**1. SUMMARY**

Nowadays we are dependent on our mobile phones, if we forget the phone at home it seems we have lost a limb. That time we think that it would be good to access our mobile remotely, like the web browser. The application like Remote Access, instead of accessing the computer remotely, we will access the mobile phones. This application helps us to retrieve all the data like **missed calls, contacts and message.** And we can turn on/off **Wi-Fi, Mobile data**. And we can turn on **Airplane mode**. Through this application SMS could be sent with the application’s IP addresses and then it would be the mobile phone establishing the TCP connection

We access the data remotely with the help of Android Mobile application into the web application. The basic idea is to design and develop a Backup Application which allows the user

to browse through Mobile data like SMS and contacts and allows them to upload his data on a remote server. Performance upgrades for mobile phones due to jumping development of technology, such terminals that can store information that an individual saved called smartphone were released to market.

Remotely controlling the mobile phone, the lost phones information can be handled remotely. We can access the information and also we can track the mobile phone. This is useful for protecting the mobile phone and the information of the phone. The remote terminal allows user to access the information and to send the remote SMS through this application and also to receive the SMS from application from his target mobile. The interactive system facilitates user to have a backup of his call logs, SMS, and contacts.

**2. INTRODUCTION**

**2.1 Project Overview:-**

If we forget the phone at home it seems we have lost a limb. That time we think that it would be good to access our mobile remotely, like the web browser. The application like Remote Access, instead of accessing the computer remotely, we will access the mobile phones. This application helps us to retrieves all the data like missed calls, contacts and message**.** And we can turn on/off Wi-Fi, Mobile data. And we can turn on Airplane mode. Through this application SMS could be sends with the application IP addresses and then it would be the mobile phone establishing the TCP connection

Remotely controlling the mobile phone, the lost phones information can be handled remotely. We can access the information and also we can track the mobile phone. This is useful for protecting the mobile phone and the information inside the phone. The remote terminal allows user to access the information and to send the remote SMS through this application and also to receive the SMS from application from his target mobile. The interactive system facilitates user to have a backup of his call logs, SMS, and contacts.

The project has been divided into following modules.

1. Get Contacts
2. Get Missed Calls
3. Get Message
4. Get IMEI number
5. Remote Alarm
6. Control Wi-Fi
7. Control Mobile Data
8. Control Airplane mode
9. Reboot System

**2.1.1 The Organization Profile:-**

At Webheay Technologies, we’re continuously building on our foundation of building corporate application. Our corporate application having the power to keep track your daily business activity to grow your business with a specific direction.

We offer our customers the best services & solutions, this is our main services list. Our prime emphasis is to make this world together a better and more connected place with the help of technologies. Bring people closer to technologies and providing solution to all of their needs online.

Keeping the Start Up India and Skilled India in mind, we offer our employees ample of resource needed to skill themselves and as a result of that our employees are skilled and central to our success. We have a work environment that is creative, intellectually simulating, energetic, team-focused and fun.

Our technologies helps business to grow at Desktop, Tablet Phone Level . We help our clients to carry out their Business over Web , IOS , and Android level adding Google Street view, Google services ,SEO and YouTube Promotion services so that they can meet their consumer at every remote location and stay connected with them.

**2.1.2 Study on Existing System:-**

The existing Application which allows the user to browse through Mobile data like SMS and contacts and allows them to upload his data on a remote server. Performance upgrades for mobile phones due to jumping development of technology, such terminals that can store information that an individual saved called smartphone were released to market.

Remotely controlling the mobile phone, the lost phones information can be handled remotely. We can access the information and also we can track the mobile phone. This is useful for protecting the mobile phone and the information inside the phone. The remote terminal allows user to access the information and to send the remote SMS through web application and also to receive the SMS on web application from his target mobile. The interactive system facilitates user to have a backup of his call logs, SMS, and contacts on server.

**2.1.3 Proposed System:-**

The proposed system helps us to access the mobile remotely. Nowadays we are dependent on our mobile phones, if we forget the phone at home it seems we have lost a limb. That time we think that it would be good to access our mobile remotely, like the web browser. The application like Remote Access, instead of accessing the computer remotely, we will access the mobile phones. This application helps us to retrieves all the data like **missed calls, contacts and message.** And we can turn on/off **Wi-Fi, Mobile data**. And we can turn on **Airplane mode**. Through this application SMS could be sends with the application IP addresses and then it would be the mobile phone establishing the TCP connection

To access the data remotely with the help of Android Mobile application into the web application. The basic idea is to design and develop a Backup Application which allows the user

to browse Mobile data like SMS and contacts . Performance upgrades for mobile phones due to jumping development of technology, such terminals that can store information that an individual saved called smartphone were released to market.

**2.2 Statement of the Problem:-**

A literature search has been carried out, to collect the information and data for various items listed. A literature review is both a summary and explanation of the complete and current state of knowledge on a limited topic as found in books and journal articles. The required information 9and data has been collected from the standard books, monographs, internet sites and other data base for sourcing material safety data sheets, information on and other data base available on the internet and other and data as required for the compilation of the desired data, listed under has been collected.

To create a project which helps user to find the best automated system used by their Customer. Remote Access with Anti-Theft is helps us to access the mobile remotely. Here we are getting information by sending message.

After the system is implemented, a review should be conducted to determine whether the system is meeting expectations and where improvements are needed. System quality, user confidence and operating systems statistics are accessed through such event logging, impact evaluation and attitude surveys. The review not only accesses how well the proposed system is designed and implement, but also is a valuable source of information that can be applied to a critical evaluation if the system.

The reviews are conducted by the operating personals as well as the software developers in order to determine how well the system is working, how it has been accepted, and whether adjustments are needed. The review of the system is highly essential to determine the future enhancements required by the system. The system can be considered successful only if information system has met its objectives. The review analysis the opinion of the employees and identifies the attitudes towards the new computerize system. Only when we know about the merits and demerits of the implemented system, we can determine what all the additional features it requires are. The following are the issues to be considered in the evaluation of a system.

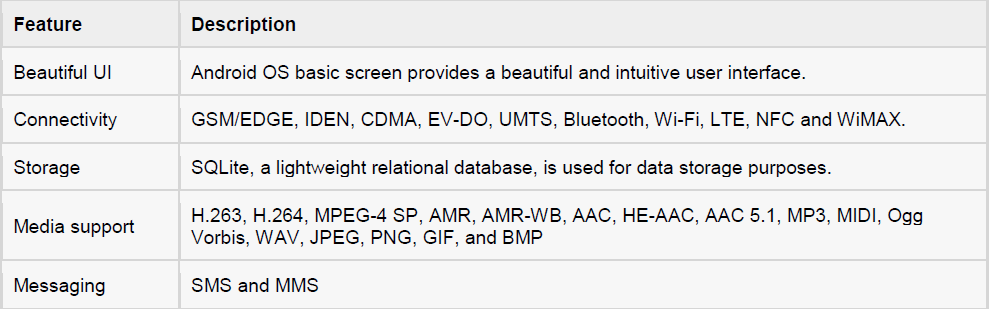
**Technologies Used**

**1. Android**

Android is an open source and Linux-based **Operating System** for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies. Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android. The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008. On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 **Jelly Bean**. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance. The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

**2. Features of Android**

Android is a powerful operating system competing with Apple 4GS and supports great features. Few of them are listed below:



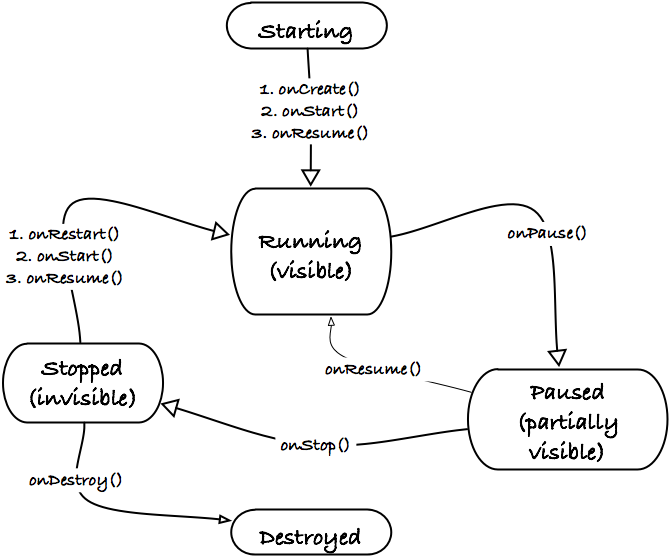
**3.Building block of Android**

An android component is simply a piece of code that has a well defined life cycle e.g. Activity, Receiver, Service etc. The core building blocks or fundamental components of android are activities, views, intents, services, content providers, fragments and AndroidManifest.xml.

****

**3.1.Activity**

An activity is a class that represents a single screen. It is like a Frame in AWT.



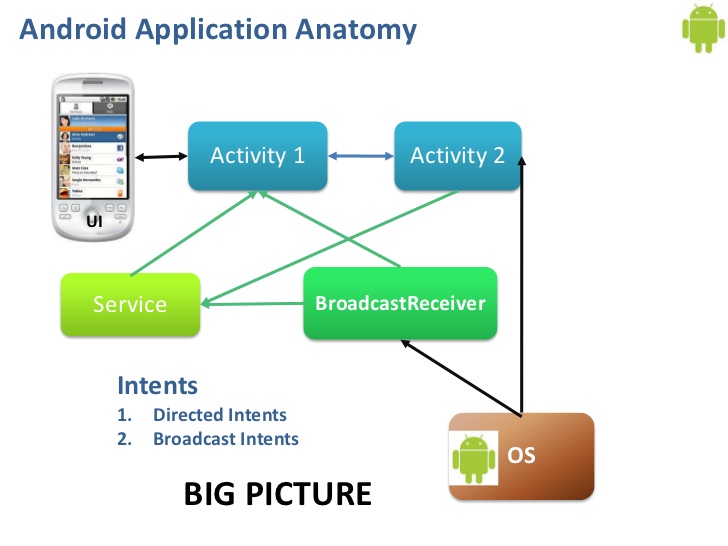
**3.2.Intent**

Intent is used to invoke components. It is mainly used to:

* Start the service
* Launch an activity
* Display a web page
* Display a list of contacts
* Broadcast a message
* Dial a phone call etc.

**3.3.Broadcast Receivers**

Broadcast Receivers simply respond to broadcast messages from other applications or from the system itself. These messages are sometime called events or intents. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action.



There are following two important steps to make Broadcast Receiver works for the system broadcasted intents −

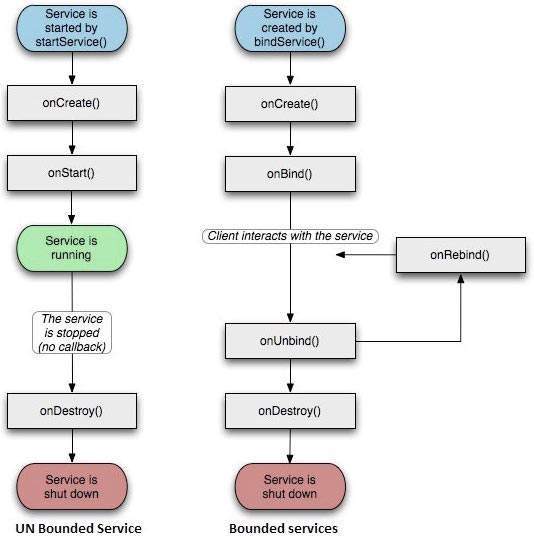
* Creating the Broadcast Receiver.
* Registering Broadcast Receiver

There is one additional steps in case you are going to implement your custom intents

then you will have to create and broadcast those intents.

**3.4. Service**

An important application component in the Android platform is a service. A Service is an application component that can perform long-running operations in the background and does not provide a user interface. Another application component can start a service and it will continue to run in the background even if the user switches to another application. Additionally, a component can bind to a service to interact with it and even perform inter process communication (IPC). For example, a service might handle network transactions, play music, perform file I/O, or interact with a content provider, all from the background. A service can essentially take two forms

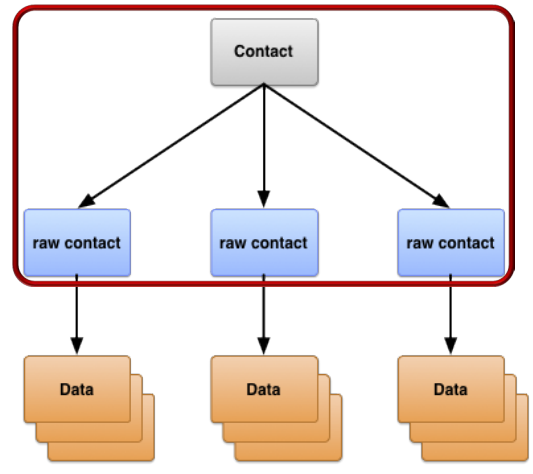


**Started:** A service is "started" when an application component (such as an activity) starts it by calling start Service(). Once started, a service can run in the background indefinitely, even if the component that started it is destroyed. Usually, a started service performs a single operation and does not return a result to the caller. For example, it might download or upload a file over the network. When the operation is done, the service should stop itself.

