"MyHelper (Access your phone from anywhere without Internet)"

A project Report

Submitted in fulfillment of the

Requirements for the award of the degree of

Bachelor of Technology (B.Tech)

Submitted by

Deepak Kumar (16EIACS019)

Manish Kumar (16EIACS040)

Akshay Maheshwari (16EIACS302)

Under the supervision of

Ms. Deepika Sharma

Assistant Professor of CSE

Department of Computer Science & Engineering

Institute of Engineering & Technology, Alwar

Rajasthan Technical University, Kota

September, 2020

INSTITUTE OF ENGINEERING & TECHNOLOGY, ALWAR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CANDIDATE'S DECLARATION

We hereby declare that the work, which is being presented in this report, entitled **"MyHelper (Access your phone from anywhere without Internet)"** in fulfilment for the award of Degree of **Bachelor of Technology in Computer Science & Engineering** and submitted to **"Institute of Engineering & Technology, Alwar",** affiliated to, **Rajasthan Technical University,** Kota is an authentic record of our own work carried out under the supervision of **Ms. Deepika Sharma,** Department of Computer Science Engineering, IET Alwar.

We have not submitted the matter presented in this report anywhere for the award of any other degree.

Deepak Kumar (16EIACS019)

Manish Kumar (16EIACS040)

Akshay Maheshwari (16EIACS302)

(B.Tech. CSE)

IET, Alwar

Counter Signature

Ms. Deepika Sharma

(Assistant Professor, CSE)

INSTITUTE OF ENGINEERING & TECHNOLOGY, ALWAR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the report entitled "**MyHelper (Access your phone from anywhere without Internet)**" submitted by **Deepak Kumar, Manish Kumar & Akshay Maheshwari** to "**Institute of Engineering & Technology, Alwar**", affiliated to **Rajasthan Technical University, Kota**, for the award of the Degree of **Bachelor of Technology** in **Computer Science & Engineering** is a record of bonafide work carried out by them under our supervision. In our opinion, the Report has reached the standards of fulfilling the requirements of the regulations to the degree.

Dr. Rohit Kumar Singhal	**Ms. Deepika Sharma**
(Professor and HOD)	**(Assistant Professor)**
Department of Computer Science & Engineering	**Department of CSE**
Institute of Engineering & Technology, Alwar	**IET, Alwar**

ACKNOWLEDGEMENT

We wish to express deep sense of gratitude to **Dr. V.K. Agarwal** (Chairman, IET group of Institutions), **Dr. Manju Agarwal** (Executive Director, IET Group of Institutions), **Prof. Dr. Anil Kumar Sharma** (Principal, Institute of Engineering & Technology, Alwar) for their constant support and time to time very useful help and suggestions and **Dr. Rohit Kumar Singhal** (Professor and HOD CSE, IET, Alwar) for his support and resources.

Also, **Dr. Shadab Ali** Assistant Professor and Project-In-Charge**, Ms. Deepika Sharma** (Assistant Professor and Project-Guide) have played a vital role for our guidance, encouragement, help and useful suggestions throughout. Their untiring and painstaking efforts, methodological approach and individual help made it possible for me to complete this work in time. We consider myself very fortunate for having been associated with the scholar like them. Their affection, guidance and scientific approach served a veritable incentive for completion of this work.

This acknowledgement will remain incomplete if we fail to express our deep sense of obligation to our parents and God for their consistent blessings and encouragement.

Deepak Kumar (16EIACS019)

Manish Kumar (16EIACS040)

Akshay Maheshwari (16EIACS302)

Department of Computer Science & Engineering

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

PDA	:	Personal Digital Assistant
ADT	:	Android Development Tool
SDK	:	Software Development Kit
OTP	:	One Time Password
GUI	:	Graphical User Interface
JVM	:	Java Virtual Machine
GPL	:	General Public License
API	:	Application Programming Interfaces
RFP	:	Request for Proposal
PDA	:	Personal Digital Assistant
JDBC	:	Java Database Connectivity
RMI	:	Remote Method Invocation
JAR	:	Java Archive
JFC	:	Java Foundation Classes
OOPS	:	Object Oriented Programming
WORA	:	Write Once and Run Anywhere
JRE	:	Java Runtime Environment
IDE	:	Integrated Development Environment
SLDC	:	Software Development Life Cycle
IT	:	Information Technology
SRS	:	Software Requirement Specification
DFD	:	Data Flow Diagram

Abstract

Objectives:

- Safety: Device gets safer as the app reduces the chances of losing the mobilephone.
- Peace of mind: Getting panicked is quite common when your phone is lost and is on silent. This app reduces the panic and thereby ensuring peace of mind.

Methods/Analysis:

- This mobile application MyHelper controls the notification.
- Android is a comprehensive platform, which means it is a complete software stack for a mobile device.
- Day by day, the technology is becoming an increasingly important part of our everyday life. Using new technology, everything seems easier.

Findings:

- This app can also be used to fetch contacts from your phone via text message.
- This app can be used to change notification setting from silent to ring.

Novelty/Improvements:

In this project we can also add many other options for quick notifications for many tasks to perform by mobile phone without accessing the internet because this project is based on accessing the mobile without using internet.

1. INTRODUCTION

1.1 Overview:

Android is a comprehensive platform, which means it is a complete software stack for a mobile device. The goal of the project is to develop a android app that can be deployed on a mobile and used to change notification setting. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Android is a software platform and operating system for mobile devices based on the Linux operating system and developed by Google and the Open Handset Alliance. It allows developers to write managed code in a Java-like language that utilizes Google-developed Java libraries, but does not support programs developed in native for mobile devices devoted to advancing to show the different mechanism of the open standards for code. The unveiling of the Android platform on 5 November 2007 was announced with the founding of the Open Handset Alliance, a consortium of 34 hardware, software and telecom companies devoted to advancing open standards for mobile devices. When released in 2008, most of the Android platform will be made available under the Apache free-software and open-source license.

Modern hand held devices such as smart phones and PDAs have become increasingly powerful in recent years. Dramatic breakthroughs in processing power along with the number of extra features included in these devices have opened the doors to a wide range of commercial possibilities. In particular, most cell phones regularly include cameras, processors comparable to PCs from only a few years ago, and Internet access. However, even with all these added abilities, there are few applications that allow much passing of the environmental information and location based services. As mobile devices become more like PCs they will come to replace objects we tend to carry around such as checkbooks, credit cards, cameras, planners, mp3 players, etc. In short, we will be using them to accomplish our daily tasks. One application that falls into this category is the Online Shopper Application developed for the Google Android Phones. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Android is a software platform and operating system for mobile devices based on the Linux operating system and developed by Google and the Open Handset Alliance. It allows developers to write managed code in a Java-like language that utilizes Google-developed Java libraries, but does not support programs developed in native code. The unveiling of the Android platform on 5 November 2007 was announced with the founding of the Open Handset Alliance, a

consortium of 34 hardware, software and telecom companies devoted to advancing open standards for mobile devices. When released in 2008, most of the Android platform will be made available under the Apache free-software and open-source license.

The Project is developed in Java Programming Language having the ADT (Android Development Tool) plug-in which have the definitions for the android libraries. We use the Android Software Development Kit (SDK) which includes a variety of custom tools that help us develop mobile applications on the Android platform. The most important of these are the Android Emulator and the Android Development Tools (ADT) plug-in.

1.2 Goals and objectives:

Day by day, the technology is becoming an increasingly important part of our everyday life. Using new technology, everything seems easier. Thus, losing one's phone can be a bizarre situation and that too on silent. So our app solves this problem.

Our Objectives are:-

- Safety: Device gets safer as the app reduces the chances of losing the mobile phone.
- Peace of mind: Getting panicked is quite common when your phone is lost and is on silent. This app reduces the panic and thereby ensuring peace of mind.

1.3 Scope of the project:

Android has been criticized for not being all open-source software despite what was announced by Google. Parts of the SDK are proprietary and closed source, and some believe this is so that Google can control the platform. Software installed by end-users must be written in Java, and will not have access to lower level device APIs. This provides end-users with less control over their phone's functionality than other free and open source phone platforms, such upcoming applications and mobile services as in cash; we follow the same process Open Moko.

With all upcoming applications and mobile services Google Android is stepping into the next level of Mobile Internet. Android participates in many of the successful open source projects. That is, architect the solution for participation and the developers will not only come but will play well together. This is notable contrast with Apple and other companies, where such architecture of the evaluation of participation is clearly tobelated.

The first Android based official devices may well be launched sometime in the early half of 2009. Obviously, that's an age away when it comes to handset design, and Android may well find itself competing against the forthcoming Nokia touch screen phones and maybe official devices may well be world one of the best i-phone.

1. This app can also be used to fetch contacts from your phone via text message.

2. This app can be used to change notification setting from silent to ring.

1.4 Myhelper:

Therein comes the use of 'MyHelper'. It's your personal offline assistant to help you with the common problems faced in daily life.

So What is my helper – works completely without Internet

1. Problem 1: Forgot your phone at home? Want to get the contact's number to make an important call?
   Solution 1: Just send an sms to your phone with contact name and you will get number back as an sms.
2. Problem 2: Did you ever misplace your phone at home and you made the whole world upside down to search for it?
   Solution 2: MyHelper will help you change the sound profile of the phone from silent to normal mode so you could search for it easily.
3. Problem 3:Lost your phone want to know where is it exactly?
   Solution 3: MyHelper will send you an sms with your phones current location immediately.
4. Problem 4: Want to lock your phone?
   Solution 4: MyHelper will help you lock your screen immediately

Working: We provide you a very simple offline 800kb packet of Android app which will solve all the issues. All you need to do is simply send an SMS from any basic phone with the passcode you set on your Android app. The app works totally in background, A user just has to set a passcode and he is done with it, Now he does not need to do anything else after that, the app will automatically detect Incoming message filter it and search for the passcode on success reads command and perform the task as per user requirement.

1.5 Features of project:

1. An easy way to access your phone's contact at anytime, anywhere just through a simple SMS.
2. An sms can help you change the sound profile of your phone(silent to normal) without Internet
3. Remote Access without the Internet.
4. Track your phone through an sms
5. Hassle-Free as no OTP and ID PASSWORD is required
6. Fast
7. User friendly
8. Helps during Emergency.

2. SYSTEM REQUIREMENT SPECIFICATION

2.1 Non Functional Requirements:

2.1.1 Operational Requirements:

90% of users will not need to read the user manual to be able to use the application. General users who uses Android app can use the application smoothly.

