hide Shrink Copy Code

```

MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
    face,
    1.2,
    10,
    Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
    new Size(20, 20));

//Action for each element detected
foreach (MCvAvgComp f in facesDetected[0])
{
    t = t + 1;
}

```

```

        result = currentFrame.Copy(f.rect).Convert<gray,>().Resize(100, 100,
Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

        //draw the face detected in the 0th (gray) channel with blue color
        currentFrame.Draw(f.rect, new Bgr(Color.Red), 2);

    if (trainingImages.ToArray().Length != 0)
    {
        //TermCriteria for face recognition with numbers of trained images like maxIteration
        MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);

        //Eigen face recognizer
        EigenObjectRecognizer recognizer = new EigenObjectRecognizer(
            trainingImages.ToArray(),
            labels.ToArray(),
            5000,
            ref termCrit);

        name = recognizer.Recognize(result);

        //Draw the label for each face detected and recognized
        currentFrame.Draw(name, ref font, new Point(f.rect.X - 2, f.rect.Y - 2), new Bgr(Color.LightGreen));
    }
}

```

Parameters:

Code of training button (This perform the adding of training faces and labels for each):

Hide Shrink Copy Code

```

try
{
    //Trained face counter
    ContTrain = ContTrain + 1;
}

```

```

//Get a gray frame from capture device
gray = grabber.QueryGrayFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);

//Face Detector
MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(
face,
1.2,
10,
Emgu.CV.CvEnum.HAAR_DETECTION_TYPE.DO_CANNY_PRUNING,
new Size(20, 20));

//Action for each element detected
foreach (MCvAvgComp f in facesDetected[0])
{
    TrainedFace = currentFrame.Copy(f.rect).Convert<gray,>();
    break;
}

//resize face detected image for force to compare the same size with the
//test image with cubic interpolation type method
TrainedFace = result.Resize(100, 100, Emgu.CV.CvEnum.INTER.CV_INTER_CUBIC);
trainingImages.Add(TrainedFace);
labels.Add(textBox1.Text);

//Show face added in gray scale
imageBox1.Image = TrainedFace;

//Write the number of triained faces in a file text for further load
File.WriteAllText(Application.StartupPath + "TrainedFaces/TrainedLabels.txt",
trainingImages.ToArray().Length.ToString() + "%");

//Write the labels of triained faces in a file text for further load
for (int i = 1; i < trainingImages.ToArray().Length + 1; i++)

```

```

        {
            trainingImages.ToArray()[i - 1].Save(Application.StartupPath + "/TrainedFaces/face" + i + ".bmp");
            File.AppendAllText(Application.StartupPath + "/TrainedFaces/TrainedLabels.txt", labels.ToArray()[i -
1] + "%");
        }

```

```

        MessageBox.Show(textBox1.Text + "&acute;s face detected and added :)", "Training OK",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
    }
    catch
    {
        MessageBox.Show("Enable the face detection first", "Training Fail", MessageBoxButtons.OK,
        MessageBoxIcon.Exclamation);
    }
}

```

Capture

```

package edu.gvsu.cis.masl.camerademo;

```

```

import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

```

```

public class MyCameraActivity extends Activity {
    private static final int CAMERA_REQUEST = 1888;
    private ImageView imageView;

    @Override
    public void onCreate(Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.main);
this.imageView = (ImageView)this.findViewById(R.id.imageView1);
Button photoButton = (Button) this.findViewById(R.id.button1);
photoButton.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        Intent cameraIntent = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(cameraIntent, CAMERA_REQUEST);
    }
});
}

protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == CAMERA_REQUEST && resultCode == RESULT_OK) {
        Bitmap photo = (Bitmap) data.getExtras().get("data");
        imageView.setImageBitmap(photo);
    }
}

a = (ImageButton)findViewById(R.id.imageButton1);

a.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View v) {
        selectImage();
    }
});
}

private File savebitmap(Bitmap bmp) {
    String extStorageDirectory = Environment.getExternalStorageDirectory().toString();