**Bound:** A service is "bound" when an application component binds to it by calling bind Service(). A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with inter process communication (IPC). A bound service runs only as long as another application component is bound to it. Multiple components can bind to the service at once, but when all of them unbind, the service is destroyed.

**3.5 content provider**

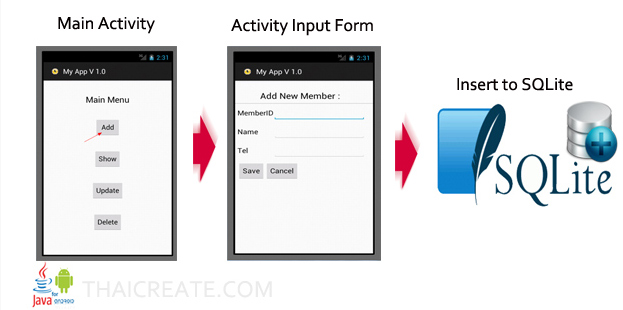
Contacts Provider is a powerful & flexible component that manages the device's central repository of data about people. A raw contact represents a person's data coming from a single account type & account name .The Contacts Provider allows multiple raw contacts for the same person



**4. About SQLite**

SQLite is an in-process library that implements a self-contained, server less, zero-configuration, transactional SQL database engine. It is the one database, which is zero-configured, that means like other database you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. The SQLite accesses its storage files directly.



* SQLite does not require a separate server process or system to operate (server less).
* SQLite comes with zero-configuration, which means no setup or administration needed.
* A complete SQLite database is stored in a single cross-platform disk file.
* SQLite is very small and light weight, less than 400KiB fully configured or less than 250KiB with optional features omitted.
* SQLite is self-contained, which means no external dependencies.
* SQLite transactions are fully ACID-compliant, allowing safe access from multiple processes or threads.
* SQLite supports most of the query language features found in the SQL92 (SQL2) standard.
* SQLite is written in ANSI-C and provides simple and easy-to-use API.
* SQLite is available on UNIX (Linux, Mac OS-X, Android, iOS) and Windows (Win32, WinCE, WinRT).

5. JAVA

**J**ava was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at SUN Microsystems Incorporation in the year 1991.It took 18 months to develop the 1st working version. This language was initially called “OAK”, but was renamed “JAVA” in 1995, many more contributed to the design and evolution of the language.

**JAVA OVERVIEW**

Java is a powerful but lean object-oriented programming language. It has generated a lot of excitement because it makes it possible to program for

Internet by creating Applets. Programs that can be embedded in web page. The context of an applet can be an animation with sound, an interactive game or a ticker tape. With constantly updated stock prices. Applets can be just little decorations to liven up web page, or they can be serious applications like Word processor or Spreadsheet.

But Java is more than a programming language for writing Applets. It is being used more and believe it will become standard language for both general purpose and Internet programming.

There are many buzzwords associated with Java, but because of its spectacular growth in popularity, a new buzzword has appeared ubiquitous. Indeed, all indications are that it will soon be everywhere.

Java builds on the strength of C++. It has taken the best features of C++ and discarded the more problematic and error prone parts. To this lean core, it has added garbage collection (automatic memory management), multithreading (the capacity for one program to do more than one thing at a time), security capabilities. This result is that Java is simple, elegant, and powerful and easy-to-use.

Java is actually a platform consisting of 3 components:

1. Java Programming Language.
2. Java Library of Classes and Interfaces.
3. Java Virtual Machine

**Advantages of JAVA**

One of the biggest advantages Java offers is that it is portable. An application written in Java will run on all the major platforms. Any computer with a Java-based browser can run the applications or Applets written in the Java-Programming-Language. A programmer no longer has to write one program to run on a Macintosh, another program to run on a Windows-machine still another to run on a UNIX-machine and so on. In other words, with Java developers write their programs only once.

The Virtual Machine is what gives Java is cross platform capabilities. Rather being compiled into machine language, which is different for each OS’s and computer architecture, Java code is compiled into Byte codes.

With other languages, the program code is compiled into a language that the computer can understand. The problem is that other computers with different machine instruction set cannot understand that language. Java code on the other hand is compiled into Byte-Code rather than a machine language. These byte codes go to the JVM, which executes them directly or translates them into the language that is understood by the machine running it.

**2.3 Waterfall Model**

The proposed system uses waterfall model, where the stages of development process are shown as cascading from one to another. The waterfall model presents a very high level view of what goes on during development and suggests the sequence of events to the developer. It requires that a phase is complete before the next phase is started. Because of the explicit recognition of phases and sequencing, it helps in contract realization with reference to delivery and payment schedules. The pictorial representation of waterfall model is as shown in Figure 2.1

Requirements Analysis and Definition

System and Software Design

Implementation and Unit Testing

Integration and System testing

Operation and Maintenance

**Figure 2.1: Waterfall Model**

Various steps in waterfall model are explained below:

* Requirements Analysis and Definition
* System and Software Design
* Implementation and Unit Testing
* Integration and System Testing
* Operation and Maintenance

Though waterfall model is widely used, it has certain limitation since the waterfall model is linear, rigid and monolithic. To rectify the errors, all the stages should be repeated once again. It will increase the time and cost. Thus the software process is not a simple linear model but involves a sequence of iteration of the various development activities.

**3. PROBLEM ANALYSIS**

Analysis means to break something up into parts, pieces, reasons, or steps and look at how those pieces are related to each other. Analysis usually goes together with synthesis because first we break down a concept/idea into its important parts of analysis, so that we can draw useful conclusions or make decisions about the topic or problem.

Collecting the information related to social networking and sorting it out based on the categories such as messages, emails, friends, groups, discussion forums etc. and building each of the components and merging them into a single web application.

Here, the analysis is carried out by dividing all the modules as an individual module and analyze it. After analyzing all the modules and conclusions we integrate all modules and analyze it to check whether it’s working properly as the user expected or not.

**Feasibility Analysis**

Feasibility Analysis is the process by which feasibility is measured. It is an ongoing process done frequently during systems development projects in order to achieve a commitment from the user and continually assess the current status of the project. A creeping commitment is one that continues over time to reinforce the user’s commitment and ownership of the information system being developed. Knowing a project’s current status at strategic points in time gives us and the user the opportunity to continue the project as planned, make changes to the project or cancel the project.

**Requirement Analysis**

The requirements determination activity is the most difficult part of information systems analysis. Requirements determination addresses the gathering and documenting of the true and real requirements for the information system being developed. Many textbooks refer to this activity as the portion of information systems development. In other words, the systems analyst is primarily thinking and trying to answer the question, what must the system do? during the requirements determination activity. Once information systems development progresses to the design activities, the systems analyst and the programmers focus their attention primarily on the question, how does the system do what it is supposed to do?

Why is requirements determination difficult? There are several reasons why this is true. Most are attributed to the fact that this is a highly imaginary and creative activity for all of the members of the development team, including the users. Requirements determination represents perhaps one of the last frontiers still awaiting significant automated and intelligent support.

**Content Analysis**

In this analysis, we identified the full spectrum of content such as graphics and images. Content analysis or textual analysis is a methodology for studying the content of communication. It can allow for both quantitative and qualitative operations which provide historical/cultural insights over time through analysis of texts. Focus is on the individual web page/site as the unit of analysis. Systematic set of criteria based on items or features of web pages developed into a coding scheme that is applied to sites to provide quantitative measures of content, functionality, usability and design features. Content analysis is a summarizing, quantitative analysis of messages that relies on the scientific method (including attention to objectivity, a priori design, reliability, validity and hypothesis testing) and is not limited as to the types of variables that may be measured or the context in which the messages are created or presented."

**Interaction Analysis**

In this analysis, we analyze how the user interacts with our pages. We designed the frame layout of each and other pages and selected the meaningful text and animated images for the appropriate topic. The interaction analysis allows user to make interaction with the web pages. To interact between the user and web pages we made the web pages more attractive and navigation based. Navigation based means the user can navigate easily from one page to another. We created menus so that user can easily see the contents of the menu by clicking on it.

**Functional Analysis**

In this analysis, all operations and functions are described in detail. The page scenario has created part of interaction analysis define the operation that will be applied to web application content and imply after having many operations such as input, validation, subscription from validation, checking user authentication and performing web page navigation. For user authentication, the user has to first register to the site. After registering, one link will be sent to user’s email id. By clicking on that link the user account will be activated. For navigation purpose, we created the menus so that user can easily navigate from one page to another.

**Configuration Analysis**

The environment and infrastructure in which the web application resides are described in details. The web application can be on the internet or intranet. In addition, infrastructure (the component infrastructure and degree to which a database will be used to generate content) for web applications should be identified at this stage. Configuration means arrangement, i.e. we arranged the icons, widgets as well as comment boxes at the place from where the user access the information easily and quickly.

**4. FEASIBILITY STUDY**

All projects are feasible, given unlimited resources and infinite time. But the development of software is plagued by the scarcity of resources and difficult delivery rates. It is both necessary and prudent to evaluate the feasibility of a project at the earliest possible time.

Three key considerations are involved in the feasibility analysis.

**4.1 Technical Feasibility:-**

Technical feasibility speaks about the existing hardware and the software that we are using and the deviations that we have to make from the existing one, as we are developing the application using java there is no change in the hardware that we are using. So we a say that this application is technically feasible as there is no change in the configuration more over it is cost effective.

**4.2 Economic Feasibility:-**

Economic feasibility talks about the benefits that which we get from this project. Here with the introduction of this online process we are not only reducing the time take for the registration of the entrants we even reduce the burden on the administrator. As this project is not only reducing the time but also the work burden of the user we say that this product is economically feasible.

**4.3 Operational Feasibility:-**

As this project is a user friendly version there is not much training required for the people to use. This product is not only making the task of the administrator easy but it is reducing the time that is taken otherwise. So we say that this product is operationally feasible.

**5. SOFTWARE ENGINEERING PARADIGM APPLIED**

Software Engineering is a planned and systematic approach to the development of software. It is a discipline that consists of methods, tools and techniques used for developing and maintaining software. To solve actual problems in an industry setting, a software engineer or team of engineers must incorporate a development strategy that encompasses the process, methods and tool layers and generic phases. This strategy is often referred to as a process model or Software engineering paradigm.

For developing a software product, user requirements are identified and the design is made based on these requirements. The design is then translated into a machine executable language that can be interpreted by a computer. Finally, the software product is tested and delivered to the customer. The Spiral model incorporates the best characteristics of both the waterfall and prototyping model.

 In addition, the Spiral model also contains a new component called Risk Analysis, which is not there in waterfall and prototype model. In the Spiral model, the basic structure of the software product is developed first. After the basic structure is developed, new features such as user interface and data administration are added to the existing software product. This functionality of the Spiral model is similar to a spiral where the circles of the spiral increase in diameter. Each circle represents a more complete version of the software product.

**6. SOFTWARE REQUIREMENTS SPECIFICATION**

A Software Requirements Specification (SRS) is a requirements specification for a software system which is a complete description of the behavior of a system to be developed and may include a set of use cases that describe interactions of the user with the software. In addition to a description of the software functions, the SRS also contains non-functional requirements. Non-functional requirements impose constraints on the design or implementation such as performance engineering requirements, quality standards or design constraints.

Software requirement is a sub-field of software engineering that deals with the elicitation, analysis, specification and validation of requirements for software. The aim of the SRS document is to list out the user requirements in an organized manner. It defines all the constraints and requirements needed to understand this specification and documentation. It also gives the design plan. The user should be able to understand the proposed system after going through the SRS document and should be in a position to incorporate some changes if required.

**User Requirements**

User requirements are statements, in a natural language and diagrams, of what service the system is expected to provide and the constraints under which it must operate. User requirements are often classified as viz.

* **Functional Requirements:** There are the statements of services the system should provide, how the system should react to particular inputs and how the system should behave in particular situations. In some cases, the functional requirements may also explicitly state what the system should not do.
* **Non- Functional Requirements:** These are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, standards, etc.

**1. Functional Requirement**

The following sections describe the functional requirements of the system for those requirements.

**Output Design:** Outputs from computer systems are required primarily communicating the results of processing to users. They are also used to provide a permanent copy of the results for later consultation.