2.1.2 Performance Requirements:

- Response Time -Response time should be less than a seconds.
- Work Load - The system should be capable of handling multiple users at a time.

2.1.3 Security Requirements:

- Access to application is limited to specific users.
- Only authenticated user can control the application.

2.2 System Requirements Specification:

This android app interacts with the user through G.U.I. The interface is simple, easy to handle and self-explanatory. Once opened, user will easily come into the flow with the application and easily uses all interfaces properly.

2.2.1 Hardware Requirement:

Minimum requirements will be as follows:

1. Processor: Intel core 3 or above
2. RAM: 256 MB
3. Hard disk 40 GB
4. Android Device

2.2.2 Software Requirement

1. Android 3.2(Honeycomb) or above
2. Android Studio

3. INTRODUCTION ABOUT JAVA

3.1 Introduction:

Java is a set of computer software and specifications developed by Sun Microsystems, which was later acquired by the Oracle Corporation, that provides a system for developing application software and deploying it in a cross-platform computing environment. Java is used in a wide variety of computing platforms from embedded devices and mobile phones to enterprise servers and supercomputers. While they are less common than standalone Java applications, Java applets run in secure, sandboxed environments to provide many features of native applications and can be embedded in HTML pages.

Writing in the Java programming language is the primary way to produce code that will be deployed as byte code in a Java Virtual Machine (JVM), byte code compilers are also available for other languages, including Ada, JavaScript, Python, and Ruby. In addition, several languages have been designed to run natively on the JVM, including Scala, Clojure and Apache Groovy. Java syntax borrows heavily from C and C++, but object-oriented features are modeled after Smalltalk and Objective-C. Java eschews certain low-level constructs such as pointers and has a very simple memory model where every object is allocated on the heap and all variables of object types are references. Memory management is handled through integrated automatic garbage collection performed by the JVM. On November 13, 2006, Sun Microsystems made the bulk of its implementation of Java available under the GNU General Public License (GPL).

The latest version is Java 8, the only supported (with e.g. security updates) version as of 2016. Oracle states that using older versions presents serious risks due to unresolved security issues. The heart of the Java platform is the concept of a "virtual machine" that executes Java byte code programs. This byte code is the same no matter what hardware or operating system the program is running under. There is a JIT (Just In Time) compiler within the Java virtual machine, or JVM. The JIT compiler translates the Java byte code into native processor instructions at run-time and caches the native code in memory during execution.

3.2 History of JAVA:

The Java platform and language began as an internal project at Sun Microsystems in December 1990, providing an alternative to the C++/C programming languages. Engineer Patrick Naughton had become increasingly frustrated with the state of Sun's C++ and C application programming interfaces (APIs) and tools. While considering moving to NeXT, Naughton was offereda chance to work on new technology, and thus the Stealth Project started. The Stealth Project was soon renamed to the Green Project, with James Gosling and Mike Sheridan joining Naughton. Together with other engineers, they began work in a small office on Sand Hill Road in Menlo Park, California. They aimed to develop new technology for programming next-generation smart appliances, which Sun expected to offer major new opportunities.

The team originally considered using C++, but rejected it for several reasons. Because they were developing an embedded system with limited resources, they decided that C++ needed too much memory and that its complexity led to developer errors. The language's lack of garbage collection meant that programmers had to manually manage system memory, a challenging and error-prone task. The team also worried about the C++ language's lack of portable facilities for security, distributed programming, and threading. Finally, they wanted a platform that would port easily to all types of devices. Bill Joy had envisioned a new language combining Mesa and C. In a paper calledFurther, he proposed to Sun that its engineers should produce an object-oriented environment based on C++. Initially,Gosling attempted to modify and extend C++ (a proposed development that he referred to as "C++ ++ --") but soon abandoned that in favor of creating a new language, which he called Oak, after the tree that stood just outside his office.

By the summer of 1992, the team could demonstrate portions of the new platform, including the Green OS, the Oak language, the libraries, and the hardware. Their first demonstration, on September 3, 1992, focused on building a personal digital assistant (PDA) device named Star7 that had a graphical interface and a smart agent called "Duke" to assist the user. In November of that year, the Green Project was spun off to become First person, a wholly owned subsidiary of Sun Microsystems, and the team relocated to Palo Alto, California. The Firstperson team had an interest in building highly interactive devices, and when Time Warner issued a request for proposal (RFP) for a set-top box, Firstperson changed their target and responded with a proposal for a set-top box platform. However,

the cable industry felt that their platform gave too much control to the user, so Firstperson lost their bid to SGI. An additional deal with The 3DO Company for a set-top box also failed to materialize.

Table 3.1 Java version history

Java Version	**Code Named**	**Features**
JDK Alpha and Beta	Alpha and Beta	Released in 1995 and was highly unstable. The supplied Java web browser was named Web Runner.
JDK 1.0	Oak	Released on January 23, 1996 and called Oak. First stable version, JDK 1.0.2, is called Java.
JDK 1.1	-	JavaBeans. JDBC (Java Database Connectivity). RMI (Remote Method Invocation). Reflection (only Introspection). Inner Classes. Extensive retooling of AWT event model.
J2SE 1.2	Playground	Collection Framework. JIT (Just In Time) compiler. Java String memory map for constants. Jar Signer for signing Java Archive (JAR) file. Policy tool for granting access to system resources. Java Foundation Classes (JFC) which consist of Swing 1.0, Drag and Drop, and Java 2D class libraries. Java Plug-in.

		Scrollable result sets, BLOB, CLOB, batch update, user defined types in JDBC. Audio Support in Applets.
J2SE 1.3	Kestrel	Java Sound. Java Indexing. A huge list of enhancement in almost all java area.
J2SE 1.4	Merlin	XML Processing. Java Print Service. Logging API. Java Web Start. JDBC 3.0 API. Assertions. Preference API. Chained Exception. IPV6 Support. Regular Expression. Image I/O API.
J2SE 5.0	Tiger	Generics. Enhanced for loop. Autoboxing/Unboxing. Typesafe Enums. Varargs. Static Import. Metadata (Annotation). Instrumentation.
Java SE6	Mustang	Scripting language support.

		JDBC 4.0 API. Java Compiler API. Pluggable Annotation. Native PKI, Java GSS, Kerberos and LDAP support. Integrated Web Services.
Java SE7	Dolphin	String in switch statement. Type inference for Generic Instance Creation. Multiple Exception Handling. Support for Dynamic Language. Try with Resources. Java nio package. Binary Literals, underscore in literals. Diamond Syntax. Automatic null handling.
Java SE8	No code name	Lambda Expression. Pipelines and Stream. Date and Time API. Default Methods. Type Annotation. Nashhorn JavaScript Engine. Concurrent Accumulator. Parallel operations. PermGen Error Removed. TLS SNI.

3.3 Platform:

The Java platform is a suite of programs that facilitate developing and running programs written in the Java programming language. A Java platform will include an execution engine (called a virtual machine), a compiler and a set of libraries; there may also be additional servers and alternative libraries that depend on the requirements. Java is not specific to any processor or operating system as Java platforms have been implemented for a wide variety of hardware and operating systems with a view to enable Java programs to run identically on all of them. Different platforms target different classes of device and application domains:

- Java Card: A technology that allows small Java-based applications (applets) to be run securely on smart cards and similar small-memory devices.
- Java ME (Micro Edition): Specifies several different sets of libraries (known as profiles) for devices with limited storage, display, and power capacities. It is often used to develop applications for mobile devices, PDAs, TV set-top boxes, and printers.
- Java SE (Standard Edition): For general-purpose use on desktop PCs, servers and similar devices.
- Java EE (Enterprise Edition): Java SE plus various APIs which are useful for multitier client–server enterprise applications.

3.3.1 Java Virtual Machine:

The heart of the Java platform is the concept of a "virtual machine" that executes Java bytecode programs. This bytecode is the same no matter what hardware or operating system the program is running under. There is a JIT (Just In Time) compiler within the Java Virtual Machine, or JVM. The JIT compiler translates the Java bytecode into native processor instructions at run-time and caches the native code in memory during execution.

3.3.2 Class Libraries:

The Java class libraries serve three purposes within the Java platform. First, like other standard code libraries, the Java libraries provide the programmer a well-known set of functions to perform common tasks, such as maintaining lists of items or performing complex string parsing. Second, the class libraries provide an abstract interface to tasks that would normally depend heavily on the hardware and operating system. Tasks such as network access and file access are often heavily intertwined with the distinctive

implementations of each platform. The libraries implement an abstraction layer in native OS code, then provide a standard interface for the Java applications to perform those tasks. Finally, when some underlying platform does not support all of the features a Java application expects, the class libraries work to gracefully handle the absent components, either by emulation to provide a substitute, or at least by providing a consistent way to check for the presence of a specific feature.

3.4 Features:

There is given many features of java. They are also known as java buzzwords. The Java Features given below are simple and easy to understand.

1. Simple
2. Object-Oriented
3. Platform independent
4. Secured
5. Robust
6. Architecture neutral
7. Portable
8. Dynamic
9. Interpreted
10. High Performance
11. Multithreaded
12. Distributed

3.4.1 Simple:

According to Sun, Java language is simple because:

- Syntax is based on C++ (so easier for programmers to learn it after C++).
- Removed many confusing and/or rarely-used features e.g., explicit pointers, operator overloading etc.
- No need to remove unreferenced objects because there is Automatic Garbage Collection in java.

3.4.2 Object Oriented:

Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior. Object-oriented programming (OOPs) is a methodology that simplify software development and maintenance by providing some rules.