```

```

OutputStream outStream = null;
// String temp = null;
File file = new File(extStorageDirectory, "temp.png");
if (file.exists()) {
    file.delete();
    file = new File(extStorageDirectory, "temp.png");
}
try {
    outStream = new FileOutputStream(file);
    bmp.compress(Bitmap.CompressFormat.PNG, 100, outStream);
    outStream.flush();
    outStream.close();

} catch (Exception e) {
    e.printStackTrace();
    return null;
}
return file;
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}

private void selectImage() {
    final CharSequence[] options = { "Take Photo", "Choose from Gallery","Cancel" };
    AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
    builder.setTitle("Add Photo!");
    builder.setItems(options, new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int item) {
            if (options[item].equals("Take Photo"))

```

```

{
    Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    File f = new File(android.os.Environment.getExternalStorageDirectory(), "temp.jpg");
    intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(f));
    //pic = f;
    startActivityForResult(intent, 1);
}
else if (options[item].equals("Choose from Gallery"))
{
    Intent intent = new
Intent(Intent.ACTION_PICK, android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
    startActivityForResult(intent, 2);
}
else if (options[item].equals("Cancel")) {
    dialog.dismiss();
}
}
});
builder.show();
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_OK) {
        if (requestCode == 1) {
            //h=0;
            File f = new File(Environment.getExternalStorageDirectory().toString());
            for (File temp : f.listFiles()) {
                if (temp.getName().equals("temp.jpg")) {
                    f = temp;
                    File photo = new File(Environment.getExternalStorageDirectory(), "temp.jpg");
                    //pic = photo;
                    break;

```



```

    }
}
try {
    Bitmap bitmap;
    BitmapFactory.Options bitmapOptions = new BitmapFactory.Options();
    bitmap = BitmapFactory.decodeFile(f.getAbsolutePath(),
        bitmapOptions);
    a.setImageBitmap(bitmap);
    String path = android.os.Environment
        .getExternalStorageDirectory()
        + File.separator
        + "Phoenix" + File.separator + "default";
    //p = path;
    f.delete();

    OutputStream outFile = null;
    File file = new File(path, String.valueOf(System.currentTimeMillis()) + ".jpg");
    try {
        outFile = new FileOutputStream(file);
        bitmap.compress(Bitmap.CompressFormat.JPEG, 85, outFile);
//pic=file;
        outFile.flush();
        outFile.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } catch (Exception e) {
        e.printStackTrace();
    }
} catch (Exception e) {
    e.printStackTrace();
}