**Input Design:** Input design is a part of overall system design. The main objective during the input design is as given below:

* To produce a cost-effective method of input.
* To achieve a highest possible level of accuracy.
* To ensure that the input is acceptable and understood by the user

**Error Avoidance:** At this stage, care is to be taken to ensure that input data remains accurate from the stage which it is recorded up to the stage. In which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

**Error Detection:** Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors are always likely to occur, these types of errors can be discovered by using validations to check the input data.

**Data Validation:** Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the use to commit errors. The system will not accept invalid data. Validations have been included where necessary.

**2. Non-Functional Requirements**

The following sections describe the non-functional requirements of the system for those requirements.

**External Requirement:** External requirement include the entire requirement that may arise from the external factors. These include interoperability requirement that define the compatibility of the proposed system with other systems, legislative requirement to ensure that the product compiles with the state legislation and ethical requirement to ensure social acceptability.

The proposed system is fully interoperable with existing because it is being developed with the consideration crashes, performance attributes are measured.

**Performance Requirements:** Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It tests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system.

The requirement specification for any system can be broadly stated as given below:

* The system should be able to interface with the existing system.
* The system should be accurate.
* The system should be better than the existing system.

**Design Constraints:** Design constraints are the conditions that have to be followed while designing a system. The design constraints are described hereunder.

* This design is valid with the current set of requirements only. Any changes in the requirement would call for changes in design.
* The system will not have any provision to calculate any interest on advance, security etc.
* The layout of the screens are given in the annexure is for giving a preview and uses our current terminology. There could be some minor differences between this and final screens.
* The system will have to satisfy the requirements then the design phase is held.

**6.1 Software and Hardware Requirement:-**

**1.For Computer System**

**Hardware Requirements**

* Processor :Pentium4 and above
* Memory :2GB and above
* Storage :120GB and above

**Software Requirements**

* Programming Language : ANDROID,JAVA,
* Web Technologies : XML
* Operating system : Windows, Linux
* Database : SQLite

**2.For Mobile**

**Hardware Requirements**

* Processor :Quad-core one and above
* Memory :512MB and above
* Storage :1GB and above

**Software Requirements**

* Operating system : Android Versions
* Database : SQLite

**7. SYSTEM DESIGN**

The most creative and challenging phase of the life cycle is system design. The term design describes a final system and the process by which it is developed. It refers to the technical specifications that will be applied in implementations of the candidate system. The design may be defined as “the process of applying various techniques and principles for the purpose of defining a device, a process or a system with sufficient details to permit its physical realization”*.*

The designer’s goal is how the output is to be produced and in what format. Samples of the output and input are also presented. Second input data and database files have to be designed to meet the requirements of the proposed output. The processing phases are handled through the program Construction and Testing. Finally, details related to justification of the system and an estimate of the impact of the candidate system on the user and the organization are documented and evaluated by management as a step toward implementation.

The importance of software design can be stated in a single word “Quality”. Design provides us with representations of software that scan be assessed for quality. Design is the only way where we can accurately translate a customer’s requirements into a complete software product or system. Without design we risk building an unstable system that might fail if small changes are made. It may as well be difficult to test, or could be one who’s quality can’t be tested. So it is an essential phase in the development of a software product.

The design of a system is perhaps the most critical factors affecting the quality of the software, and has a major impact on the later phases, particularly the testing and the maintenance. The outer of this phase of the design of the system is essentially a blue print or a plan for the system. System design involves the logical interrelationships of the system components. The design of the system reflects the strength of the software. Better the design, better the quality, efficiency and reliability of the software.

It is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. On could see it as the application of systems to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. System design is therefore the process of defining and developing systems to satisfy specified requirements of the user. Until the 1990s the system design had a crucial and respected role in data processing industry. In the 1990s the standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of then system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is authenticated verified, how it is processed and how it is displayed as output. The physical portion of the systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User interface design is concerned with how users add information to the system and with how the systems presents information back to them. Data design is concerned with how the data is represented and stored within the system. Finally, Process design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer’s physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots etc. It involves detail design of user and a product database structure processing and control processor.

**7.1 Design Notations:-**

The Basic Notation used to create a DFD’s are as follows:

**Dataflow:** Data move in a specific direction from an origin to a

Destination.

**Process:** People, procedures, or devices that use or produce

(Transform) Data. The physical component is not identified.

**Source:** External sources or destination of data, which may be

People, programs, organizations or other entities.

**Data Store:** Here data are stored or referenced by a process in the

System

**7.1.1 Data Flow Diagram (DFD):-**

A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also know as a data flow graph or a bubble chart.

**Types of data flow diagrams**

DFDs are two types

**1. Physical DFD**

Structured analysis states that the current system should be first understand correctly. The physical DFD is the model of the current system and is used to ensure that the current system has been clearly understood. Physical DFDs shows actual devices, departments, people etc., involved in the current system

**2. Logical DFD**

Logical DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during design activity this is taken as the basis for drawing the system’s structure charts.

The top-level diagram is often called a “*context diagram”*. It contains a single process, but it plays a very important role in studying the current system. The context diagram defines the system that will be studied in the sense that it determines the boundaries. Anything that is not inside the process identified in the context diagram will not be part of the system study. It represents the entire software element as a single bubble with input and output data indicated by incoming and outgoing arrows respectively.

**Table: DFD Symbols Used**

|  |  |  |
| --- | --- | --- |
| **Name Of Symbol** | **Symbol** | **Description** |
| **Arrow** |  | A data flow is a root, which enables packets of data to travel from one point to another. Arrow identifies the data flow ---- data in motion. It is a pipeline through which information flows. |
| **Process** |  | A process represents transformations where incoming data flows are changed into outgoing data flows. A circle or a “bubble” represents a process that transforms incoming data flows into outgoing data flow. |
| **Data source** |  | A data source represents a repository of data that is to be stored for use by one or more processes and may be simple as a buffer or queue or sophisticated as a relational database. They should have clear names. |
| **A source or sink** |  | A source or sink is a person or part of an organization, which enters or receives information from the system but is considered to be outside the context of data flow model. |

* A repository of data that is to be stored for use by one or more processed may be simple or sophisticated.
* A DFD describes what data flows (logically) rather than how they are processed so that it doesn’t depend on the hardware, software, data structure or file organization.
* The DFD methodology is quite effective, especially when the required design is unclear and the user and the analyst need a national language for communication.
* The DFD is easy to understand after a brief orientation.

**Level-0 DFD**

USER

Home page

**Figure: Data Flow diagram**

**Level-1 DFD**

DATA BASE

**7.1.2 Use Case Diagram**

Use case diagrams are used to identify the required usage of a system in terms of actors and use-cases. They are used to model a system’s desired functions and its environment. The basic use case symbols and notations are:

: Represents the system’s functions.

: Represents the actors who are the users of the system.

**<<Uses>>** : Relationship indicates that one use case is needed by another in order to perform a task.

**<<Includes>>** : Relationship indicates including a particular option under a certain use case.

**<<Extends>>** : Relationship indicates alternative options under a certain use case.

**Change Password and Phone number**

Database

User

**Figure: Use case diagram for Registration and Login**

**Customer Enquiry**

User

Server

**Figure: Use case diagram for customer enquiry**

**7.1.3 Sequence Diagram:-**

Sequence diagram describes interaction among classes in terms of an exchange of messages over time. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. The basic symbols and notations used are:

Object : UML object symbols illustrate class roles, but doesn’t list object attributes.

: Activation box represents the time an object needs to complete a task.

Sequence diagram of Customer

Contacts

User

SMS

Mobile Control

Missed Calls

Contacts

Applications

Registered

Missed calls

SMS

Mobile Data

Wi-Fi

Flight mode

Mode

**7.1.4 Activity Diagram:-**

An activity diagram illustrates the dynamic nature of a system by modeling the flow of control from activity to activity. An activity represents an operation on some class in the system that results in a change in the state of the system. Typically, activity diagrams are used to model workflow or business processes and inter operation. Because an activity diagram is a special kind of state chart diagram, it uses some of the same modeling conventions. Activity diagrams are constructed from a limited number of shapes connected with arrows. The most important shape types:

* Rounded rectangles represent activities.
* Diamonds represent decisions.
* Bars represent the start (split) or end (join) of concurrent activities.
* A black circle represents the start (initial state) of the workflow.
* An encircled black circle represents the end (final state).

The basic activity symbols and notations are:

**Table: Activity Diagram Symbols Used**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
|  | Actions state represents the no interruptible action of objects. |
|  | Action flow arrows illustrate the relationships among action states. |
|  | A filled circle followed by an arrow represents the initial action. |
|  | An arrow pointing to a filled circle nested inside another circle represents the final action state. |

Set password

Check password

Contacts

Missed calls

SMS

Sent the message

**Fail**

**Success**

Control Mobile controle

**Figure: Activity diagram for Authentication of Customer**

The above activity diagram of authentication of user represents the flow of events.

While sub-location in-charge is logging into the application:

1. Enter the password.
2. Entered password are valid or not will be checked by the system.
3. If the password is correct application will send the required details.

**7.1.5 Entity –Relationship Diagrams:-**

An Entity–Relationship Diagram is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data – an object or concept about which data is stored. A relationship is how the data is shared between entities.

The logical structure of the database can be expressed graphically with E-R diagrams. Its components are:

* A Rectangle represents an Entity set.
* An Ellipse represents an Attributes.
* A Diamond represents a Relationship Set.

**E-R DIAGRAMS**

has

Message

Pass word

Phone number

**7.1.6 System Architecture**

3-Tier architecture generally contains

* UI or Presentation Layer.
* Business Access Layer (BAL) or Business Logic Layer.
* Data Access Layer (DAL).

**Presentation Layer**

**Business Access Layer**

**Data Access Layer**

**Database**

**Figure 5.1 System Architecture**

**Presentation Layer (UI)**: Presentation layer contains pages like .php where data is presented to the user or input is taken from the user.

**Business Access Layer (BAL) or Business Logic Layer:** BAL contains business logic, validations or calculations related with the data, if needed.

**Data Access Layer (DAL)**: DAL contains methods that helps business layer to connect the data and perform required action, might be returning data or manipulating data (insert, update, delete etc.).

**7.1.7 Module Description:-**Following are the list of modules, which have been used to develop this application.

**Functional Modules**:-The project has been divided into following modules:

1. Home Page
2. Get Contacts.
3. Get Missed Calls.
4. Get Unread message.
5. Get IMEI Number.
6. Turn on/off Wi-Fi.
7. Turn on/off Mobile data.
8. Turn on Airplane mode.
9. Reboot System.
10. System Alarm
11. Change Password.
12. Change Phone number.
13. Inbox.
14. Sent Message.
15. Call logs.

**Home page:**

Home page will display the slider images of Missed call, Unread SMS, Contacts, Mobile Data, Flight Mode And Wi-Fi.

**Get Contacts:**

This Module is used send the contacts when the message arrives with the demand of contact. And if the user is authorized person it sends requested contact through the message .

**Get Missed call:**

This Module is used send the missed calls when the message arrives with the demand of Missed call. And if the user is authorized person it sends through the message .

**Get Unread SMS:**

This Module is used send the unread message when the message arrives with the demand of unread SMS. And if the user is authorized person it sends the unread SMS.

**Contacts:**

It displays the contacts from the phone book.

**Incoming calls:**

It displays the incoming calls from the call log.

**Outgoing calls:**

It displays the outgoing calls from the call log.

**Missed calls:**

It displays the missed calls from the call log.

**Inbox:**

It displays the messages from inbox.

**Sent Message:**

It displays the message from sent message.

**Mobile data:**

This module helps us to control the mobile data by sending message. By using this module we can turn on/off the mobile data.

**Wi-Fi:**

This module helps us to control the Wi-Fi by sending message. By using this module we can turn on/off the Wi-Fi.

**8. CODE EFFICIENCY ANALYSIS**

Software quality measurement is about quantifying to what extent a system or software possesses desirable characteristics. This can be performed through qualitative or quantitative means or a mix of both. In both cases, for each desirable characteristic, there are a set of measurable attributes the existence of which in a piece of software or system tend to be correlated and associated with this characteristic. For example, an attribute associated with portability is the number of target-dependent statements in a program. More precisely, using the [Quality Function Deployment](http://en.wikipedia.org/wiki/Quality_Function_Deployment) approach, these measurable attributes are the "how’s" that need to be enforced to enable the "what’s" in the Software Quality definition above.

The structure, classification and terminology of attributes and metrics applicable to software quality management have been derived or extracted from the [ISO 9126-3](http://en.wikipedia.org/wiki/ISO_9126) and the subsequent ISO/IEC 25000:2005 quality model. The main focus is on internal structural quality. Subcategories have been created to handle specific areas like business application architecture and technical characteristics such as data access and manipulation or the notion of transactions.