Basic concepts of OOPs are:

1. Object
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

3.4.3 Platform Independent:

A platform is the hardware or software environment in which a program runs. There are two types of platforms software-based and hardware-based. Java provides software-based platform. The Java platform differs from most other platforms in the sense that it's a software-based platform that runs on top of other hardware-based platforms. It has two components:

- Runtime Environment
- API(Application Programming Interface)

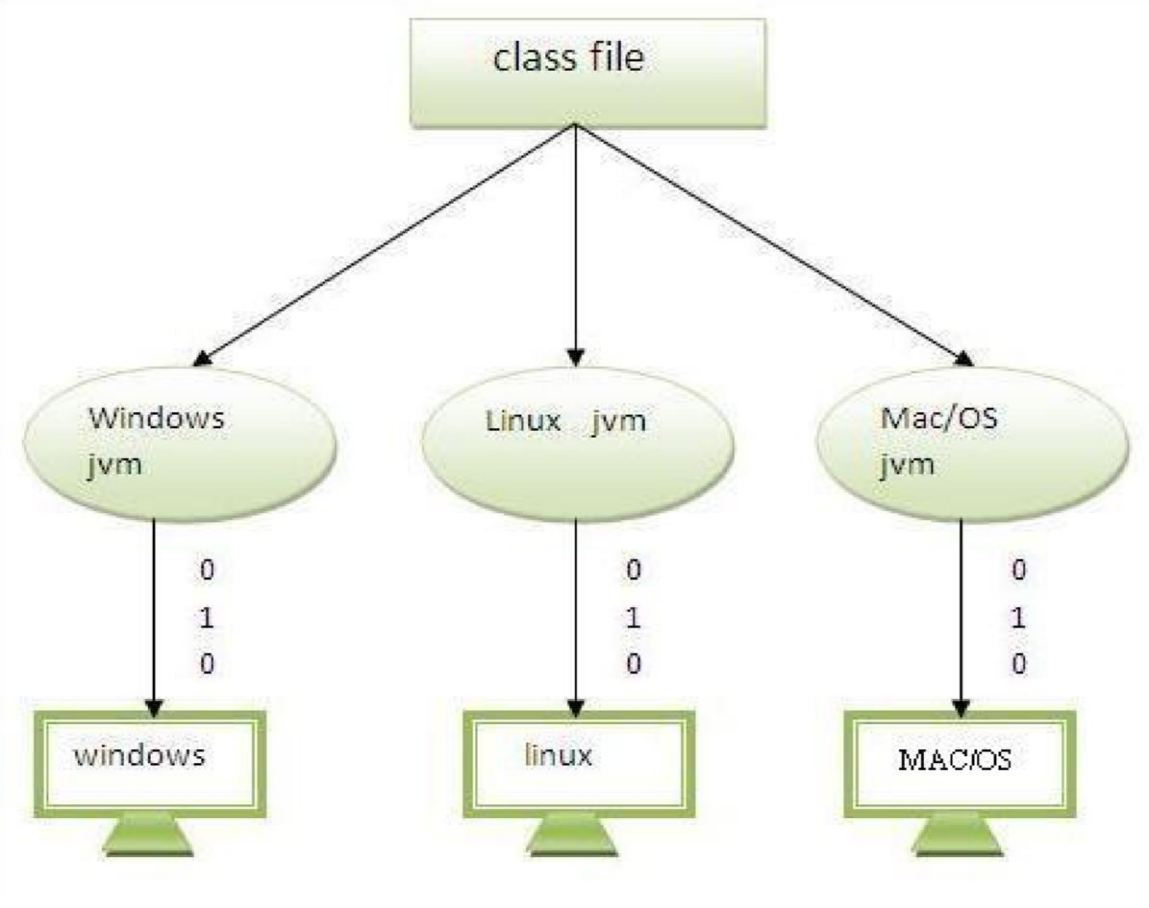

Fig. 3.1 Java Class File

Java code can be run on multiple platforms e.g. Windows, Linux, Sun Solaris, Mac/OS etc. Java code is compiled by the compiler and converted into bytecode. This bytecode is a platform independent code because it can be run on multiple platforms i.e. Write Once and Run Anywhere (WORA).

3.4.4 Secured:

- Java is secured because:
- No explicit pointer
- Programs run inside virtual machine sandbox.
- Classloader- adds security by separating the package for the classes of the local file system from those that are imported from network sources.
- Bytecode Verifier- checks the code fragments for illegal code that can violate access right to objects.

- Security Manager- determines what resources a class can access such as reading and writing to the local disk.
- These security are provided by java language. Some security can also be provided by application developer through SSL, JAAS, cryptography etc.

3.4.5 Robust:

Robust simply means strong. Java uses strong memory management. There are lack of pointers that avoids security problem. There is automatic garbage collection in java. There is exception handling and type checking mechanism in java. All these points makes java robust.

3.4.6 Architectural-Neutral:

There is no implementation dependent features e.g. size of primitive types is set.

3.4.7 Portable:

We may carry the java bytecode to any platform.

3.4.8 High Performance:

Java is faster than traditional interpretation since byte code is "close" to native code still somewhat slower than a compiled language (e.g., C++)

3.4.9 Distributed:

We can create distributed applications in java. RMI and EJB are used for creating distributed applications. We may access files by calling the methods from any machine on the internet.

3.4.10 Multi Threaded:

A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multithreading is that it shares the same memory. Threads are important for multi-media, Web applications etc.

3.5 Licensing:

The source code for Sun's implementations of Java (i.e. the de facto reference implementation) has been available for some time, but until recently, the license terms severely restricted what could be done with it without signing (and generally paying for) a contract with Sun. As such these terms did not satisfy the requirements of either the Open Source Initiative or the Free Software Foundation to be considered open source or free software, and Sun Java was therefore a proprietary platform. While several third-party projects (e.g. GNU Class path and Apache Harmony) created free software partial Java implementations, the large size of the Sun libraries combined with the use of clean room methods meant that their implementations of the Java libraries (the compiler and VM are comparatively small and well defined) were incomplete and not fully compatible. Sun announced inJavaOne2006 that Java would become free and open source software, and on October 25, 2006, at the Oracle OpenWorldconference, Jonathan I. Schwartz said that the company was set to announce the release of the core Java Platform as free and open source software within 30 to 60 days.

3.6 Usage:

Its uses are following

3.6.1 Desktop Use:

Some Java applications are in fairly widespread desktop use, including the NetBeans and Eclipse integrated development environments, and file sharing clients such as LimeWire and Vuze. Java is also used in the MATLAB mathematics programming environment, both for rendering the user interface and as part of the core system. Java provides cross platform user interface for some high end collaborative applications like Lotus Notes. According to Oracle, the Java Runtime Environment is found on over 850 million PCs. Microsoft has not bundled aJava Runtime Environment (JRE) with its operating systems since Sun Microsystems sued Microsoft for adding Windows-specific classes to the bundled Java runtime environment, and for making the new classes available through Visual J++. Apple no longer includes a Java runtime with OS X as of version 10.7, but the system prompts the user to download and install it the first time an application requiring the JRE is launched. Many Linux distributions include the partially compatible free software package GNU Classpath and increasingly mostly compatible Iced Tea.

Some Java applications are in fairly widespread desktop use, including the NetBeans and Eclipse integrated development environments, and file sharing clients such as LimeWire and Vuze. Java is also used in the MATLAB mathematics programming environment, both for rendering the user interface and as part of the core system. Java provides cross platform user interface for some high end collaborative applications like Lotus Notes. Oracle plans to first deprecate the separately installable Java browser plug-in from the Java Runtime Environment in JDK 9 then remove it completely from a future release, forcing web developers to use an alternative technology.

3.6.2 Mobile Devices:

Java ME has become popular in mobile devices, where it competes with Symbian, BREW, and the .NET Compact Framework. The diversity of mobile phone manufacturers has led to a need for new unified standards so programs can run on phones from different suppliers – MIDP. The first standard was MIDP 1, which assumed a small screen size, no access to audio, and a 32kB program limit. The more recent MIDP 2 allows access to audio, and up to 64kB for the program size. With handset designs improving more rapidly than the standards, some manufacturers relax some limitations in the standards, for example, maximum program size. Google's Android operating system uses the Java language, but not its class libraries, therefore the Android platform cannot be called Java. Android executes the code on the ART VM (formerly the Dalvik VM up to Android 4.4.4) instead of the Java VM.

Java ME has become popular in mobile devices, where it competes with Symbian, BREW, and the .NET Compact Framework. The diversity of mobile phone manufacturers has led to a need for new unified standards so programs can run on phones from different suppliers – MIDP. The first standard was MIDP 1, which assumed a small screen size, no access to audio, and a 32kB program limit. The more recent MIDP 2 allows access to audio, and up to 64kB for the program size. With handset designs improving more rapidly than the standards, some manufacturers relax some limitations in the standards, for example, maximum program size. Google's Android operating system uses the Java language, but not its class libraries, therefore the Android platform cannot be called Java. Android executes the code on the ART VM (formerly the Dalvik VM up to Android 4.4.4) instead of the Java VM.

3.6.3 Web Server and Enterprise Use:

The Java platform has become a mainstay of enterprise IT development since the introduction of the Enterprise Edition in 1998, in two different ways:

1. Through the coupling of Java to the web server, the Java platform has become a leading platform for integrating the Web with enterprise backend systems. This has allowed companies to move part or all of their business to the Internet environment by way of highly interactive online environments (such as highly dynamic websites) that allow the customer direct access to the business processes (e.g. online banking websites, airline booking systems and so on). This trend has continued from its initial Web-based start:
    a. The Java platform has matured into an Enterprise Integration role in which legacy systems are unlocked to the outside world through bridges built on the Java platform. This trend has been supported for Java platform support for EAI standards like messaging and Web services and has fueled the inclusion of the Java platform as a development basis in such standards as SCA, XAM and others.
    b. Java has become the standard development platform for many companies' IT departments, which do most or all of their corporate development in Java. This type of development is usually related to company-specific tooling (e.g. a booking tool for an airline) and the choice for the Java platform is often driven by a desire to leverage the existing Java infrastructure to build highly intelligent and interconnected tools.
2. The Java platform has become the main development platform for many software tools and platforms that are produced by third-party software groups (commercial, open source and hybrid) and are used as configurable (rather than programmable) tools by companies. Examples in this category include Web servers, application servers, databases, enterprise service buses, business process management (BPM) tools and content management systems.
3. Enterprise use of Java has also long been the main driver of open source interest in the platform. This interest has inspired open source communities to produce a large amount of software, including simple function libraries, development frameworks (e.g. the Spring Framework, Apache Wicket, Dojo Toolkit, Hibernate), and open source implementations of standards and tools (e.g. Apache Tomcat, the Glass Fish application server, the Mule and Apache Service Mix Enterprise service buses).

3.7 Security:

The Java platform provides a security architecture which is designed to allow the user to run untrusted bytecode in a "sandboxed" manner to protect against malicious or poorly written software. This "sandboxing" feature is intended to protect the user by restricting access to certain platform features and APIs which could be exploited by malware, such as accessing the local filesystem, running arbitrary commands, or accessing communication networks. In recent years, researchers have discovered numerous security flaws in some widely used Java implementations, including Oracle's, which allow untrusted code to bypass the sandboxing mechanism, exposing users to malicious attacks. These flaws affect only Java applications which execute arbitrary untrusted bytecode, such as web browser plug-ins that run Java applets downloaded from public websites. Applications where the user trusts, and has full control over, all code that is being executed are unaffected.

