**FRIEND LOCATION ALERT**

*Submitted in the partial fulfillment of the requirements*

*for the award of the degree in*

**BACHELOR OF TECHNOLOGY**

In

**Computer Science & Engineering**

By

P.Smruthi (10H51A0541)

S.Lata (105H1A0530)

M.Chaitra Sailaja (10H51A0532)

Under the guidance of

Mrs.E.V.N.Jyothi



**DEPARTMENT OF COMPUTER SCIENCE &ENGINEERING**

**CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

**Kandlakoya (V), Medchal Road, Hyderabad**

** CMR COLLEGE OF ENGINEERING & TECHNOLOGY**

Kandlakoya (V), Medchal Road, Hyderabad

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **P.SMRUTHI (10H51A0541), S.LATA (105H1A0530), M.CHAITRA SAILAJA (10H51A0532)** of III B.Tech CSE, who carried out the project entitled **“FRIEND LOCATION ALERT”** is submitted in partial fulfillment of the requirements for the award of the degree in BACHELOR OF TECHNOLOGY in Computer Science & Engineering, CMR College of Engineering & Technology, Hyderabad during the Academic Year 2013-2014

**Mrs.E.V.N.Jyothi Mr. K. Srinivas Rao**

**Professor**

**Internal Guide**  **Head of the Department**

**Submitted for Viva Voce Examination held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner External Examiner**

**ACKNOWLEDGEMENT**

It plunges us in exhilaration in taking privilege in expressing our heartfelt gratitude to our Project Coordinator, **Mrs. Sahana D** her extreme support, guidance and timely inspiration in making this project successful.

We also reveal our deepest gratitude to our project guide, **E.V.N.Jyothi,** for her extreme support, guidance and timely inspiration in making this project successful.

We express our sincere gratitude to **Idealabs of Renaissance Soft Labs**, for its kind acceptance in providing us project successfully and providing us with encouraging guidance throughout.

Our sincere thanks to the Head of the Department, Computer Science & Engineering, **Prof. K. Srinivas Rao,** CMR College of Engineering & Technology, JNTUH for providing us his assistance and necessary facilities for carrying out this project successfully.

It plunges us in exhilaration in taking privilege in expressing our heartfelt gratitude to our Principal **Prof. Dr. M. Rama Linga Reddy**, CMR College of Engineering & Technology, JNTUH, for helping us finishing out this project successfully.

With pleasure, we also wish to convey our thanks to all those, who directly or indirectly contributed their assistance in finishing this project successfully.

**P.Smruthi (10H51A0541)**

**S.Lata (105H1A0530)**

**M.Chaitra Sailaja (10H51A0532)**

**CONTENTS**

**Title**

**ABSTRACT**

1. **INTRODUCTION**

1. **BACKGROUND WORK**
   1. Existing System
      1. Disadvantages
2. **PROPOSED SYSTEM**
   1. Advantages
   2. Module Description
   3. Feasibility Study
      1. Technical Feasibility
      2. Operational Feasibility
      3. Economic Feasibility
   4. SDLC Model
3. **SOFTWARE REQUIREMENT SPECIFICATION** 
   1. Software Interfaces
   2. Hardware Interface
   3. Communication Interface
   4. Functional Requirements
   5. Non Functional Requirements
4. **LITERATURE SURVEY**
   1. Java
   2. Eclipse Galileo
   3. Android
   4. SQLite
5. **SYSTEM DESIGN** 
   1. Introduction
   2. Architecture
   3. Entity-Relationship Diagrams
   4. UML Design
6. **USER INTERFACES OR OUTPUT SCREENS**
   1. Login Page
   2. Registration Page
   3. Forgot password page
   4. Profile Page
   5. Search Page
   6. Friend list page
   7. Tracking page
7. **TESTING**
   1. Testing Introduction
   2. Unit Testing
   3. Integration Testing
   4. Validation Testing
   5. White Box Testing
   6. Black Box Testing
   7. Test Cases
8. **CONCLUSION AND FUTURE ENHANCEMENT**

* **BIBLIOGRAPHY**
* **REFERENCES**

**ABSTRACT**

Friend location alert is a mobile application which is used to give an alert when our contacts using this app are in nearby location. The user can instantly call or message them. Using this application, we can locate our friends on the map. Many times we wish to know our friends location without calling or messaging them. This is possible using this application. This is a very useful application to be connected with our friends.