**8.1 Optimization of Code**

The dependence tree between software quality characteristics and their measurable attributes is represented in the diagram on the right, where each of the 5 characteristics that matter for the user (right) or owner of the business system depends on measurable attributes (left):

Application Architecture Practices

Coding Practices

Application Complexity

Documentation

Portability

Technical & Functional Volume

One of the founding member of the [Consortium for IT Software Quality](http://en.wikipedia.org/wiki/CISQ), the OMG ([Object Management Group](http://en.wikipedia.org/wiki/Object_Management_Group)), has published an article on "How to Deliver Resilient, Secure, Efficient, and Easily Changed IT Systems in Line with CISQ Recommendations" that states that correlations between programming errors and production defects unveil that basic code errors account for 92% of the total errors in the source code. These numerous code-level issues eventually count for only 10% of the defects in production. Bad software engineering practices at the architecture levels account for only 8% of total defects, but consume over half the effort spent on fixing problems, and lead to 90% of the serious reliability, security, and efficiency issues in production.

**Code-based analysis**

Software quality measurement is about quantifying to what extent a system or software rates along these dimensions. The analysis can be performed using a qualitative or quantitative approach or a mix of both to provide an aggregate view [using for example weighted average(s) that reflect relative importance between the factors being measured.

Structural quality analysis and measurement is performed through the analysis of the [source code](http://en.wikipedia.org/wiki/Source_code), the [architecture](http://en.wikipedia.org/wiki/Software_architecture), [software framework](http://en.wikipedia.org/wiki/Software_framework), [database schema](http://en.wikipedia.org/wiki/Database_schema) in relationship to principles and standards that together define the conceptual and logical architecture of a system. This is distinct from the basic, local, component-level code analysis typically performed by [development tools](http://en.wikipedia.org/wiki/Development_tool) which are mostly concerned with implementation considerations and are crucial during [debugging](http://en.wikipedia.org/wiki/Debugging) and [testing](http://en.wikipedia.org/wiki/Software_testing) activities.

**Reliability**

The root causes of poor reliability are found in a combination of non-compliance with good architectural and coding practices. This non-compliance can be detected by measuring the static quality attributes of an application. Assessing the static attributes underlying an application’s reliability provides an estimate of the level of business risk and the likelihood of potential application failures and defects the application will experience when placed in operation. Assessing reliability requires checks of at least the following software engineering best practices and technical attributes. Depending on the application architecture and the third-party components used (such as external libraries or frameworks), custom checks should be defined along the lines drawn by the above list of best practices to ensure a better assessment of the reliability of the delivered software.

**Efficiency**

As with Reliability, the causes of performance inefficiency are often found in violations of good architectural and coding practice which can be detected by measuring the static quality attributes of an application. These static attributes predict potential operational performance bottlenecks and future scalability problems, especially for applications requiring high execution speed for handling complex algorithms or huge volumes of data.

Assessing performance efficiency requires checking at least the following software engineering best practices and technical attributes:

Application Architecture Practices

**Security**

Most security vulnerabilities result from poor coding and architectural practices such as SQL injection or cross-site scripting. These are well documented in lists maintained by CWE, and the SEI/Computer Emergency Center [(CERT)](http://en.wikipedia.org/wiki/CERT_Coordination_Center) at Carnegie Mellon University.

Assessing security requires at least checking the following software engineering best practices and technical attributes:

Application Architecture Practices

Multi-layer design compliance

Security best practices (Input Validation, SQL Injection, Cross-Site Scripting, etc.)

Programming Practices

Error & Exception handling

Security best practices (system functions access, access control to programs)

**Maintainability**

Maintainability includes concepts of modularity, understandability, changeability, testability, reusability, and transferability from one development team to another. These do not take the form of critical issues at the code level. Rather, poor maintainability is typically the result of thousands of minor violations with best practices in documentation, complexity avoidance strategy, and basic programming practices that make the difference between clean and easy-to-read code vs. unorganized and difficult-to-read code.

Maintainability is closely related to Ward Cunningham's concept of [technical debt](http://en.wikipedia.org/wiki/Technical_debt), which is an expression of the costs resulting of a lack of maintainability. Reasons for why maintainability is low can be classified as reckless vs. prudent and deliberate vs. inadvertent, and often have their origin in developers' inability, lack of time and goals, their carelessness and discrepancies in the creation cost of and benefits from documentation and, in particular, maintainable [source code](http://en.wikipedia.org/wiki/Source_code).

**Size**

Measuring software size requires that the whole source code be correctly gathered, including database structure scripts, data manipulation source code, component headers, configuration files etc. There are essentially two types of software sizes to be measured, the technical size (footprint) and the functional size: There are several [software technical sizing](http://en.wikipedia.org/wiki/Software_Sizing) methods that have been widely described. The most common technical sizing method is number of Lines Of Code (#LOC) per technology, number of files, functions, classes, tables, etc., from which backfiring Function Points can be computed;

The most common for measuring functional size is [Function Point Analysis](http://en.wikipedia.org/wiki/Function_Point). Function Point Analysis measures the size of the software deliverable from a user’s perspective. Function Point sizing is done based on user requirements and provides an accurate representation of both size for the developer/estimator and value (functionality to be delivered) and reflects the business functionality being delivered to the customer. The method includes the identification and weighting of user recognizable inputs, outputs and data stores.

**9. VERIFICATION CHECKS**

**Verification:-**

**A verification check ensures that data is correctly transferred into a computer from the medium that it was originally stored on.** Verification checks are usually used to check that information written on a data collection form has been correctly typed into a computer by a data entry worker.

**Methods of Verification:-**

The two most common methods of verification are:

**On-Screen Prompts**: After a user has entered some data it is redisplayed on the screen. The user is prompted to read the data and confirm that it has been entered correctly. If the user has entered any data incorrectly he should respond that the data is inaccurate and retype the incorrect parts.

**Dual Input**: This method is used when data is entered at the keyboard. The data to be entered is typed in twice by two different operators. The two copies of the data are then compared. Any differences are detected. The operators will be prompted to retype the sections that differ until both copies agree. When the two copies agree it is assumed by the computer that the data has been entered correctly.

**Validation:-**

**A validation check is an automatic check made by a computer to ensure that any data entered into the computer is sensible.** A validation check does not make sure that data has been entered correctly. It only ensures that the data is sensible. For this reason validation checks are not usually as effective as verification checks. They can however be carried out automatically by a computer and therefore require less work by computer operators making them cheaper to use.

**Methods of Validation:-**

There are many different methods of validation. The most appropriate method(s) to use will depend upon what data is being entered. The most common methods are listed here.

**Presence Check**: Checks that data has been entered into a field and that it has not been left blank. E.g. check that a surname is always entered into each record in a database of addresses.

[**Type Check**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Type Check) : Checks that an entered value is of a particular type. e.g. check that age is numeric.

[**Length Check**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Length Check) : Checks than an entered value e.g. surname is no longer than a particular number of characters.

[**Range Check**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Range Check) : Checks that an entered value falls within a particular range. For example the age of a person should be in the range 0 to 130 years.

[**Format Check**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Format Check) : Checks that an entered value has a particular format. e.g. a new-style car registration number should consist of a letter followed by 1 to 3 numbers followed by 3 letters.

[**Check Digit**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Check Digits) : A check digit is a digit attached to the end of a string of digits. It is calculated from the other digits and used to help ensure that the whole string is inputted correctly.

[**Parity Check**](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/validate.htm#Parity Checks) : Used in data communications to ensure that data is not corrupted when it is sent down a transmission medium such as a telephone line.

**10. VALIDATION CHECKS**

Validation checks can be performed by any piece of software. However you are most likely to encounter them when creating a new database. Sophisticated database packages will let you implement validation checks using **validation rules**. You can provide different validation rules for each different field in the database. Below are some examples of how the common validation methods can be used.

**Type Check**

Databases perform type checks automatically on all entered data. When a database is created each field in the database is given a type. Whenever data is entered into a field the database will check that it is of the correct type, e.g. alphabetic or numeric. If it is not then an error message will be displayed and the data will have to be re-entered. Here are some example field names and appropriate types.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Type** | **Valid Data** | **Invalid Data** |
| Phone number | Numeric | 9876543210 | ABC562 |

Notice that a type check is not a very good validation check. Many of the entries in the Valid Data column in the table pass the type check but are clearly incorrect.

**Length Check**

As with type checks, most databases will automatically perform length checks on any entered data. The length check ensures that the data entered is no longer than a specified maximum number of characters. This is particularly important if a fixed length field is being used to store the data. If this is the case then any extra characters typed that made the data longer than the space available to store it would be lost. Here are some example field names and appropriate maximum lengths.

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Maximum Length** | **Valid Data** | **Invalid Data** |
| Phone number | 10 | 9876543210 | 97887 |

Length checks are usually only performed on alphabetic or alphanumeric data. A similar test can be performed on numeric and date data by using a range check.

**Range Check**

Range checks are used on data made up of numbers or dates which must fall into a particular range. A lower and upper boundary for sensible values is specified. Any values which fall outside of this range will be rejected. Most sophisticated databases will let you set valid ranges for each [field](http://www.hollyfield.kingston.sch.uk/gcseit/GCSE/database.htm#Fields).

Sometimes there is only one boundary required for a particular field. For example the minimum volume of a cube would be zero cubic centimeters, but there is no maximum volume. When there is only one boundary to check the type of check used is known as a **limit check** rather than a range check

**Format Check:-**

Format checks can be performed on data entered into a database field. The format that data entered into a field must be in is specified using an **input mask**. The input mask is made up of special characters which indicate what characters may be typed.In a particular database the following special characters can be used to define an input mask:

0 - A number must be typed in this position.

9 - A number may be typed in this position or it may be left blank.

L - A letter must be typed in this position.

l - A letter may be typed in this position or it may be left blank.

A - A letter or number must be typed in this position.

a - A letter or number may be typed in this position or it may be left blank.

**11. TESTING AND IMPLEMENTATION**

**11.1. TESTING:-**

This document describes the plan for testing the skilled manpower system. All the major testing activities that were undertaken in the project are specified here.

**Test units**

In this project, two levels of testing is performed: Unit Testing and System Testing. The basic units to be tested are:

1. Modules and object operations of database.
2. Modules and objects operation to forms and menus.
3. The testing for these different units will be done independent.

**Features to be tested**

All the functional features specified in the requirements document will be tested and the performance testing will be done.

1. Increase system speed.
2. Decrease saving time response.
3. Decrease the time taken for Request and Response from the server.

**Approaches for Testing**

For unit testing, structural testing based on statement, branch and path coverage criteria will be used as per need in the modules and operations. The goal is to achieve branch and path coverage of more that 95%.

**Test deliverables**

The following documents are required (besides this test plan):

* Unit test report for each unit
* Test case specification for system testing
* The report for system testing
* Error report

**11.1.1 Types of Testing:-**

* Unit Testing
* Functional Testing
* Performance Testing
* Stress Testing
* Structural Testing
* Integration Testing
* Validation Testing
* Black Box Testing
* Output Testing
* User Acceptance Testing
* **Unit Testing**

This unit testing discusses its attention on a part of software called a module. This testing ensures to validate the software within a particular area of the software. This unit testing verifies detailed design description as a guide, important control paths, and helps to recover errors within the boundary of the module. This testing checks the first module to assure as that the data flow properly into and out of the unit under test. The data structure is examined to ensure that the data stored temporarily maintains its integrity during all steps in an algorithm execution.

The boundary region for the module under consideration is tested to ensure the module operates properly and that all the paths in the control structure are exercised. To ensure all statement in the module are executed at least once and finally all error handling.

The proposed system’s each unit are thoroughly tested and implemented successfully. Each unit’s deliverables are accomplished without any errors and every unit gives the anticipated results.

* **Functional Testing**

This case is involved by exercising the code with nominal input values for which the expected results are known. This test was done on each module, say for at the tour advance the sub system level, the subsystem values given as input for which the corresponding subsystem. The output was verified with the existing manual system. The functional test was carried out at all levels.

The proposed system is functionally tested and the results are observed to be correct. When the user jumps a comic page from some number of pages, he/she is redirected to specified page.

* **Performance Testing**

It determines the amount of execution time spent in various parts the unit programs, throughput, and response time and device utilization by the program unit. During the product development, if the execution time crosses the normal execution time for a function, then the statements that cause the control to flow for compiling the codes on other events was deleted and it was made linear throughout the function. The proposed system is auto configured to detect type of browsers and operating systems, so as it can execute in least possible time.

* **Stress Testing**

It determines the strength and limitations of a program by examining the manner in which the program unit breaks also it is performed to intentionally break the unit.