4. ANDROID

4.1 Introduction:

Android is a mobile operating system developed by Google. It is based on Linux kernel. Android is primarily designed for mobile devices such as smartphones, tablets, Android televisions, wrist watches. All the above mentioned devices are touchscreen devices. Android uses touch inputs like swiping, tapping, pinching and reverse pinching on objects visible on screen along with a virtual keyboard. It could be also used in game consoles and other electronics.

The list of features in Android Operating System are mentioned below:

1. Messaging: SMS and MMS are the means of exchanging messages. Android Google Cloud Messaging(GCM) is also a part of Android Push Messaging service.
2. Screen Capture: Android supports capturing a screenshot by pressing the power and volume-down buttons at the same time.
3. Connectivity: Android supports connectivity technologies including GSM/EDGE, Wi-Fi, Bluetooth, LTE, CDMA, EV-DO, UMTS, NFC, IDEN and WiMAX.

4.2 History of Android:

The version history of the Android Mobile Operating System began with the release of the Android Alpha in November 5, 2007. The first commercial version, Android 1.0, was released in September 2008. Android is continually developed by Google and the Open Handset Alliance(OHA), and it has seen a number of Updates to its base operating system since the initial release. Versions 1.0 and 1.1 were not released under specific code names, but since April 2009's Android 1.5 "Cupcake", Android versions have had confectionery-themed code names. Each is in alphabetical order, with the most recent being Android 7.0 "Nougat", released in August 2016.

Fig. 4.1 Versions of Android

4.3 Activity in Android:

An activity represents a single screen with a user interface just like window or frame of Java.Android activity is the subclass of Context Theme Wrapper class. If you have worked with C, C++ or Java programming language then you must have seen that your program starts from **main ()** function. Very similar way, Android system initiates its program with in an **Activity** starting with a call on call back method. There is a sequence of call back methods that start up an activity and a sequence of call back methods that tear down an activity as shown in the below Activity life cycle

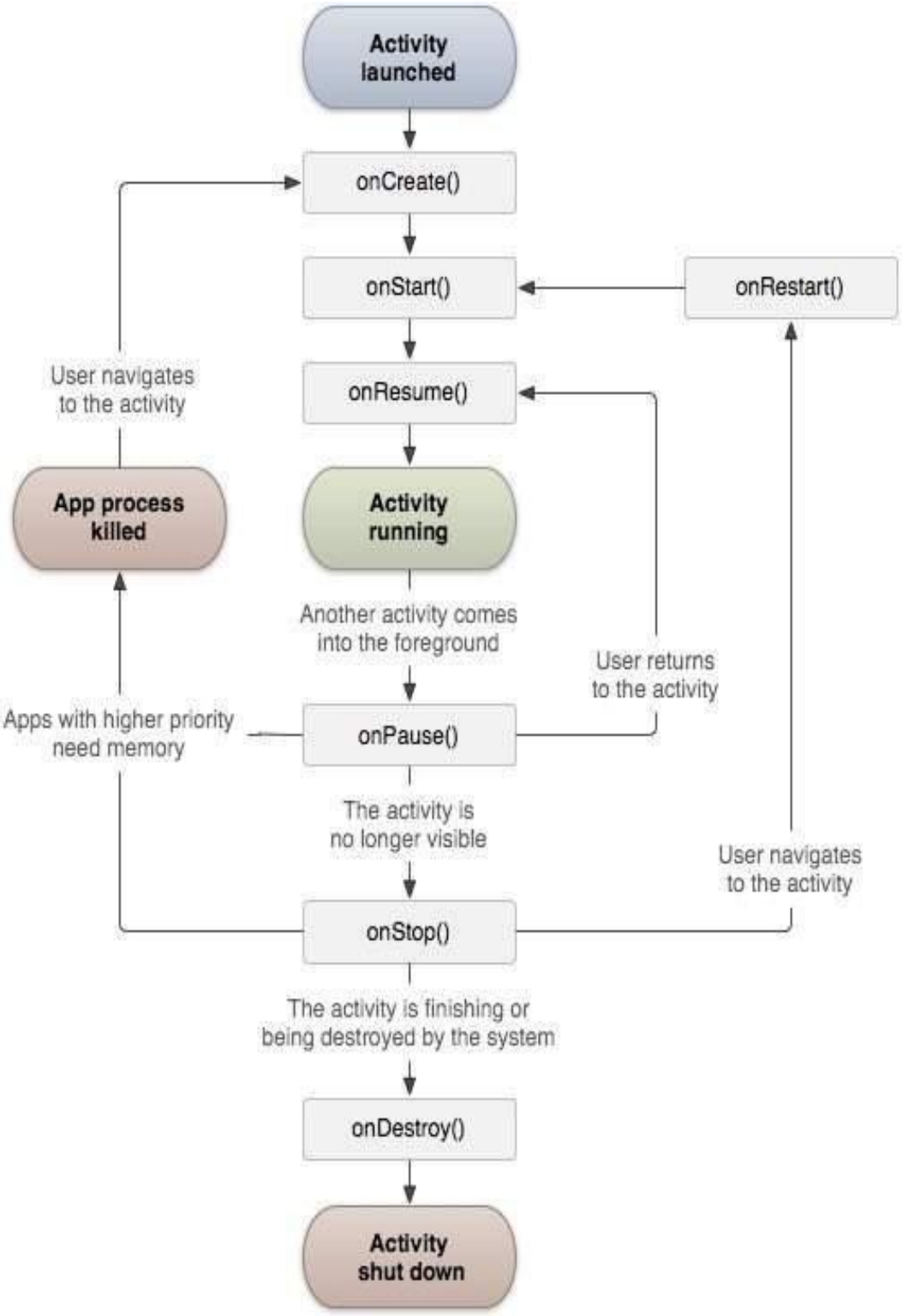

Fig. 4.2 Activity Diagram of Android

4.4 Android Environment:

4.4.1 Classes & Interfaces& Method required:

- **ImageButton**: It is used for on click purpose basic need of the image button is for go on the next activity by clicking on it.
- **Text view:** it is used for show text on screen
- **Button:** Button plays an important role for clicking. And for go on the another activity
- **Image view:** This is used for set image on the screen.
- **SeekTo(Position)**: This method take a integer and move to that particular second.
- **getCurrentDuration():**This method return the current position of song in millisecond.
- **getDuration():**This method return the total time duration of song in millisecond.
- **reset():** This method reset the media player.

4.3.2 Permission required:

- Read External Storage:-Use to read the external storage device.
- Write External Storage:- Use to perform write operation on external storage device.
- Google Map:- Use to do implement mapping function in the activity or in app with the help of Google.

4.5 Android Studio:

Android Studio is the official integrated development environment (IDE) for Android platform development. It was announced on May 16, 2013 at the Google I/O conference. Android Studio is freely available under the Apache License 2.0. Android Studio was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.Based on Jet Brains' IntelliJ IDEA software, Android Studio is designed specifically for Android development. It is available for download on Windows, Mac OS X and Linux, and replaced Eclipse Android Development Tools (ADT) as Google's primary IDE for native Android application development.

4.5.1 Features:

- Gradle-based build support.
- Android-specific refactoring and quick fixes.
- Lint tools to catch performance, usability, version compatibility and other problems.
- ProGuardintegration and app-signing capabilities.
- Template-based wizards to create common Android designs and components.
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations.
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Google Cloud Messaging and App Engine.

5. SOFTWARE DEVELOPMENT LIFE CYCLE

2.1 Introduction:

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

In other words, a life cycle model maps the various activities performed on a software product from its inception to retirement. Different life cycle models may plan the necessary development activities to phases in different ways. Thus, no element which life cycle model is followed, the essential activities are contained in all life cycle models though the action may be carried out in distinct orders in different life cycle models. During any life cycle stage, more than one activity may also be carried out.

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. The SDLC aims to produce a high quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates. Common methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, extreme programming and various types of agile methodology. Software development organizations implement process methodologies to ease the process of development. The SDLC process provides Information Technology (IT) project managers with the tools to help ensure successful implementation of systems or applications that satisfy strategic and business objectives. SDLC provides a mechanism to ensure that executive leadership, functional managers, and users sign-off on the requirements and implementation of the system. The process provides management with the capability to design, develop, and implement your intended system and ensure that its completed on time delivery and within budget. From one organization to another, or even project to project, there will be different needs and influencers promoting one development approach over another. All projects are unique by definition, and two of the most important roles a project manager has during project planning are project integration and tailoring - to ensure the right set of work is planned and executed to achieve the authorized scope of work

within the specified constraints of the project (e.g. scope, budget, resources, schedule, quality).With this in mind, the best SDLC approach for any given project has to be based on the project's constraints, the specific method preferences of the development team, the risk tolerance of executive management, and - perhaps most importantly - access to customers and end-users. The goals of SDLC are:Deliver quality systems that meet or exceed customer expectations when promised and within cost estimates.

- Provide a framework for developing quality systems using an identifiable, measurable, and repeatable process.
- Establish a project management structure to ensure that each system development project is effectively managed throughout its life cycle.
- Ensure that system development requirements are well defined and subsequently satisfied.

5.2 History:

The initial concepts of Software Development Lifecycles have been around 1960’s. According to Elliott (2004) the systems development life cycle (SDLC) can be considered to be the oldest formalized methodology framework for building information systems. The main idea of the SDLC has been "to pursue the development of information systems in a very deliberate, structured and methodical way, requiring each stage of the life cycle from inception of the idea to delivery of the final system, to be carried out rigidly and sequentially “within the context of the framework being applied. In the earliest days of computer programming, the only models for developing complex things came out of the construction and manufacturing industries. At the time, it seemed logical that the structured approaches used in those industries would equally apply to developing computing systems. For example, a developer would first need to understand the customer's requirements, then architect and design a solution, then develop the product, and finally test and deploy or turn-over the new product.This traditional development process logically progressed in a sequential manner from start to finish, with potentially some overlap for planning and preparation between the major phases of the development effort. In the last 50 years, many other factors affected the IT culture that demanded stringent methodologies. Entirely new fields of expertise in project management are now critical in the successful management of Information Technology that embody knowledge of the driving forces responsible for a successful operation.