```

```

    } else if (requestCode == 2) {
        Uri selectedImage = data.getData();
        // h=1;
//imgui = selectedImage;

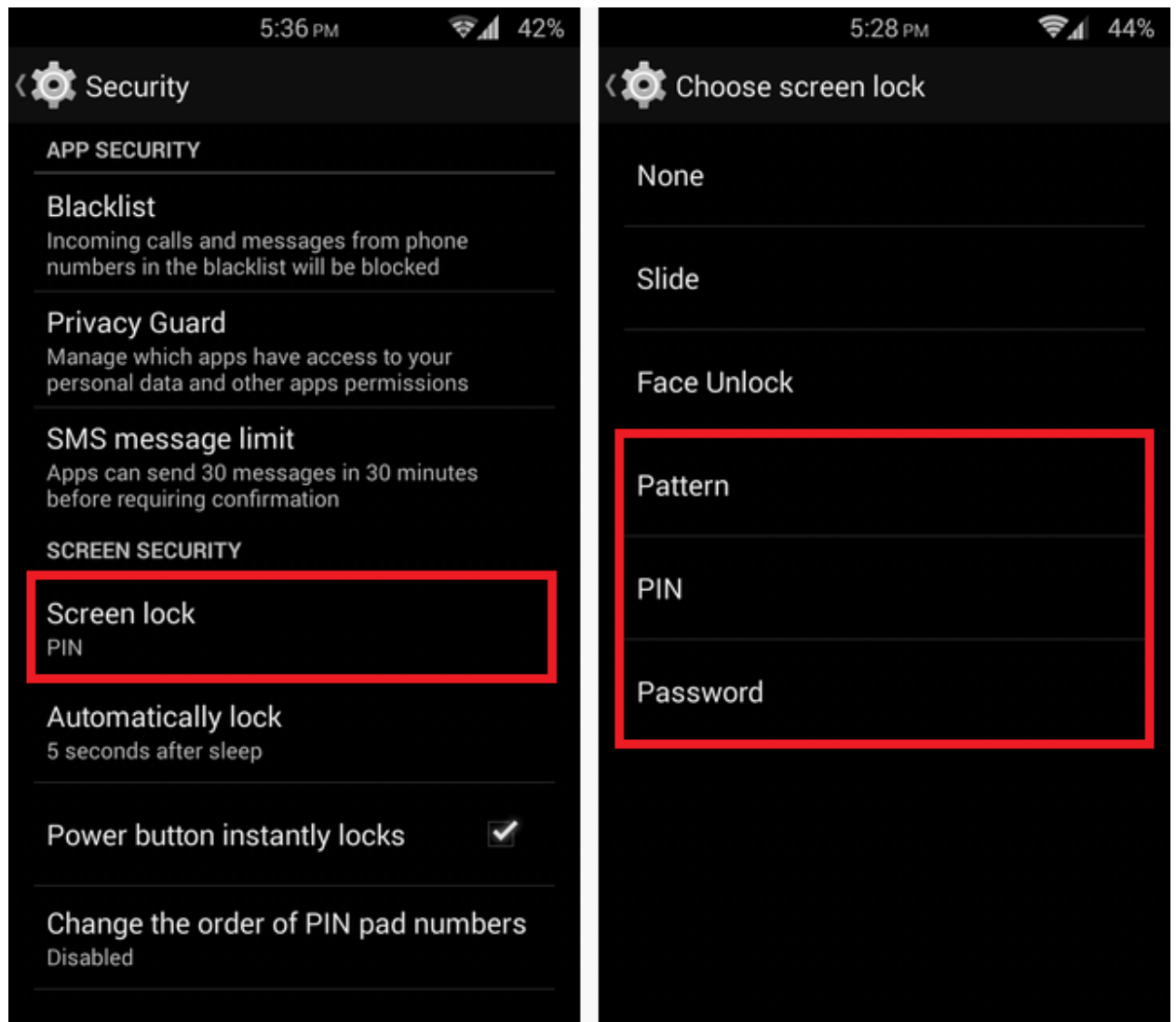
        String[] filePath = { MediaStore.Images.Media.DATA };
        Cursor c = getContentResolver().query(selectedImage,filePath, null, null, null);
        c.moveToFirst();
        int columnIndex = c.getColumnIndex(filePath[0]);
        String picturePath = c.getString(columnIndex);
        c.close();
        Bitmap thumbnail = (BitmapFactory.decodeFile(picturePath));

        Log.w("path of image from gallery.....*****.....", picturePath+"");
        a.setImageBitmap(thumbnail);
    }
}