* **Structural Testing**

It is concerned with exercising the internal logic of the program and traversing particular execution paths. This test has been done for each and every module in which the particular portion of code comprises of logical statements is executed that is built into the solution on separate compilation. The proposed system is very well organized so that each logic can be traversed easily. Thus Functional, Performance and Stress testing are called as Black **Box Testing** while Structural Testing is referred as **White Box Testing.**

* **Integration Testing**

This is to develop an incremental strategy that will limit the complexity of interactions among components as they are added to the system, developing an implementation and integration schedule that will make the modules available when needed. In this test, groups of the program modules are tested together to determine if they interface properly. This is done incrementally as they are developed until the program system is tested. In integration testing the product is tested to confirm our view of it at the specification and architectural design stage. Functional testing techniques are mainly used in the integration testing. During integration, we choose portions of the structure tree of the software to put together. Each sub-tree should have some logical reason for being tested: it is either implementing system function or part of the code essential to the function of the rest of the product.

Since the proposed system is in its 4th iteration now, it can be understood that it is very well backward compatible, which makes it easier to integrate new modules into the system.

* **Validation Testing**

Computer input procedures are designed to detect errors in the data at a lower level of detail which is beyond the capability of the control procedures. These are combined with the design of the input process itself, reading the data from the user. At the culmination of this, software is completely assembled as package interfacing errors which have been uncovered and corrected as validation succeeded when the software function in a manner that can be reasonably expected by the customer.

Software Validation is achieved through a series of black box tests that demonstrates conformity with the requirement. Black box testing method focuses on the Functional requirement of the software. The proposed meets all its functional requirements, which makes its validation testing successful.

* **Black Box Testing**

Black box testing attempts to find errors in the incorrect missing function interface errors, errors in the data structure and external data base access, performance errors and initialization and termination errors. A test plan out times the classes of tests to be conducted and a test procedure defines specific test case that are used to demonstrate conformity with the requirements.

In this system, black box test has uncovered all the errors, which have been corrected. The validated software is compared with the configuration review.

As the record values indicates that a certain process or series of process are to be performed on incorrect or non-existent value entered for a field will call the whole of such a record to be rejected showing the reason by means of alerts included in the coding. In this proposed system all the input forms are validated before submission.

If the mandatory fields are left empty then the user is given an alert message to fill those fields with the values and also numeric fields are validated to check that they contain only the numbers so that the characteristics are not allowed, in that case also the alert message is given to the user.

All functions in the proposed system are tested thoroughly during black box testing, and the results are also positive. During its 3rd iteration, the proposed system was very lack in search functions and new releases functions. In its current version these are implemented successfully.

* **Output Testing**

This test is conducted by giving sample value as input then comparing the obtained output with the expected output. It comes under Black Box Testing. The success of it depends on the value, which we are giving.

In the proposed system the sample values are given to all the modules. Sample input value is given to these modules and the outputs from these modules are compared with the expected output. If the output matches with the expected one then the success of the module is determined. This test is one of the crucial tests in this phase. This test is conducted to determine the success of all the modules as a whole, thus it rated the performance of the whole proposed system.

* **User Acceptance Testing**

Acceptance Test has the objective of convincing the user on the validity and readability on the system. It is the moment at which we prove that we have really done what we promised to do. This test is not the same as the one done at various stages of life cycle. It involves customer and intends to check that the tests dispose no errors at all.

**11.1.2 Test Cases Result:-**

A Test case (TC) is a set of test inputs, executions, and expected results developed for a particular object. An excellent test case satisfies the following criteria:

* Reasonable probability of catching an error.
* Does interesting things.
* Not redundant with other tests.
* Neither too simple nor too complex.
* Doesn’t do unnecessary things.

**11.1.3 Test Analysis:-**

The size of the database increases day-by-day, increasing the load on the database back up and data maintenance activity. Training for simple computer operations is necessary for the users working on the system. This can be enhanced for adding new functionalities and report generating method.

**11.2 IMPLEMENTATION:-**

The process of converting information obtained on a subject or unit into coded values for the purpose of data storage, and analysis.

The phase in the software development process that is often referred to interchangeably as coding, development, or implementation is the actual transformation of the requirements into executable form. Each coding unit specifies the code for a particular type of function. After document reviews and approvals, the coding may begin. Using private copies of the code, developers make the changes and add the files specified in the coding unit. Coding is delicate work, and great care is taken so that unwanted side effects do not break any of the existing code. After completion, the code is tested by the developer and carefully reviewed by other experts.

It may take several weeks of detailed study to change a few lines of code without causing unwanted side effects. Indeed, much of the effort in maintenance involves changing code written by another programmer. Because of variation in programmer staff sizes and inevitable turnover, training new programmers is important.

Changes from many coding units are periodically combined together into a so-called common load of the software system. The load is compiled, made available to developers for testing, and installed in the machines. Bringing the changes together is necessary so that developers working on different coding units of a common feature can ensure that their code works together properly and does not break any other functionality.

After all coding units associated with a feature are complete and it has been tested by the developers, the feature is turned over to the integration group for independent testing. The integration group runs tests of the feature according to a new test plan that was prepared in parallel with the integration. Eventually the new code is released. At this stage, maintenance on the code begins.

**Coding Standards**

This document requires or recommends certain practice for developing programs in the PHP language. The objective of this coding standard is to have a positive effect on:

* Avoidance of error/bugs, especially the hard-to-find ones.
* Maintainability, by promoting some proven design principle.
* Maintainability, by requiring or recommending a certain unity of style.
* Performance, by discussing wasteful practices.

**General Rules**

* Every time a recommending is not followed, this must have a good reason. Good reasons do not include personal preferences of style.
* Do not mix code from different providers in one file.

**Naming and Conventions**

* Use US-English for naming identifiers.
* Do not prefix member fields.
* Do not use casing to differentiate identifiers.
* Use abbreviations with care.
* Do not use an underscore in identifiers.
* Name an identifier according to its meaning and not its type.
* Do not add a suffix to a class or struct name.
* Use a noun or a noun phrase to name a class or struct.
* Use a verb (gerund) for naming an event.

**Comments and embedded documentation**

* Each file shall contain a header block.
* Use // for comments.
* All comments shall be written in US English.

**Object Life Cycle**

* Declare and initialize variables close to where they are used.
* If possible, initialize variables at the point of declaration.
* Use a const field to define constant values.
* Do not “shadow” a name in an outer scope.

**Control Flow**

* Do not change a loop variable inside a “for” loop back.
* Update loop variables close to where the loop condition is specified.
* All flow control primitives (if, else, while, for, do, switch) shall be followed by a block, even if it is empty.
* All switch statements shall have a difficult label as the last case label.
* Avoid multiple or conditional return statements.

**Object Oriented Programming**

* Declare all fields (data members) private.
* Provide a difficult private constructor if they are only static methods and properties on a class.
* If you must provide the ability to override a method, make only the most complete overload virtual and define the other operations in terms of it.
* Use a property rather than a method when the member is a logical data member.
* Use a member rather than a property when this is more appropriate.

**Exceptions**

* Only throw exceptions in exceptional situations.
* List the explicit exceptions a method or property can throw.
* Use standard exceptions.
* Throw informational exceptional.
* Throw the most specific exception possible.

**Coding Style**

* Do not mix coding style with in a group of closely related classes or within a model.
* The public, protected and private sections of a class shall be declared in that order.
* Write unary, increments, decrements, function call, subscript and access operators together with their operands.
* Use spaces instead of tabs.

**11.2.1 Implementation Procedure:-**

Operational Evaluation: Assessment of the manner in which the system functions including ease of use response time suitability of information formats overall reliability and level of utilization.

Organization Impact: Identification and measurement of benefits to the organization in such areas as financial concerns operational efficiency and competitive impact.

User Manager Assessment: Evaluation of the attitudes of senior and user managers within the organization as well as end-users.

Development Performance: Evaluation of the development process in accordance with such yardstick as overall development time and effort conformance to budgets and standard and other project management criteria. Unfortunately system evaluation does not always receive the attention it merits. Where properly managed however it provides a great deal of information that can improve the effectiveness of subsequent application efforts.

**11.2.2 User Training:-**

User are given a one week training to get used to in this software and are given both theoretical as well as practical knowledge as required by the user.In this training period the user all allowed to clear all his doubts and are taught with various forms and what kind of necessary and important note has to be taken down while using this software.

**11.2.3 Operational Documentation:-**

Operational documentation is the process of both installation of the software and training guide provided to the end user through various forms, links or the videos Implementation comes after the successful completion of testing verification. Prior to starting the project different processes were studied and based on this the functional needs for the software were arrived at.

**12. COST ESTIMATION OF PROJECT**

The costs of a constructed facility to the owner include both the initial capital cost and the subsequent operation and maintenance costs. Each of these major cost categories consists of a number of cost components.

The capital cost for a construction project includes the expenses related to the initial establishment of the facility, Planning and feasibility studies, Architectural and engineering design Construction, including materials, equipment and labor Construction financing Insurance and taxes during construction Owner's general office overhead Equipment and furnishings not included in construction Inspection and testing.

In total the cost of this project is Rs.80, 000 which has been initially fixed between the customer and the organization. From day one to the end of the project all the cost has been included and all the commitment has been fulfilled according to the customer and the project cost is little low as per the requirement of the client.

**13. TIME SCHEDULING PROJECT**

“Software project scheduling is an activity that distributes estimated efforts across the planned duration by allocating the effort to specific software engineering tasks. Time line chart shown below, tells the story of my each and every week spent behind the system design.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Week  Activity Month | 1 | 2 | 3 | 4 | 1 | 2 | | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Feb | | | | March | | | | | April | | | | May | | | |
| Analysis |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |
| Learning Process |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |
| Design |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |
| Coding  &  Testing |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |
| Reporting |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |
| Final Documentation |  |  |  |  |  | |  |  |  |  |  |  |  |  |  |  |  |

**14. FUTURE SCOPE AND FUTURE ENHANCEMENTS**

In this project Remote Access with Anti-theft, only few features are implemented. In future, We can include the features like switch on/off mobile, turn on/off Bluetooth, tracking of incoming and outgoing calls, sending and receiving messages, tracking the current location of the mobile.

The project is able to successfully incorporate all the requirements specified by the user. Proper care has been taken during database design to maintain data integrity and to avoid data redundancy. A client side validation has also been done with utmost care by considering all the possibilities and the requirements of different users to avoid data inconsistency. The user is provided with a very friendly interface, hiding all the technical intricacies. Design procedures and user manuals are also included in the project to help the users better understand the system. The project is designed and coded in such a way that any further modifications that are needed in the future can be easily implemented without affecting the functionality of the system. The technical documentation provided in the project report helps the application developers understand the internal architecture of the system and thus assists them in enhancing the system.

This project is purely user friendly and platform dependent, so user can run this tool in android platform. It is very easy to implement or add many features to this tool. Finally it is a very need full and simple tool for organization.