5.3 Phases of SDLC:

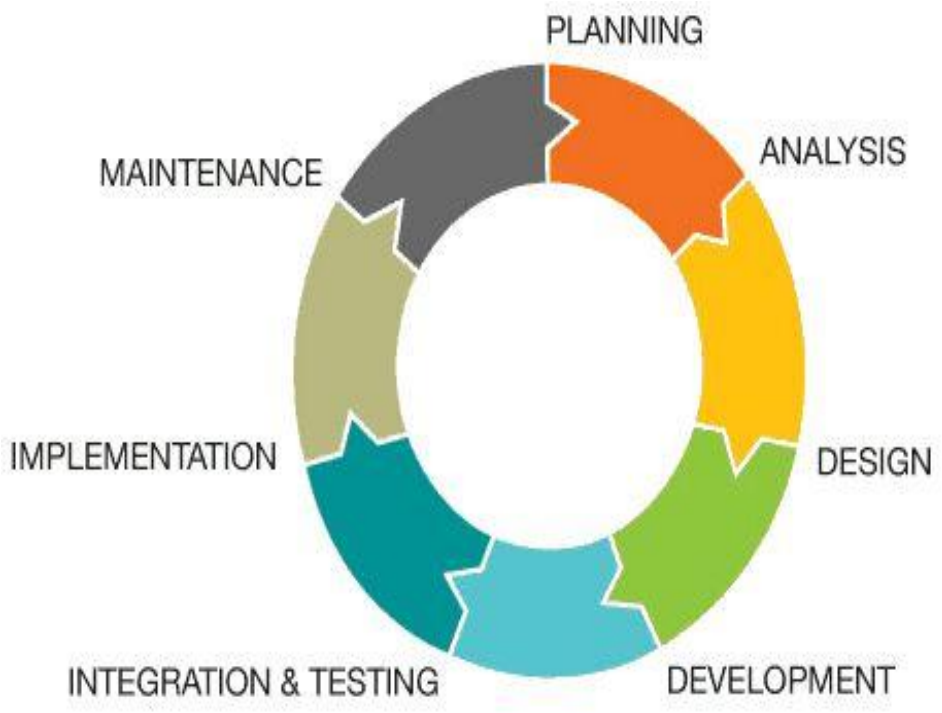

Fig. 5.1 Phases of SDLC

5.3.1 Requirement Analysis:

Requirement analysis is the most important and fundamental stage in SDLC. It is performed by the senior members of the team with inputs from the customer, the sales department, market surveys and domain experts in the industry. This information is then used to plan the basic project approach and to conduct product feasibility study in the economical, operational, and technical areas. After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

5.3.2 Defining Requirements:

Once the requirement analysis is done the next step is to clearly define and document the product requirements and get them approved from the customer or the market analysts. This is done through 'SRS' – Software Requirement Specification document which consists of all the product requirements to be designed and developed during the project life cycle.

5.3.3 Design:

Based on the requirements specified in SRS, usually more than one design approach for the product architecture is proposed and documented in a DDS - Design Document Specification. . System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model.

5.3.4 Coding and Implementation:

In this stage of SDLC the actual development starts and the product is built. The programming code is generated as per DDS during this stage. If the design is performed in a detailed and organized manner, code generation can be accomplished without much hassle. Different high level programming languages such as C, C++, Pascal, Java, and PHP are used for coding. The programming language is chosen with respect to the type of software being developed.

5.3.5 Testing:

This stage refers to the testing only stage of the product where products defects are reported, tracked, fixed and retested, until the product reaches the quality standards defined in the SRS. During this phase unit testing, integration testing, system testing, acceptance testing are done.

5.3.6 Deployment and Maintenance:

Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometime product deployment happens in stages as per the organizations' business strategy. After the product is released in the market, its maintenance is done for the existing customer base. Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the developed product is known as maintenance.

5.4 The stages of SDLC are as follows:

Stage1: Planning and requirement analysis:

- Requirement Analysis is the most important and necessary stage in SDLC.
- The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs in the industry.
- Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage.
- Business analyst and Project organizer set up a meeting with the client to gather all the data like what the customer wants to build, who will be the end user, what is the objective of the product. Before creating a product, a core understanding or knowledge of the product is very necessary.

For Example, A client wants to have an application which concerns money transactions. In this method, the requirement has to be precise like what kind of operations will be done, how it will be done, in which currency it will be done, etc.

- Once the required function is done, an analysis is complete with auditing the feasibility of the growth of a product. In case of any ambiguity, a signal is set up for further discussion.
- Once the requirement is understood, the SRS (Software Requirement Specification) document is created. The developers should thoroughly follow this document and also should be reviewed by the customer for future reference.

Stage2: Defining Requirements

- Once the requirement analysis is done, the next stage is to certainly represent and document the software requirements and get them accepted from the project stakeholders.
- This is accomplished through "SRS"- Software Requirement Specification document which contains all the product requirements to be constructed and developed during the project life cycle.

Stage3: Designing the Software

- The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project. This phase is the product of the last two, like inputs from the customer and requirement gathering.

Stage4: Developing the project

- In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code. Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

Stage5: Testing

- After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.
- During this stage, unit testing, integration testing, system testing, acceptance testing are done.

Stage6: Deployment

- Once the software is certified, and no bugs or errors are stated, then it is deployed.
- Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.
- After the software is deployed, then its maintenance begins.

Stage7: Maintenance

- Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.

5.5 SDLC Models:

There are various software development life cycle models defined and designed which are followed during software development process. These models are also referred as "Software Development Process Models". Each process model follows a Series of steps unique to its type, in order to ensure success in process of software development. Following are the most important and popular SDLC models followed in the industry:

- Waterfall Model
- Iterative Model
- Spiral Model
- Prototyping Model
- Rapid Application Development Model
- Agile Model

During the project development in my training, I have used waterfall model of software development.

5.6 Waterfall Model:

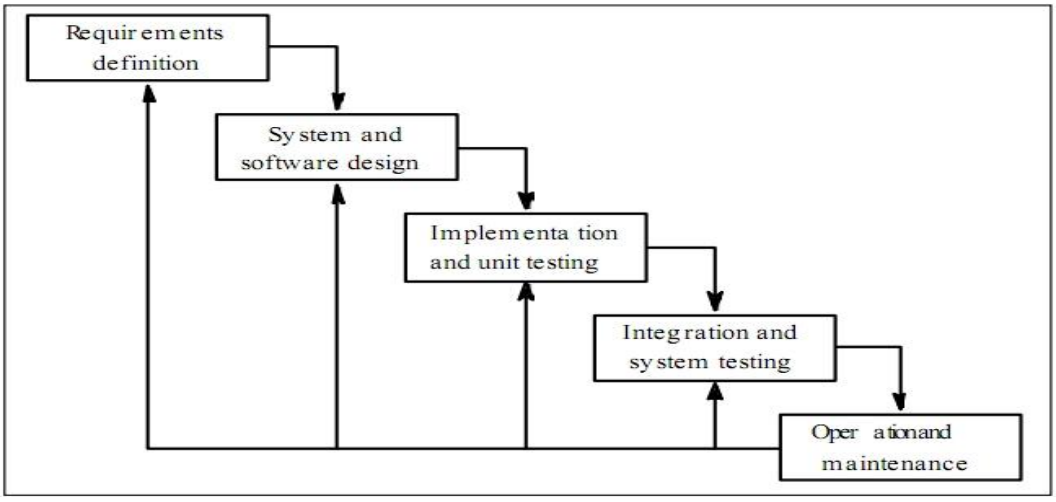

Fig. 5.2 Stages of Waterfall Model

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases. Waterfall model is the earliest SDLC approach that was used for software development . The waterfall Model illustrates the software development process in a

linear sequential flow; hence it is also referred to as a linear-sequential life cycle model. This means that any phase in the development process begins only if the previous phase is complete. In waterfall model phases do not overlap.

The sequential phases in waterfall model are:

- **Requirement Gathering and analysis:** All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.
- **System Design:** The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation:** With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.
- **Integration and Testing:** All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system:** Once the functional and non functional testing is done, the product is deployed in the customer environment or released into the market.
- **Maintenance:** There are some issues which come up in the client environment. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

5.6.1 Advantages:

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model .each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.

- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented.

5.6.2 Disadvantages:

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.

6. SYSTEM MODELS

6.1 Introduction:

System modelling helps the analyst to understand the functionality of the system and models are used to communicate with customers.

Different models present the system from different perspectives

- External perspective showing the system's context or environment;
- Behavioural perspective showing the behaviour of the system;
- Structural perspective showing the system or data architecture.

6.2 Data Flow Diagram:

Data flow diagrams are the basic building blocks that define the flow of data in a system to the particular destination and difference in the flow when any transformation happens. It makes whole procedure like a good document and makes simpler and easy to understand for both programmers and non-programmers by dividing into the sub process. The data flow diagrams are the simple blocks that reveal the relationship between various components of the system and provide high level overview, boundaries of particular system as well as provide detailed overview of system elements

The data flow diagrams start from source and ends at the destination level i.e., it decomposes from high level to lower levels. The important things to remember about data flow diagrams are: it indicates the data flow for one way but not for loop structures and it doesn't indicate the time factors. A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system. It is usually beginning with a context diagram as level 0 of the DFD diagram, a simple representation of the whole system. To elaborate further from that, we drill

down to a level 1 diagram with lower-level functions decomposed from the major functions of the system. This could continue to evolve to become a level 2 diagram when further analysis is required. Progression to levels 3, 4 and so on is possible but anything beyond level 3 is not very common. Please bear in mind that the level of detail for decomposing a particular function depending on the complexity that function.

The general notations for constructing a block diagram in this project are:

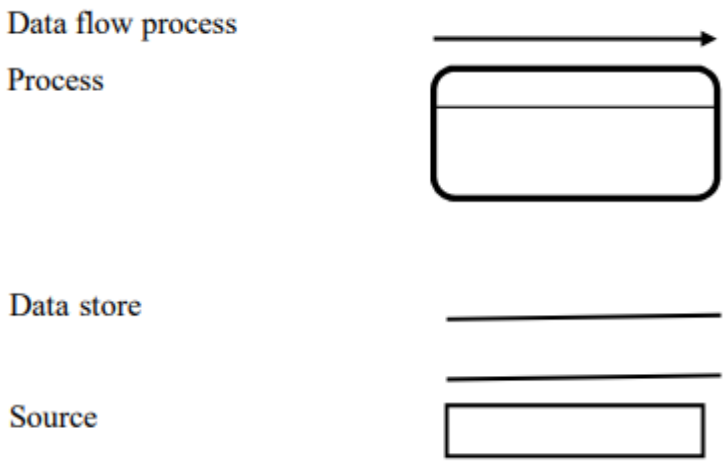

Fig. 6.1 Symbol of DFD

- **Data flow processes:** It will define the direction i.e., the data flow from one entity to another entity.
- **Process:** Process defines the source from where the output is generated for the specified input. It states the actions performed on image such that they are transformed, stored or distributed.
- **Data store:** It is the place or physical location where the data is stored after extraction from the data source.
- **Source:** It is the starting point or destination point of the data, stating point fromλ. where the external entity acts as a cause to flow the data towards destination.