```

3.4 Form Design:

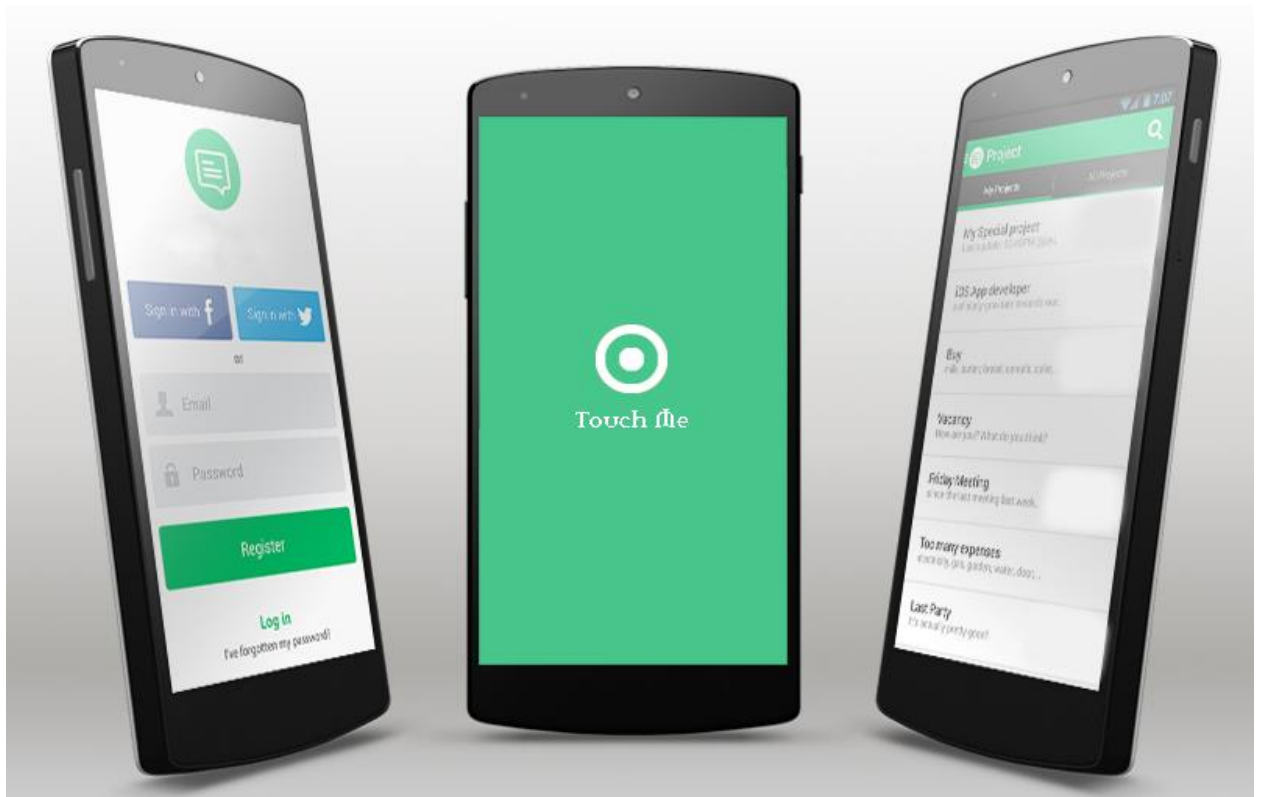
Settings



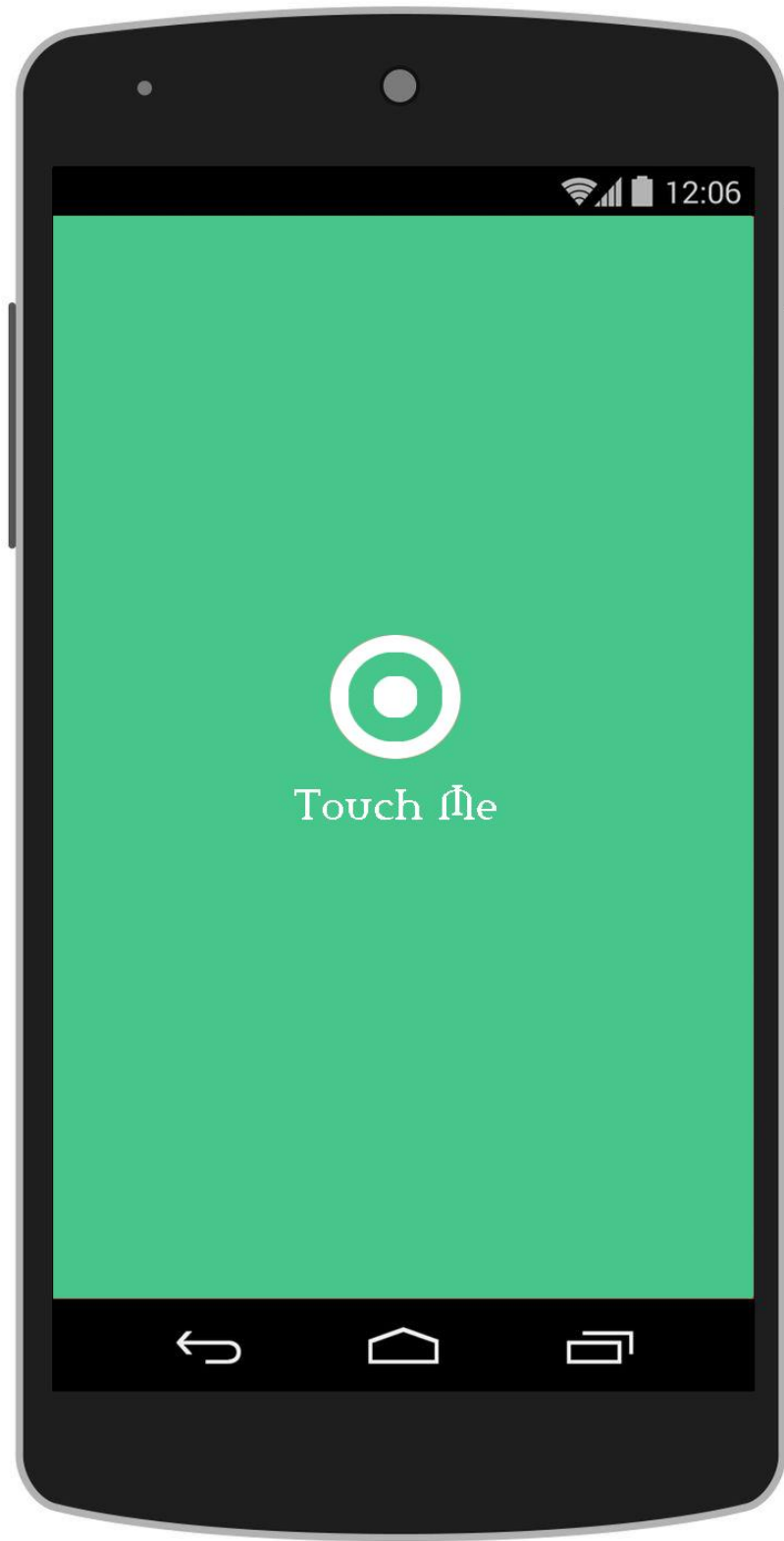
Pattern



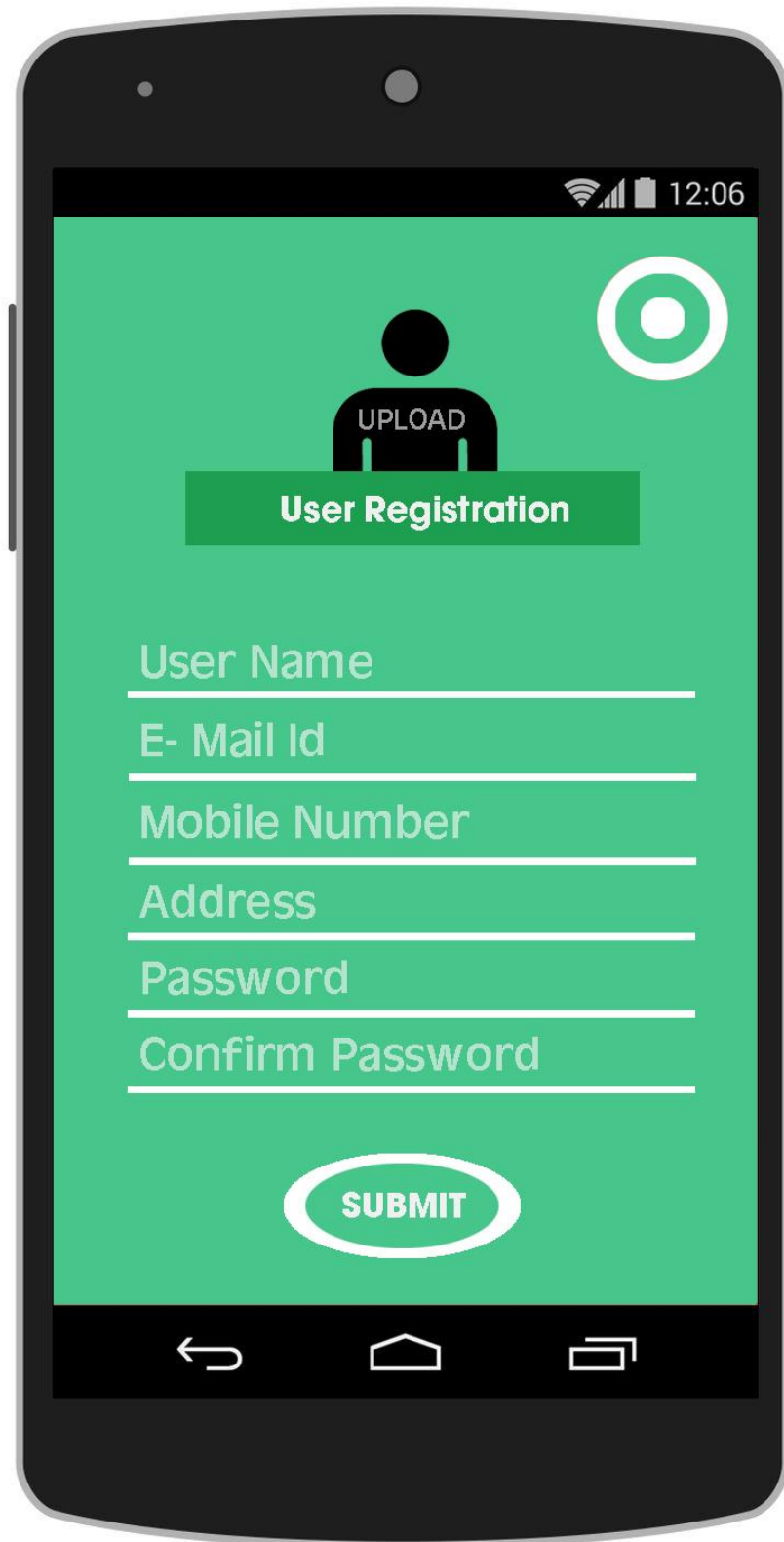
Application Overview



Front Page



First Time Registration



The image shows a mobile application interface for user registration. The screen has a green background. At the top, there is a status bar with a Wi-Fi icon, a battery icon, and the time 12:06. Below the status bar, there is a black silhouette of a person with the word "UPLOAD" written below it. To the right of the person icon is a white circular target icon. Below these elements is a green rectangular button with the text "User Registration" in white. Underneath the button are six white text input fields, each with a label above it: "User Name", "E- Mail Id", "Mobile Number", "Address", "Password", and "Confirm Password". At the bottom of the form is a white oval button with the text "SUBMIT" in black. The entire form is framed by a black border, and at the very bottom, there is a black navigation bar with three white icons: a back arrow, a home house icon, and a recent apps icon.