**15. LIST OF ACTIVITY**

1.Home Page

2.Get Contacts

3.Get Missed Calls

4.Get Unread message

5.Get IMEI Number

6. Turn on/off Wi-Fi

7.Turn on/off Mobile data

8.Turn on Airplane mode

9.Reboot System

10.Remote Alarm

11. Set Password

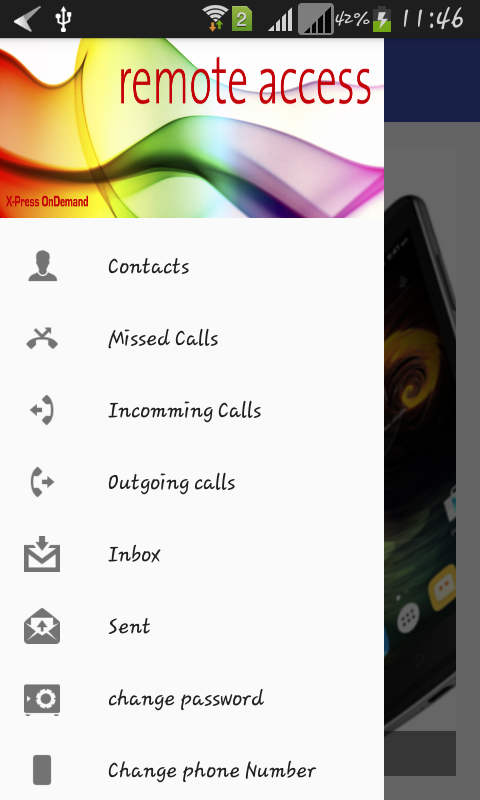
12. Set Phone number

13.Inbox

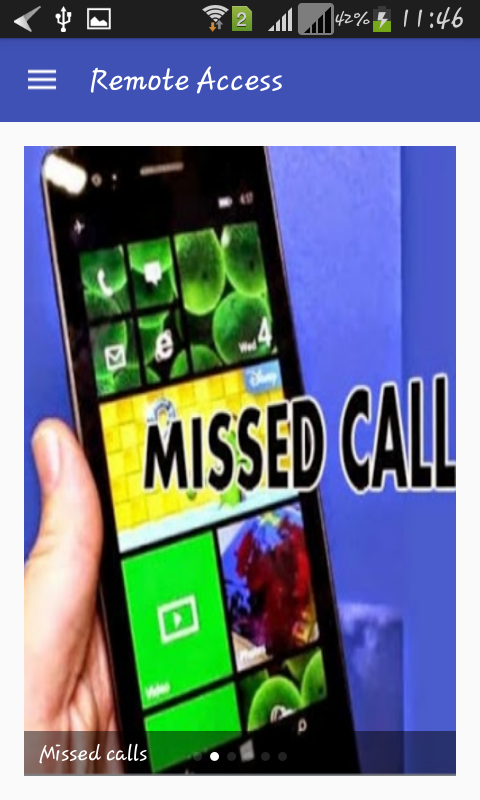
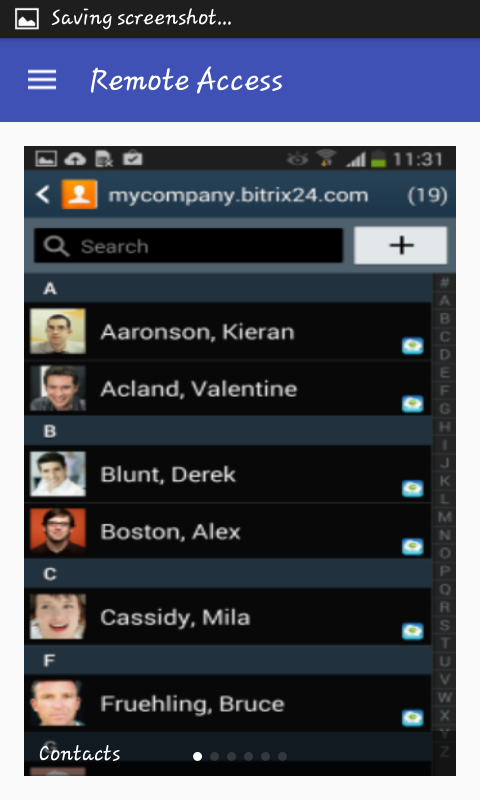
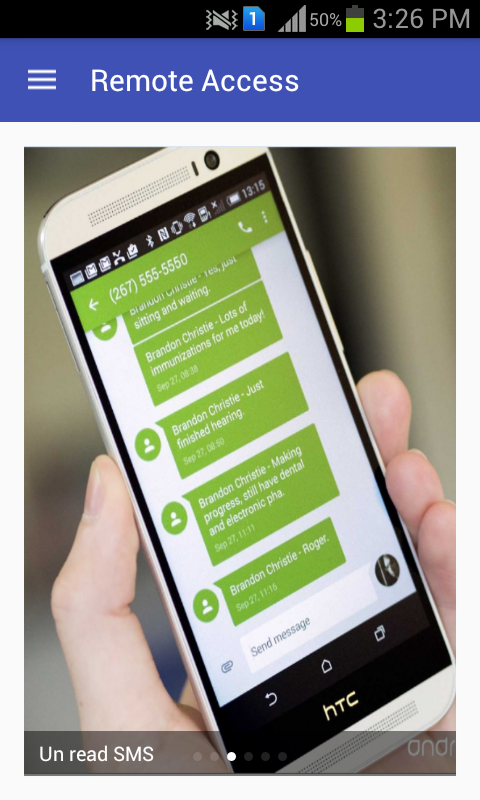
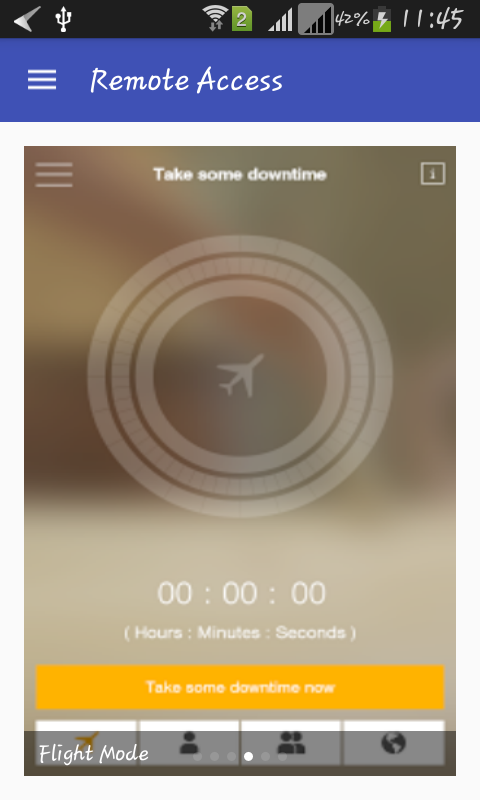
14.Sent Message

15.Call logs

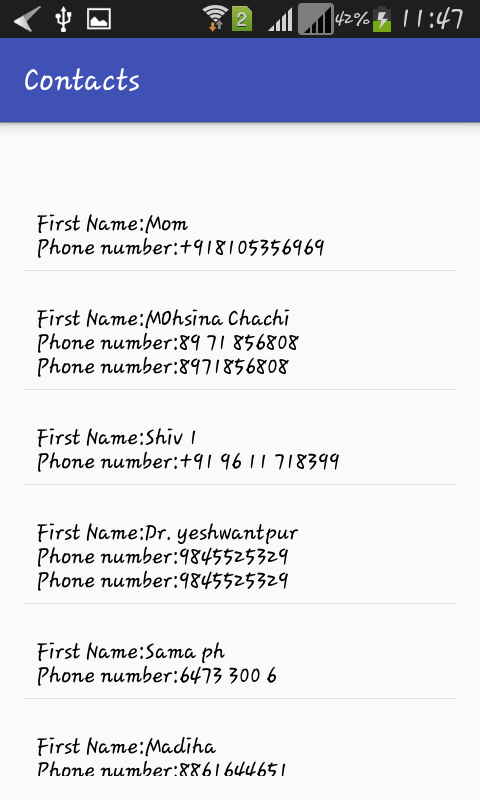
**16. LIST OF SCREENS**

­

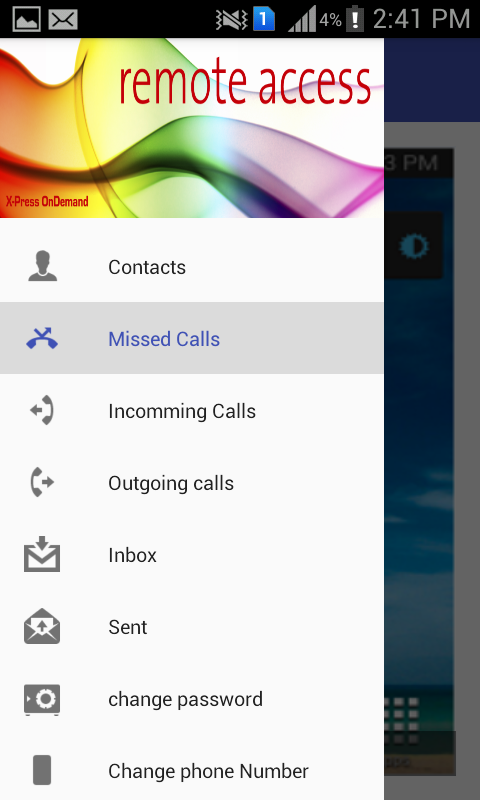
Home page

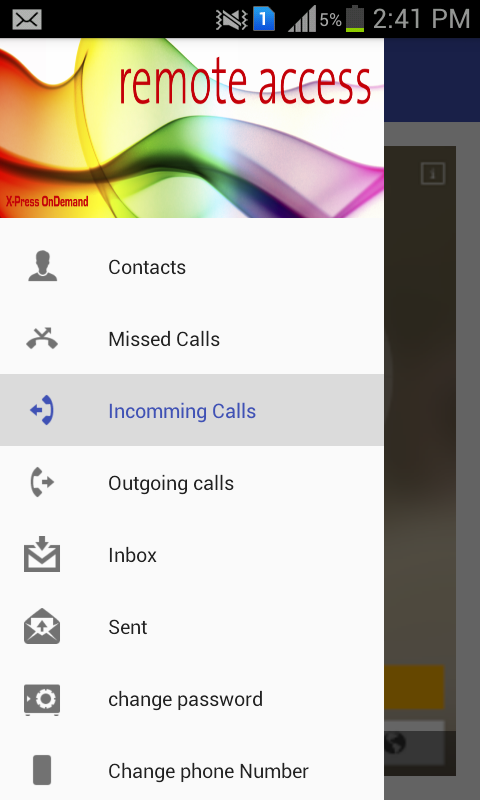
**Slide Screen of the home page**

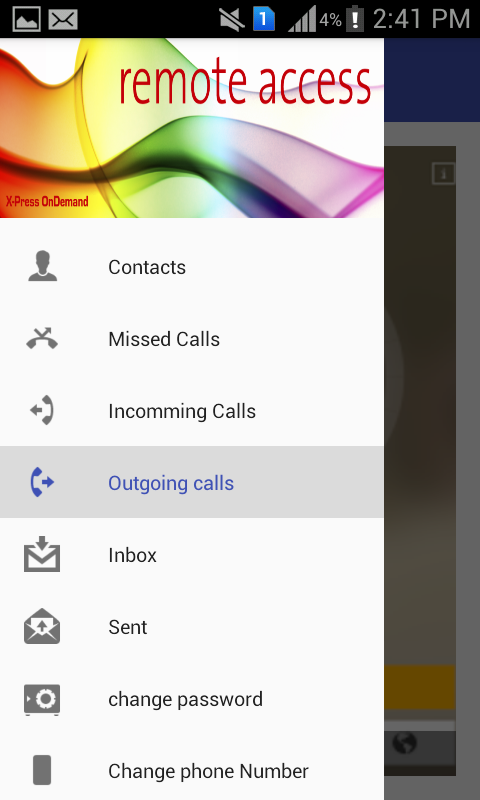
**Contacts**

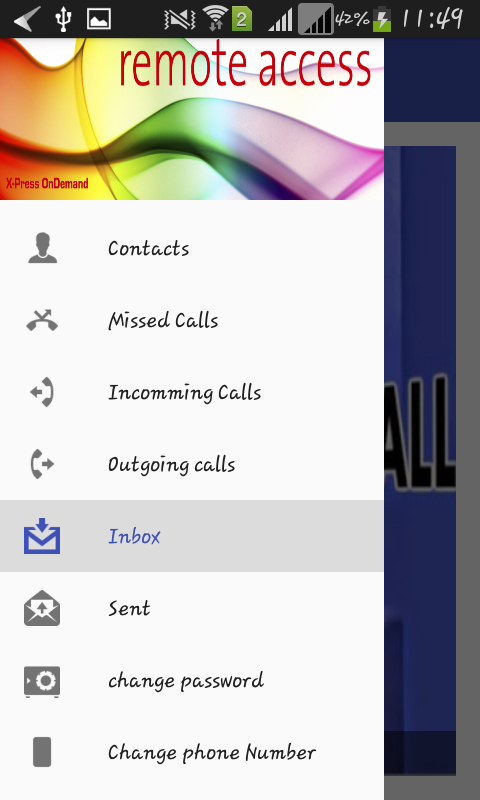
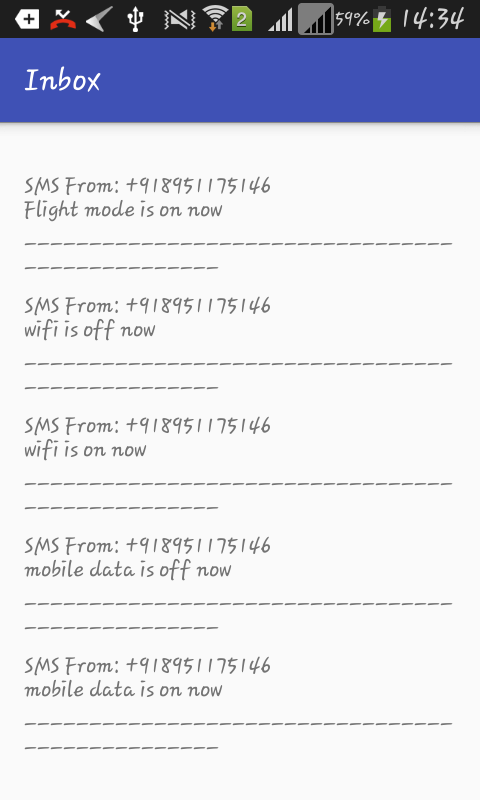
**Missed Calls**