Fig. 6.1 DFD for Set Passkey

Fig. 6.3 DFD for Send passkey to change notification

6.3 Class Diagram:

Class UML diagram is the most common diagram type for software documentation. Since most software being created nowadays is still based on the Object-Oriented Programming paradigm, using class diagrams to document the software turns out to be a common-sense solution. This happens because OOP is based on classes and the relations between them. In a nutshell, class diagrams contain classes, alongside with their attributes (also referred to as data fields) and their behaviors (also referred to as member functions). More specifically, each class has 3 fields: the class name at the top, the class attributes right below the name, the class operations/behaviors at the bottom. The relation between different classes (represented by a connecting line), makes up a class diagram.

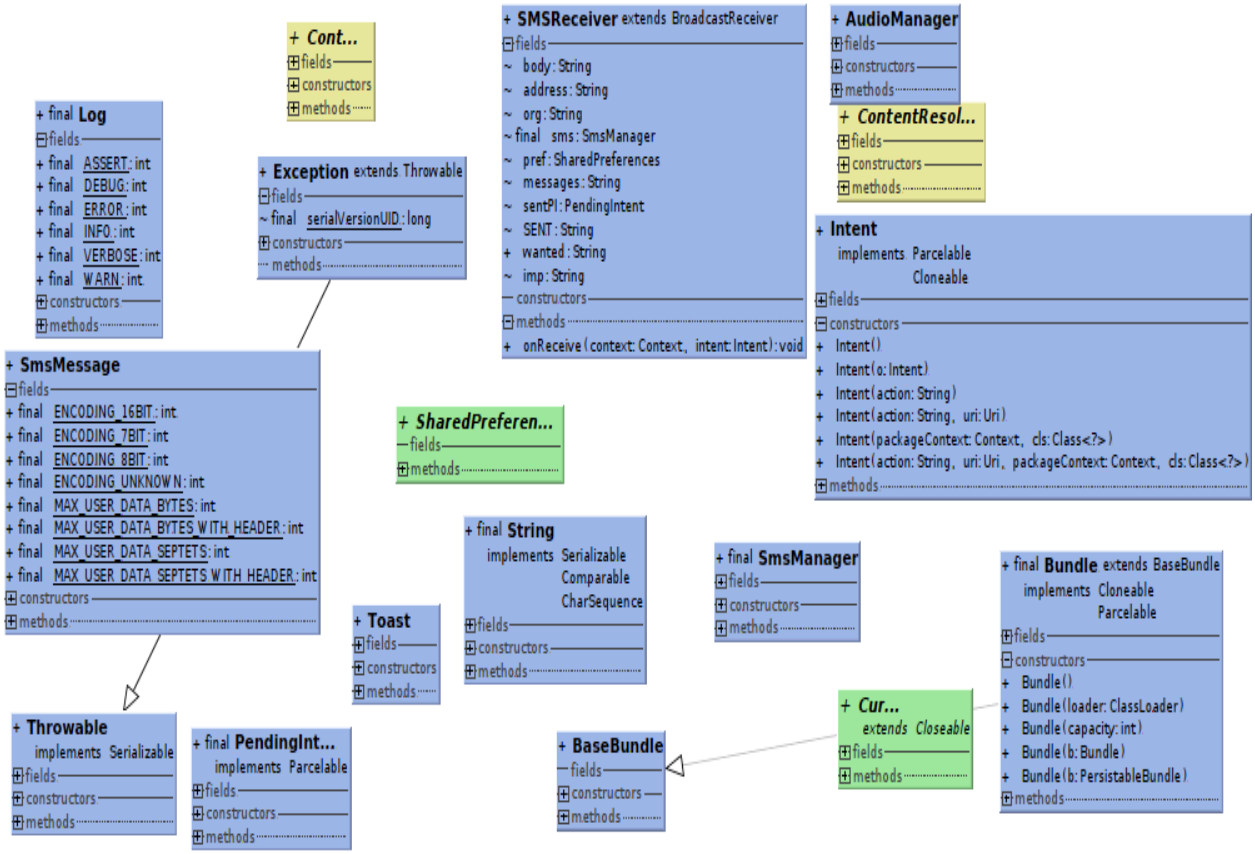

Fig. 6.4 Class Diagram

7. SOURCE CODE

7.1 Main Activity:

```
package com.personal.user.myhelper;

import android.app.Activity;

import android.content.Context;

import android.content.SharedPreferences;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;


public class MainActivity extends Activity {


 private EditText e1;

 private Button b1;




  @Override

  protected void onCreate(Bundle savedInstanceState)

  {
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
addListenerOnButton();
   }

   private void addListenerOnButton()
   {
      e1=(EditText)findViewById(R.id.e1);
      b1=(Button)findViewById(R.id.button);

      b1.setOnClickListener(new View.OnClickListener()
      {

         public void onClick(View view)
         {
            String s1=e1.getText().toString();
SharedPreferencesprefs=  getSharedPreferences("passcode",  Context.MODE_PRIVATE);
//to store the data
SharedPreferences.Editor e=prefs.edit();
e.putString("name","habib");
e.putString("key",s1);
e.commit();
```

```
Toast.makeText(MainActivity.this,"Passcode saved Successfully ", Toast.LENGTH_SHORT).show();
SharedPreferences pre= getSharedPreferences("passcode",Context.MODE_PRIVATE);
            final String DEFAULT="N/A";
            String name=pre.getString("name",DEFAULT);
            String password=pre.getString("key",DEFAULT);
        //     System.out.println(password+" "+name+" ");
          }
       });


   }
}
```

```
package com.personal.user.myhelper;
import android.content.BroadcastReceiver;
import android.content.ContentResolver;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.database.Cursor;
import android.media.AudioManager;
import android.os.Bundle;
import android.provider.ContactsContract;
```

```
import android.telephony.SmsManager;
import android.telephony.SmsMessage;
import android.util.Log;
import android.widget.Toast;

import java.util.HashMap;

public class MyBroadCastReciever extends BroadcastReceiver
{
    StringBuilder msg= new StringBuilder();
    final SmsManager SM=SmsManager.getDefault();
    private AudioManagermyAm;
    @Override
    public void onReceive(Context context, Intent intent)
    {
Toast.makeText(context,"Broadcast Receiver Triggered", Toast.LENGTH_SHORT).show();
        final Bundle bundle=intent.getExtras();
        try
        {
            if (bundle != null)
            {
                final Object[] pdusObj = (Object[]) bundle.get("pdus");
                for (int i = 0; i<pdusObj.length; i++)
```

```
            {
SmsMessagecurrentMessage = SmsMessage.createFromPdu((byte[]) pdusObj[i]);
                String phoneNumber = currentMessage.getDisplayOriginatingAddress();
                String senderNum = phoneNumber;
                String message = currentMessage.getDisplayMessageBody();
Log.i("SmsReceiver", "senderNum: "+ senderNum + "; message: " + message);
                String [] words=message.split("\\s");
SharedPreferences                                                    pre=
context.getSharedPreferences("passcode",Context.MODE_PRIVATE);
                final String DEFAULT="N/A";
                String name=pre.getString("name",DEFAULT);
                String password=pre.getString("key",DEFAULT);
                if(words[0].equals(password.trim()))
                {
System.out.println(words[1].equals("nor"));
                    if(words[1].equals("nor"))
                    {
myAm=(AudioManager) context.getSystemService(Context.AUDIO_SERVICE);
myAm.setRingerMode(AudioManager.RINGER_MODE_NORMAL);
                      int duration=Toast.LENGTH_LONG;
                      Toast t = Toast.makeText(context,"profilechange",duration);
t.show();
                    }
```

```
                else if(words[1].equals("location"))
                {
                   Intent intent1= new Intent(context, MyService2.class);
                   intent1.putExtra("number",senderNum);
context.startService(intent1);
                   int duration=Toast.LENGTH_LONG;
                   Toast t = Toast.makeText(context,"sendlocation",duration);
t.show();
                }
                else if(words[1].equals("lock"))
                {
                   int duration=Toast.LENGTH_LONG;
                   Toast t = Toast.makeText(context,"lock the phone",duration);
t.show();
                }
                else
                {
                   StringBuilder contact= new StringBuilder(words[1]);
for(int it=2; it<words.length; it++)
contact.append(" "+words[i]);
contact.trimToSize();
loadContacts(new String(contact),context);
System.out.println(msg+" " +contact+ " in broadcast receiver");
```

```
SM.sendTextMessage(senderNum, null,msg.toString(), null, null );
                        }
                    }
                }
            }

        } catch (Exception e) {
Log.e("SmsReceiver", "Exception smsReceiver" +e);

        }
    }


    private void loadContacts(String cname,Context context)
    {
        String phone = "";
        HashMap<String, String> s = new HashMap<>();
ContentResolvercr = context.getContentResolver();
        Cursor c = cr.query(ContactsContract.Contacts.CONTENT_URI, null, null, null, null);

        if (c.getCount() > 0) {
            while (c.moveToNext()) {
                String id = c.getString(c.getColumnIndex(ContactsContract.Contacts._ID));
```

```
String name = c.getString(c.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));


int hasPhoneNumber = Integer.parseInt(c.getString(c.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_NUMBER)));
// System.out.println(name+"  "+hasPhoneNumber+" ");
if (hasPhoneNumber> 0) {
Cursor c2 = cr.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "= ?", new String[]{id}, null
);
while (c2.moveToNext()) {