The main objective of this application is we automatically get a notification when our contacts are in nearby location within a particular proximity. The radius can be altered based on the user’s need. User is given an option of instant calling and instant messaging. Friends can be located on map. If the user doesn’t wish to share his location with anyone of his contacts, there is an option called track on/off beside the contact name.

The user interfaces are mobile specific to give distributed accessibility for the overall system. The internal database has been selected as SQLite. The SQLite was a choice as it provides the constructs of high-level reliability and security with very less space occupation in the smart phones. The total front end was dominated using the JAVA and Android technologies. At all proper levels high care was taken to check that the system manages the data consistency with proper business rules or validations. The authentication and authorization was crosschecked at all the relevant stages.

CHAPTER – 1

INTRODUCTION

1. **INTRODUCTION**

**ANDROID**

Android is an operating system based on the Linux kernel,and designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc., which Google backed financially and later bought in 2005, Android was unveiled in 2007 along with the founding of the Open Handset Alliance—a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices. The first publicly available Smartphone running Android, the HTC Dream, was released on October 22, 2008.

The user interface of Android is based on direct manipulation, using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching and reverse pinching to manipulate on-screen objects. Internal hardware such as accelerometers, gyroscopes and proximity sensors are used by some applications to respond to additional user actions, for example adjusting the screen from portrait to landscape depending on how the device is oriented. Android allows users to customize their home screens with shortcuts to applications and widgets, which allow users to display live content, such as emails and weather information, directly on the home screen. Applications can further send notifications to the user to inform them of relevant information, such as new emails and text messages.

**WHAT IS MOBILE TECHNOLOGY?**

A mobile phone (also known as a cellular phone, cell phone, and a hand phone) is a device that can make and receive telephone calls over a radio link while moving around a wide geographic area. It does so by connecting to a cellular network provided by a mobile phone operator, allowing access to the public telephone network. Mobile technology is the technology used for cellular communication. Mobile code division multiple access (CDMA) technology has evolved rapidly over the past few years. Since the start of this millennium, a standard mobile device has gone from being no more than a simple two-way pager to being a mobile phone, GPS navigation device, an embedded web browser and instant messaging client, and a handheld game console. Many experts argue that the future of computer technology rests in mobile computing with wireless networking. Mobile computing by way of tablet computers are becoming more popular. Tablets are available on the 3G and 4G networks.

**ADVANTAGES OF ANDROID**

* *Multitasking* – Android phones can run many applications, it means you can browse, Facebook while listened to the song.
* *Ease of Notification* – Any SMS, Email, or even the latest articles from an RSS Reader, there will always be a notification on the Home Screen Android phone, do not miss the LED indicator is blinking, so you will not miss a single SMS, Email or even Misscall .
* *Can install a modified ROM* – not satisfied with the standard view of Android, do not worry there are many custom ROM that can be used in your mobile phones Android .
* *Widget* – absolutely right, with the widgets on the homescreen, You can easily access a variety of settings quickly and easily.
* *Google Maniac*

CHAPTER – 2

BACKGROUND WORK

1. **BACKGROUND WORK**
   1. **EXISTING SYSTEM**

In the current scenario it is not possible to locate the friends nearby i.e., in many situations our friends may be present in the same area but we may not be knowing it. Many times we wish to know our friend’s location without calling or messaging them by tracking them on map which is not possible in our existing system.

**2.1.1 DISADVANTAGES OF THE EXISTING SYSTEM**

* We cannot know the friends nearby, we need to call and find out.
* We cannot locate friends on our map.
* We cannot know the location of our contact\friends.
* More wastage of money involves in calling and asking the location of our friend.
* Waste of time.