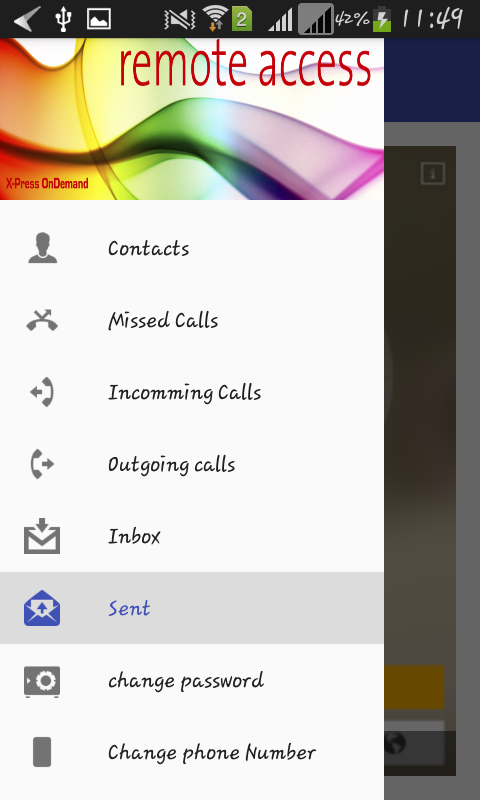
**Incoming Calls**

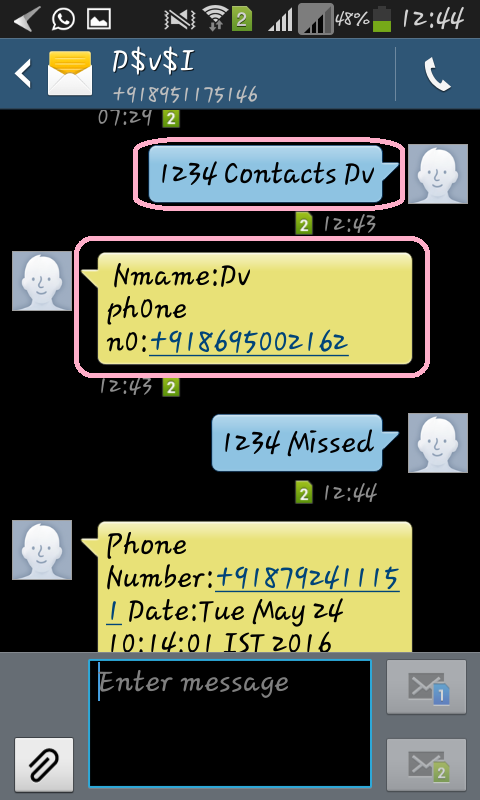
**Outgoing Calls**

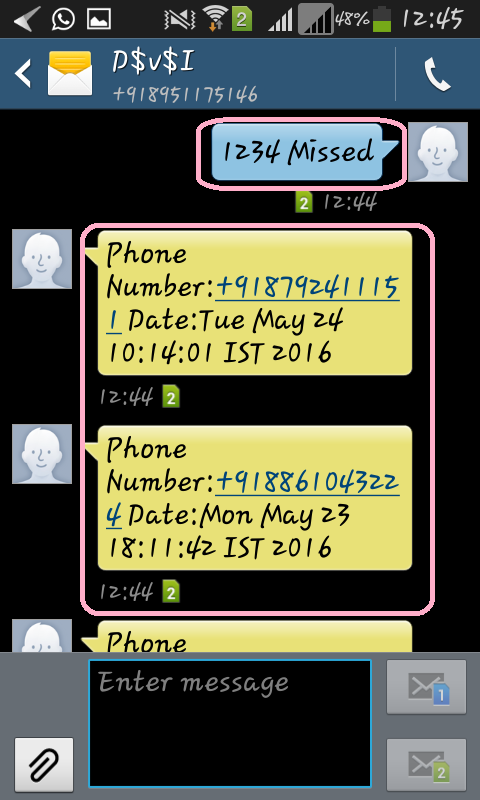
**Inbox**

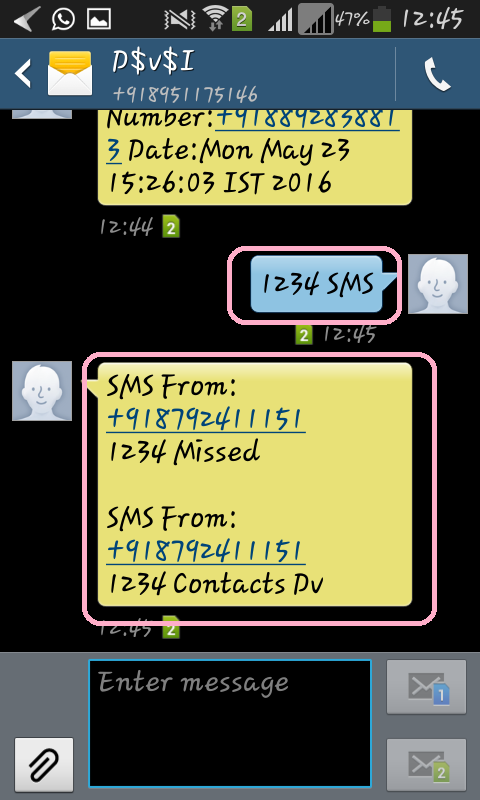
**Sent Message**



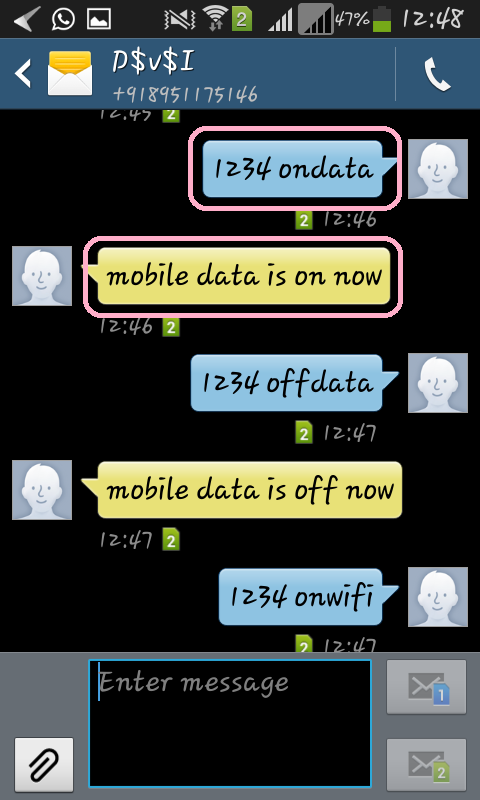
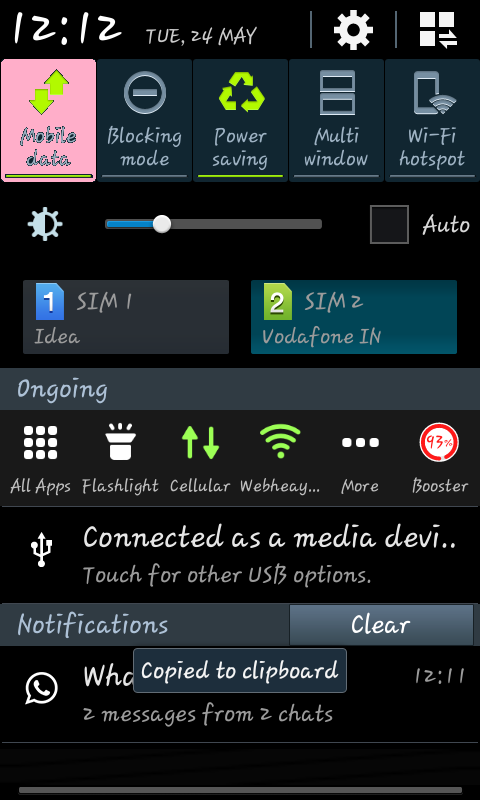
**Get Contact**

****

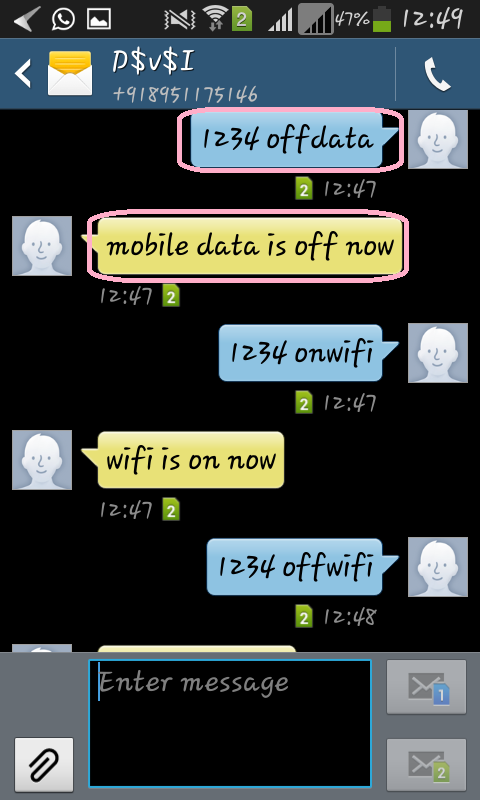
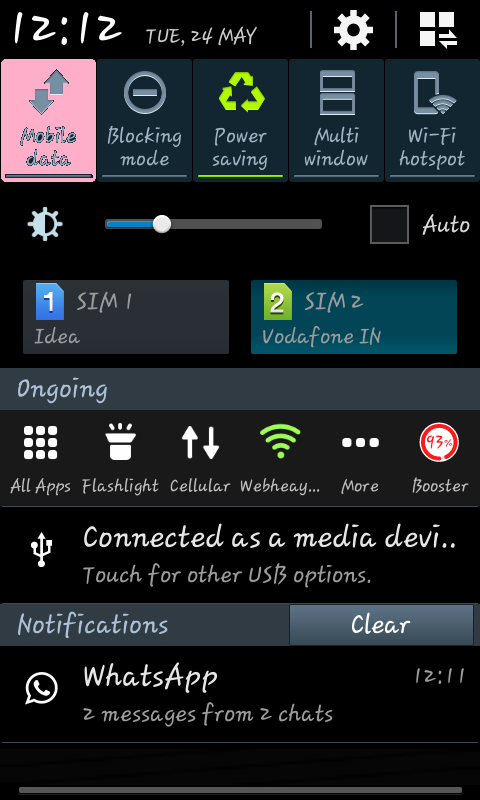
**Get Missed Calls**

****

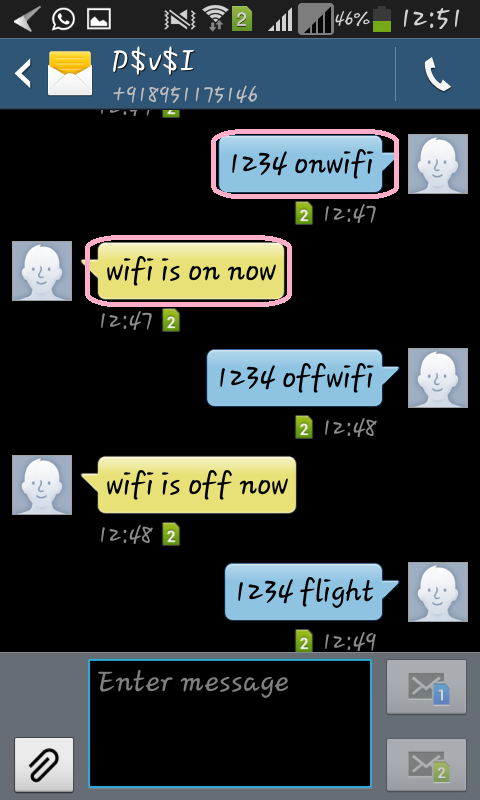
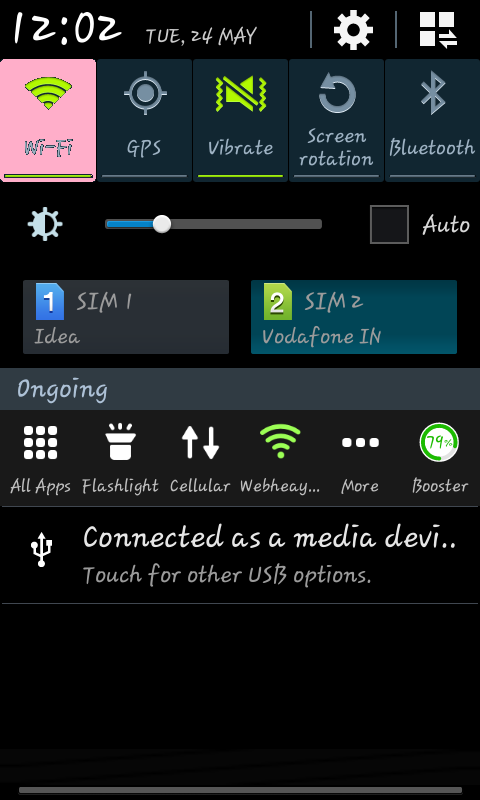
**Get Unread SMS**