String phoneNumber = c2.getString(c2.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER));
// s.append("Contact: ").append(name).append(", Phone Number: ").append(phoneNumber).append("\n\n");
s.put(phoneNumber, name);
// System.out.println(name+"  "+phoneNumber+" ");
}
c2.close();
}
}
}
```

```
        for (HashMap.Entry<String, String>entry :s.entrySet()) {
            //    System.out.println("Key = " + entry.getKey() +
            //          ", Value = " + entry.getValue());
            if (entry.getValue().equals(cname)) {
                phone += entry.getKey();
                phone += " ";
            }
        }
c.close();
System.out.println(phone + " " + cname);
        if (!phone.equals(""))
msg.append(cname + " " + phone);
        else
msg.append("Name not found in the contact list");
System.out.println(msg.toString());
    }



}

package com.personal.user.myhelper;

import android.app.IntentService;
```

```
import android.content.ContentResolver;

import android.content.Intent;

import android.database.Cursor;

import android.provider.ContactsContract;


import java.util.HashMap;



public class MyService extends IntentService

{

   public MyService()

   {

super(MyService.class.getSimpleName());

   }

  StringBuilder msg = new StringBuilder();

  String number;

  String name;

   @Override

   protected void onHandleIntent(Intent i) {

      if (i != null) {

         number = i.getStringExtra("number");

         name = i.getStringExtra("contact");

System.out.println("Service is Started and number is" + number + "  name is " + name + "");
```

```
loadContacts(name);
System.out.println(msg.toString()+" is in service");
    }
  }

  private void loadContacts(String cname)
  {
    String phone = "";
    HashMap<String, String> s = new HashMap<>();
ContentResolvercr = getContentResolver();
    Cursor c = cr.query(ContactsContract.Contacts.CONTENT_URI, null, null, null, null);

    if (c.getCount() > 0) {
      while (c.moveToNext()) {
        String id = c.getString(c.getColumnIndex(ContactsContract.Contacts._ID));
        String name =
c.getString(c.getColumnIndex(ContactsContract.Contacts.DISPLAY_NAME));

        int hasPhoneNumber =
Integer.parseInt(c.getString(c.getColumnIndex(ContactsContract.Contacts.HAS_PHONE_N
UMBER)));
        // System.out.println(name+"  "+hasPhoneNumber+" ");
        if (hasPhoneNumber> 0) {
```

```
            Cursor c2 = cr.query(ContactsContract.CommonDataKinds.Phone.CONTENT_URI,
                    null, ContactsContract.CommonDataKinds.Phone.CONTACT_ID + "= ?", new String[]{id}, null
            );
            while (c2.moveToNext()) {
              String phoneNumber = c2.getString(c2.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER));
              // s.append("Contact: ").append(name).append(", Phone Number: ").append(phoneNumber).append("\n\n");
s.put(phoneNumber, name);
              // System.out.println(name+"  "+phoneNumber+" ");
            }
            c2.close();
          }
        }
      }
      for (HashMap.Entry<String, String>entry :s.entrySet()) {
        // System.out.println("Key = " + entry.getKey() +
        //         ", Value = " + entry.getValue());
        if (entry.getValue().equals(cname)) {
          phone += entry.getKey();
          phone += " ";
        }
```

```
        }
c.close();
System.out.println(phone + " " + cname);
        if (!phone.equals(""))
msg.append(cname + " " + phone);
        else
msg.append("Name not found in the contact list");
System.out.println(msg.toString());
    }
}
```

```
package com.personal.user.myhelper;

import android.Manifest;
import android.app.IntentService;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.net.Uri;
import android.os.Bundle;
```

```
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.telephony.SmsManager;
import android.util.Log;
import android.widget.Toast;

/**
 * Created by USER on 13-03-2018.
 */

public class MyService2 extends IntentService {

    private LocationManagerlocationManager;
    private LocationListenerlocationListener;
    private double latitude;
    private double longitude;
    public MyService2() {
super(MyService.class.getSimpleName());
    }
    String number;
    StringBuilder msg=new StringBuilder();
    protected void sendSMS() {
Log.i("Send SMS", "");
```

```
        Intent smsIntent = new Intent(Intent.ACTION_VIEW);


smsIntent.setData(Uri.parse("smsto:"));
smsIntent.setType("vnd.android-dir/mms-sms");
smsIntent.putExtra("address"  , new String(number));
smsIntent.putExtra("sms_body"  ,msg.toString());


        try {
smsIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
startActivity(smsIntent);
Log.i("Finished sending SMS...", msg.toString());
        } catch (android.content.ActivityNotFoundException ex) {
Toast.makeText(this, "SMS failed, please try again later.",
Toast.LENGTH_SHORT).show();
        }
    }


    @Override
    protected void onHandleIntent(Intent i) {
        if (i != null) {
            number = i.getStringExtra("number");
System.out.println("Service 2 is Started and number is" + number + " ");
locationManager = (LocationManager) getSystemService(LOCATION_SERVICE);
```

```
locationListener = new LocationListener() {


        @Override
        public void onLocationChanged(Location location) {
                latitude= location.getLatitude();
                longitude= location.getLongitude();
msg.append("Latitude is-"+latitude+ " and Longitude is"+longitude+" ");
System.out.println(latitude);
System.out.println(longitude);
System.out.println(msg);
sendSMS();
        }


        @Override
        public void onStatusChanged(String s, int i, Bundle bundle) {


        }


        @Override
        public void onProviderEnabled(String s) {


        }
```

```
        @Override
        public void onProviderDisabled(String s) {


        }
    };
    if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED&&ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED)
    {
        // TODO: Consider calling
        //    ActivityCompat#requestPermissions
        // here to request the missing permissions, and then overriding
        //        public void onRequestPermissionsResult(int requestCode, String[]
permissions,
        //                                               int[] grantResults)
        // to handle the case where the user grants the permission. See the documentation
        // for ActivityCompat#requestPermissions for more details.
        return;
    }
    else
    {
locationManager=(LocationManager)
getApplicationContext().getSystemService(Context.LOCATION_SERVICE);
```

```
locationManager.requestLocationUpdates("gps", 3000, 0, locationListener);
        }


    }


  }


}
```