12:06

UPLOAD

User Registration

User Name

E- Mail Id

Mobile Number

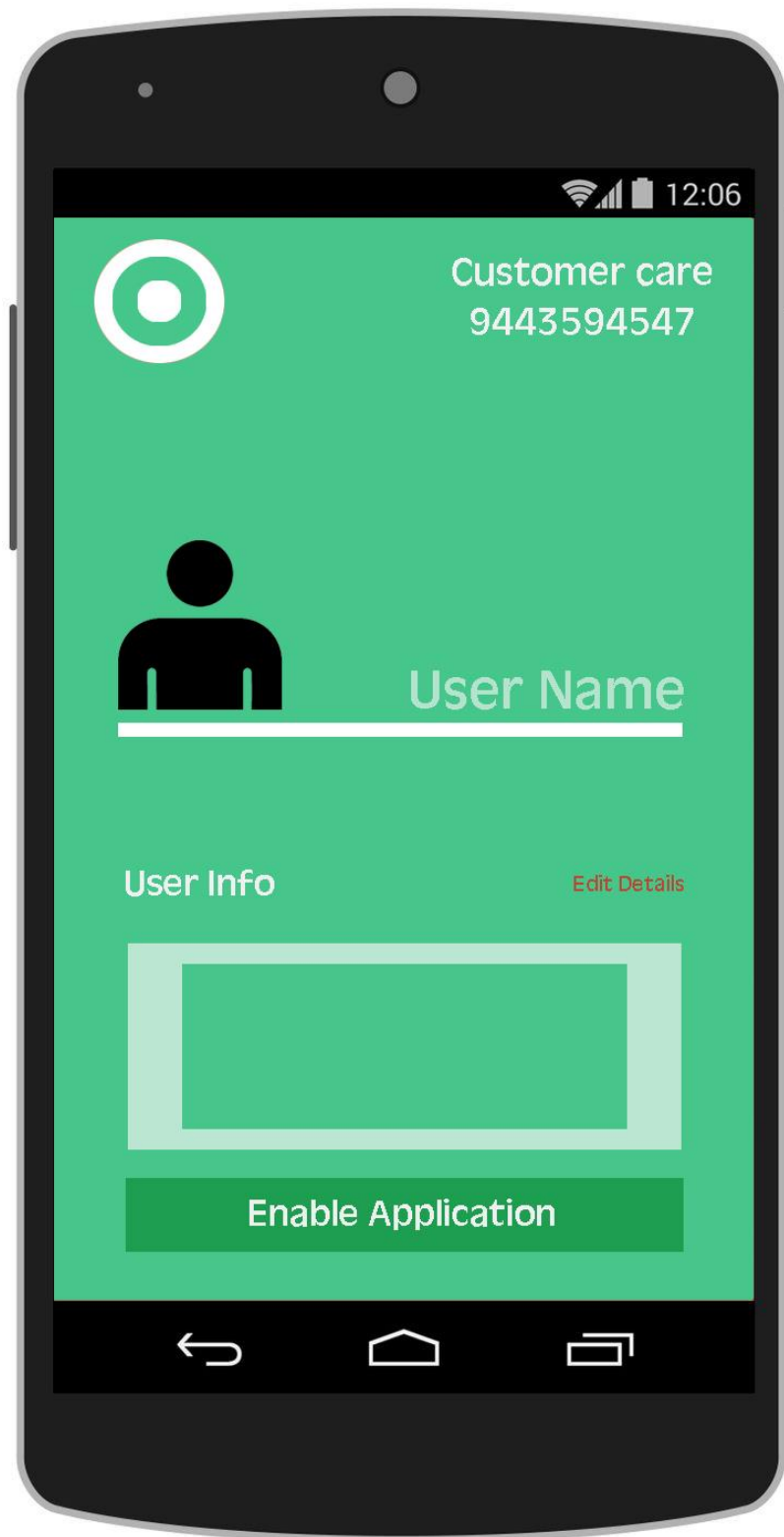
Address

Password