** **

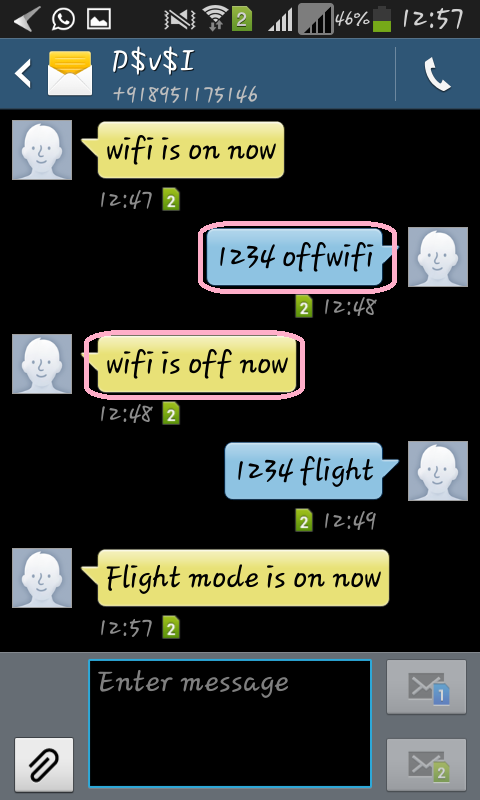
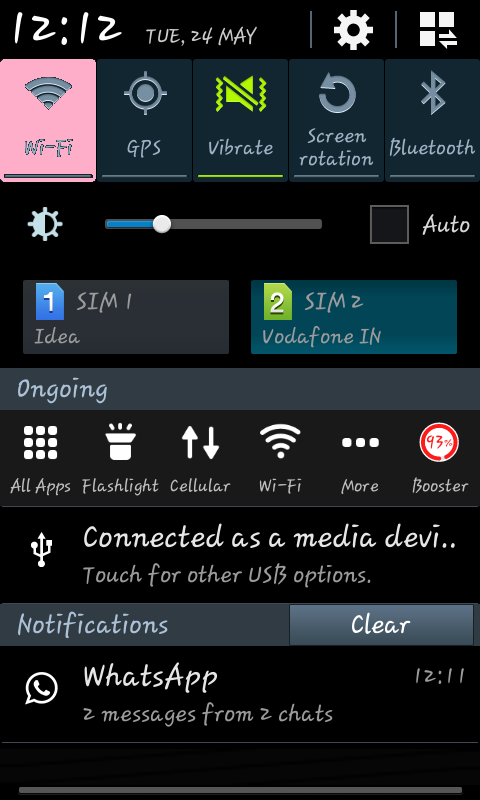
**On Mobile Data**

** **

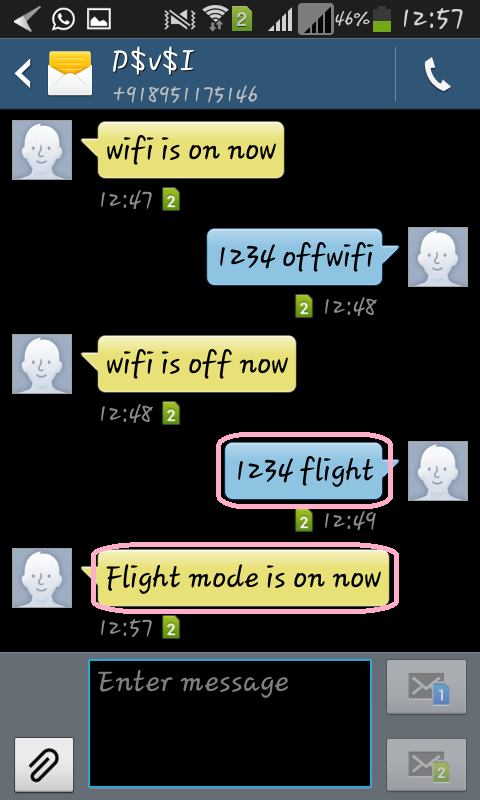
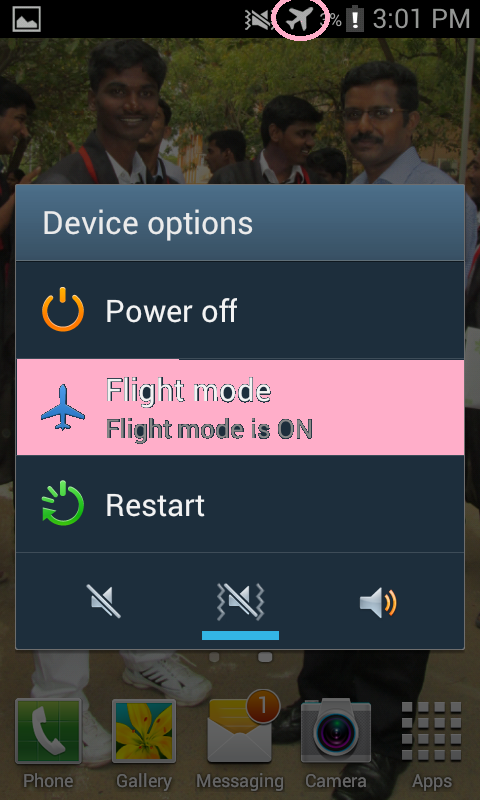
**Turn on Mobile Data**

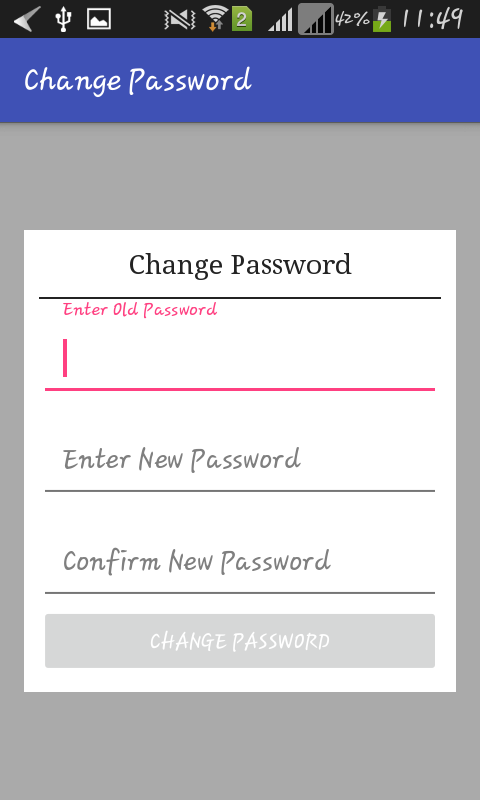
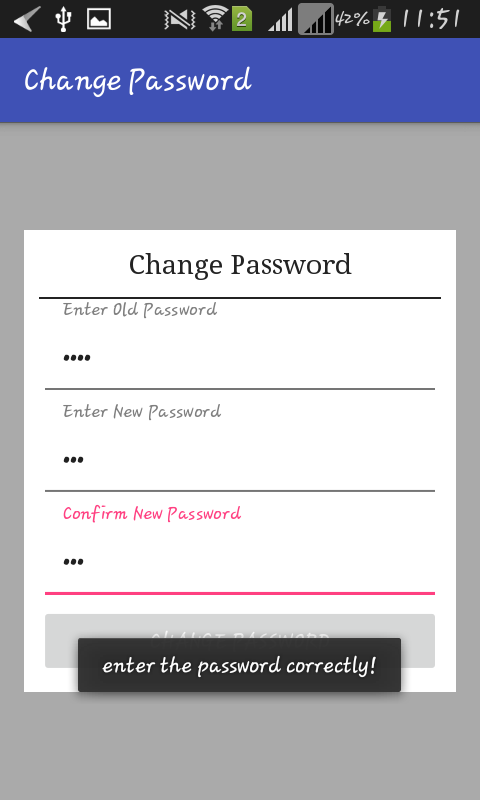
** **

**Turn On Wi-Fi**

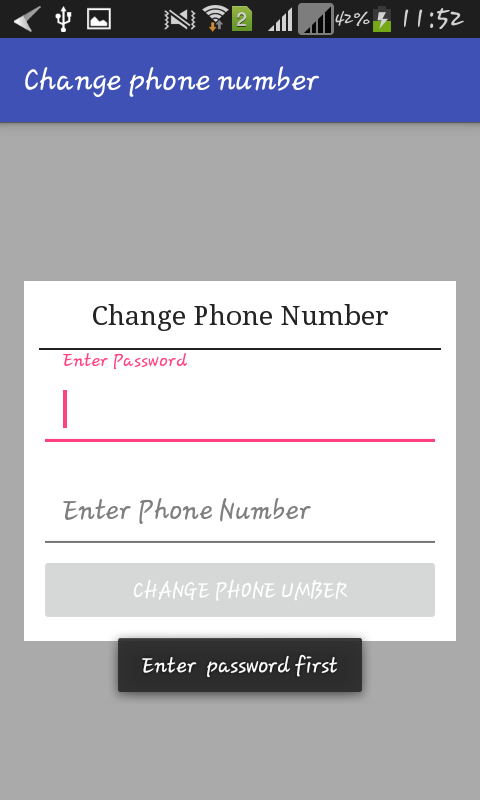
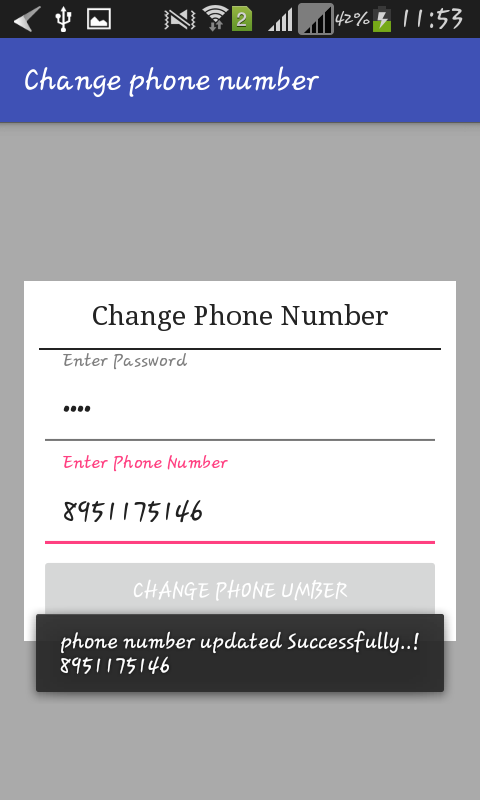
** **

**Turn off Wi-Fi**

** **

** **

**Change Password**

** **

**Change Phone number**

**17. LIST OF TABLE**

**Table: Change Password**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type/Size** | **Constraint** | **Description** |
| Id | Number | Primary Key | unique id |
| Password | varchar(15) | not null | password of the user |

**Table: Change Phone number**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field Name** | **Data Type/Size** | **Constraint** | **Description** |
| Id | Number | Primary Key | unique id |
| Phone number | varchar(15) | not null | Phone number to send the details when mobile get restart |

**18. CONCLUSION**

The project is able to successfully incorporate all the requirements specified by the user. Proper care has been taken during database design to maintain data integrity and to avoid data redundancy. A client side validation has also been done with utmost care by considering all the possibilities and the requirements of different users to avoid data inconsistency. The user is provided with a very friendly interface, hiding all the technical intricacies. Design procedures and user manuals are also included in the project to help the users better understand the system. The project is designed and coded in such a way that any further modifications that are needed in the future can be easily implemented without affecting the functionality of the system. The technical documentation provided in the project report helps the application developers understand the internal architecture of the system and thus assists them in enhancing the system.

This project is purely user friendly and platform dependent, so user can run this tool in android platform. It is very easy to implement or add many features to this tool. Finally it is a very need full and simple tool for organizations.

**19. BIBLIOGRAPHY**

Android Programming : Bill Philips

The Definitive Guide to  **SQLite** **:** Mike Owens

#### Java , the Complete Reference **:** Patrick Naughton & Herbert Schildt

Java Servlet Programming **:** Jason Hunter & William Crawford

XML **:** Martin Fowler

**20. APPENDICES**

www.sun.java.com

www.wikipidia.com

www.JavaTpoint.com

www.w3schools.com

www.wikipidia.com

www.google.com