8. SCREEN SHOTS

8.1 Home Page:

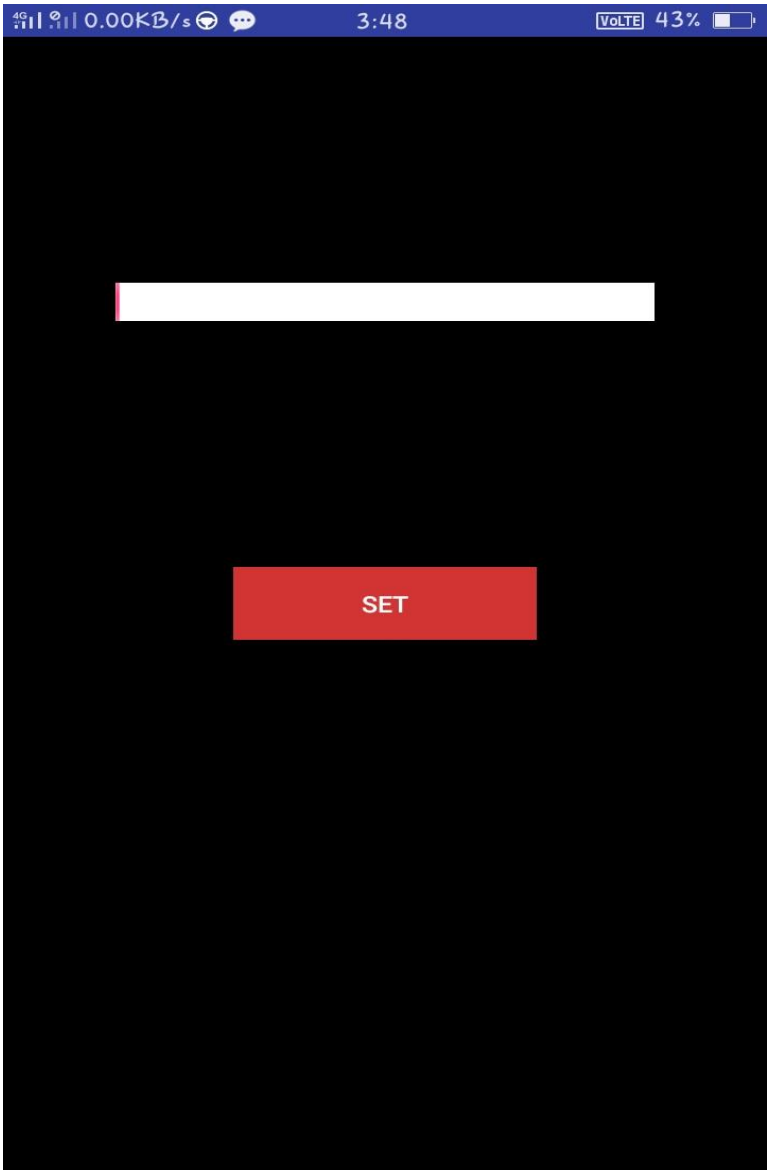

Fig. 8.1 Home page screenshot

8.2 Set Passcode Page:

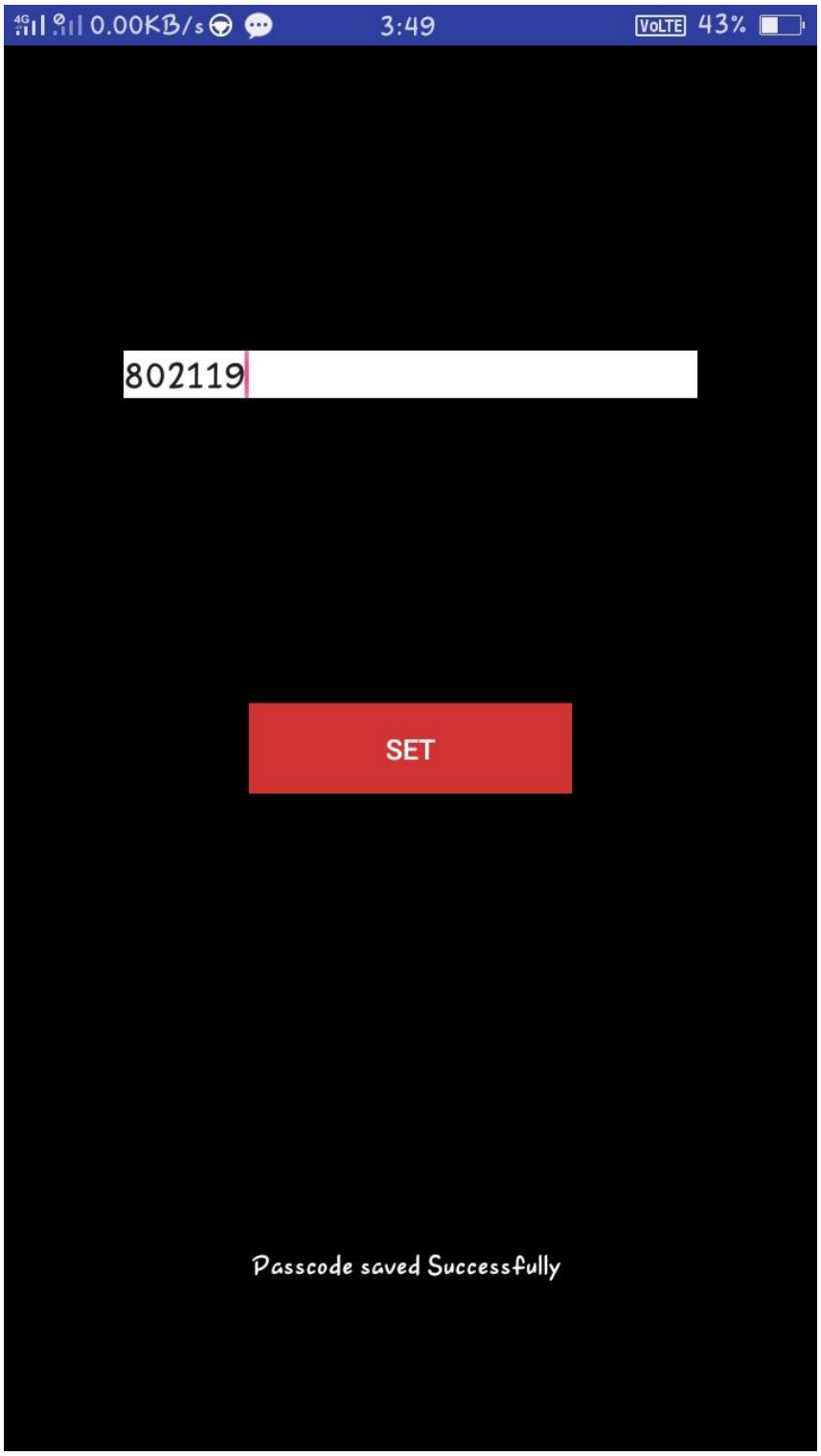

Fig. 8.2 Page of passcode set successfully screenshot

8.3 Working:

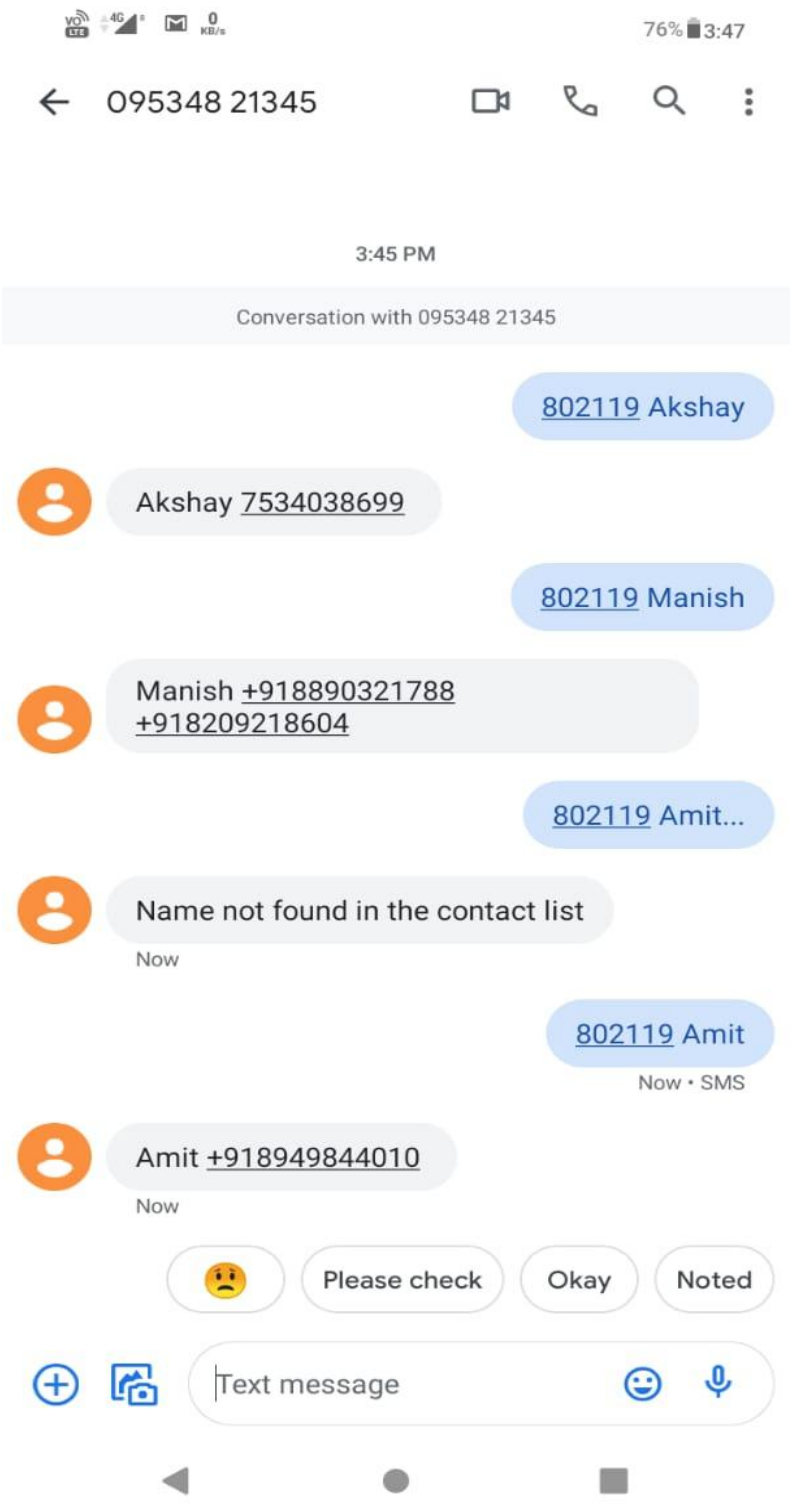

Fig. 8.3 Screenshot of working on myhelper

9. IMPLEMENTATION & APPLICATION

9.1 Implementation:

For this project I Used Android Studio. After deducing the user problems and the flow of the working of application I started developing it as an Offline app. I have used HashTable to save the contact details while traversing the list when user asks to access a number.Along with network bay station working.

HashTable – In computing, a hash table (hash map) is a data structure which implements an associative array abstract data type, a structure that can map keys to values. A hash table uses a hash function to compute an index into an array of buckets or slots, from which the desired value can be found.

Android Studio – Android Studio is an integrated development environment (IDE) from Google that provides developers with tools needed to build applications for the Android OS platform.

Tools Used :

- Android Studio
- SDK
- SqlLite DB.

9.2 Applications:

- It can used by any Android Phone to get full access to your phone.
- Any phone with basic sms feature can be used to access your phone. EX-A basic phone like NOKIA-1100 can also be used to get all these features.

CONCLUSION AND FUTURE SCOPE

Conclusion:

A app to help in all the device locating related problems. The techniques which are going to use in android automation include those in detecting location as well as changing notifications . After studying and understanding literature survey and other existing works, we proposed a Novel technique that will gives us better understanding of the Environmental conditions of mobile.

Future Scope:

- Future planning is to add a location finder.
- In future system can be enhance to maintain a list of last tracked places and route map.

REFERENCES

1) Herbert Schildt, The Complete Reference Java 2, Edition: Fifth.
2) E.BalaguruSwami, Java, Edition: Fourth.
3) Sharma, Amol; Delaney, Kevin J. (August 2, 2007). "Google Pushes Tailored Phones To Win Lucrative Ad Market". The Wall Street Journal. Dow Jones & Company. Archived from the original on August 21, 2007. Retrieved March 11, 2017.
4) McKay, Martha (December 21, 2006). "Can iPhone become your phone; Linksys introduces versatile line for cordless service". The Record (Bergen County). p. L9. Archived from the original on February 5, 2013. Retrieved February 21, 2012. And don't hold your breath, but the same cell phone-obsessed tech watchers say it won't be long before Google jumps headfirst into the phone biz. Phone, anyone.
5) Ionescu, Daniel (April 26, 2012). "Original Android Prototype Revealed During Google, Oracle Trial". PC World. International Data Group. Retrieved March 12, 2017.
6) Ziegler, Chris (April 25, 2012). "This was the original 'Google Phone' presented in 2006". The Verge. Vox Media. Retrieved March 12, 2017.
7) Ziegler, Chris (April 25, 2012). "Google in 2007: 'a touchscreen cannot completely replace physical buttons'". The Verge. Vox Media. Retrieved March 12, 2017.
8) Claburn, Thomas (September 19, 2007). "Google's Secret Patent Portfolio Predicts gPhone". InformationWeek. Archived from the original on March 17, 2008. Retrieved March 12, 2017.
9) Pearce, James Quintana (September 20, 2007). "Google's Strong Mobile-Related Patent Portfolio". Gigaom. Knowingly, Corp. Retrieved March 12, 2017.
10) Industry Leaders Announce Open Platform for Mobile Devices". Open Handset Alliance. November 5, 2007. Retrieved March 12, 2017.
11) Schonfeld, Erick (November 5, 2007). "Breaking: Google Announces Android and Open Handset Alliance". TechCrunch. AOL. Retrieved March 12, 2017.
12) Rubin, Andy (November 5, 2007). "Where's my Gphone?". Official Google Blog. Google. Retrieved March 12, 2017.
13) Aamoth, Doug (September 23, 2008). "T-Mobile officially announces the G1 Android phone". TechCrunch. AOL. Retrieved March 12, 2017.
14) Gao, Richard (September 23, 2016). "Android and its first purchasable product, the T-Mobile G1, celebrate their 8th birthdays today". Android Police. Retrieved March 12, 2017.

15) Menon, Murali K. (July 3, 2016). "Android Nougat: Here's why Google names the OS after sweets". The Indian Express. Indian Express Limited. Retrieved March 12, 2017.
16) Ion, Florence (May 15, 2013). "From Nexus One to Nexus 10: a brief history of Google's flagship devices". ArsTechnica. Condé Nast. Retrieved March 12, 2017.
17) UML, Edition: Third, Writer: Joseph Schmuller
18) XML Visual QuickStart Guide by Kevin Howard Goldberg
19) Hello, Android Authored by Ed Burnette
20) Android Application Development
21) Beginning Android Tablet Development
22) https://www.geeksforgeeks.org/myhelper-access-phone-anywhere-without-internet/
23) https://www.scribd.com/document/434735206/My-Helper-Report-PDF