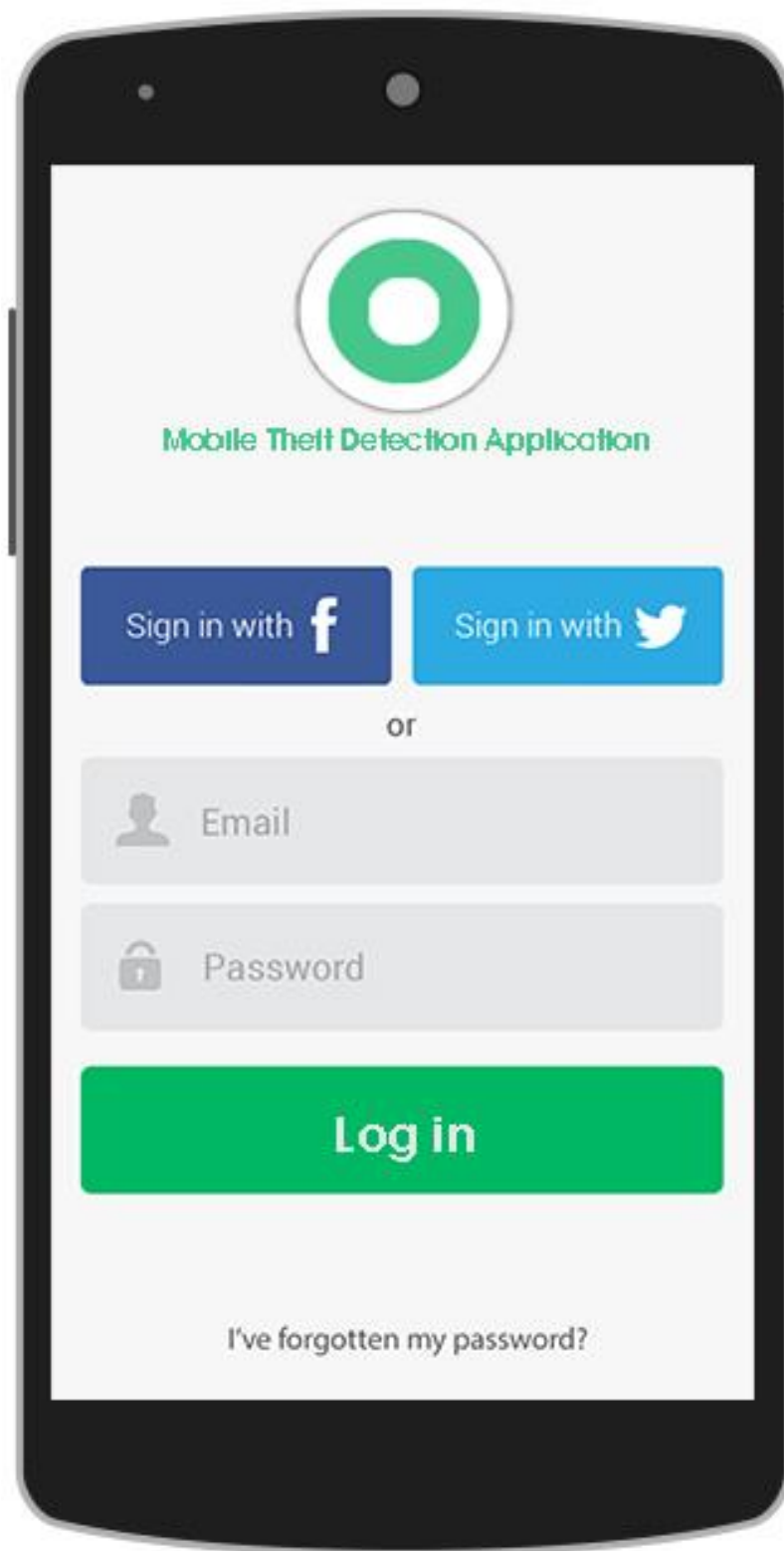
Confirm Password

SUBMIT

Home Page for Enable / Disable Application & Customer care



First Time Registration



GPS Tracker