CHAPTER – 3

PROPOSED SYSTEM

1. **PROPOSED SYSTEM**

Friend location alert application gives a notification whenever our contacts using this app are in nearby location within a particular proximity. It detects proximity between a user and the user's friends. Users have flexible choices of their proximity detection distances i.e. the radius within which friend’s can be located. The user can locate his contacts on a map.

**3.1 ADVANTAGES**

* This app notifies a user if the user is geographically close to any of his friends.
* We can locate friends on map.
* We can know the particular location of the contact when needed.
* Instant calling and messaging option.
* When we are unable to contact our friends then we can know their location.

**3.2 MODULE DESCRIPTION:**

This is a mobile application used to locate friends nearby. It gives an alert when our friend using this app is within particular radius. We can locate friends on map. When we enter a contact number we can get their location. We can also get the caller’s location.

**List of modules:**

1. Profile Management
2. Messaging & Calling
3. Location Tracker
4. **Profile Management**:

In this module user gets registered with the app. The user has an option of editing his profile. He can search his friends and then send request. The user gets a list of pending requests.

1. **Instant messaging and calling:**

When a user gets an alert that a friend is nearby then the user can instantly call. When a user gets an alert that a friend is nearby then the user can instantly message.

1. **Location Tracking:**

The user can locate his friends on map. We can traverse the path. When we enter the number of a contact using this app it gives the location.

* 1. **FEASIBILITY STUDY**

A feasibility study is a high-level capsule version of the entire system analysis and design process. The study begins by classifying the problem definition. Feasibility is to determine if it’s worth doing. Once an acceptance problem is generated, the analyst develops a logical model of the system. A search for alternative is analyzed carefully.

There are 3 parts in feasibility study

* Technical Feasibility
* Operational Feasibility
* Economical Feasibility
  + 1. **TECHNICAL FEASIBILITY:**

Evaluating the technical feasibility is the trickiest part of a feasibility study. This is because, at this point in time, not too many detailed design of the system, making it difficult to access issues like performance, costs on ( on account of kind of technology to be deployed ) etc.

A number of issues have to be considered while doing a technical analysis:

1. **Understand the different technologies involved in the proposed system:**

Before commencing the project, we have to be very clear about what are the Technologies that are to be required for the development of the new system.

1. **Find out whether the organization currently possess the required technologies:**

Is the required technology available with the organization?If so is the capacity sufficient?

For instance –“Will the application work efficiently if the same apk is run on the higher versions of Android?”

* + 1. **OPERATIONAL FEASIBILTY:**

Proposed projects are beneficial only if they can be turned into information systems that will meet the organizations operating requirements. Simply stated, this test of feasibility asks if the system will work when it is developed and installed. Are there major barriers to implementation? Here the questions that will help test the operational feasibility of a project:

* Is there sufficient support for the project from management from users? If the current system is well liked and used to the extent that persons will not be able to see reasons for change, there may be resistance.
* Are the current business methods acceptable to the user? If they are not, Users may welcome a change that will bring about a more operational and useful systems.
* Have the user been involved in the planning and development of the project?

Early involvement reduces the chances of resistance to the system and in general and increases the likelihood of successful project. In the existing system, the new system was considered to be operational feasible.

* + 1. **ECONOMIC FEASIBILITY:**

Economic feasibility attempts to weigh the costs of developing and implementing new system, against the benefits that would accrue from having the new system in place. This feasibility study gives the top management the economic justification for the new system.

A simple economic analysis which gives the actual comparisons of costs and benefits are much more meaningful in this case. In addition, this proves to be a useful point of reference to compare actual costs as the project progresses. There could be various types of intangible benefits on account of automation. These could include increased customer satisfaction, improvement in product quality better decision making timeliness of information, expediting activities, improved accuracy of operations, better documentation and record keeping, faster retrieval of information, better employee morale.

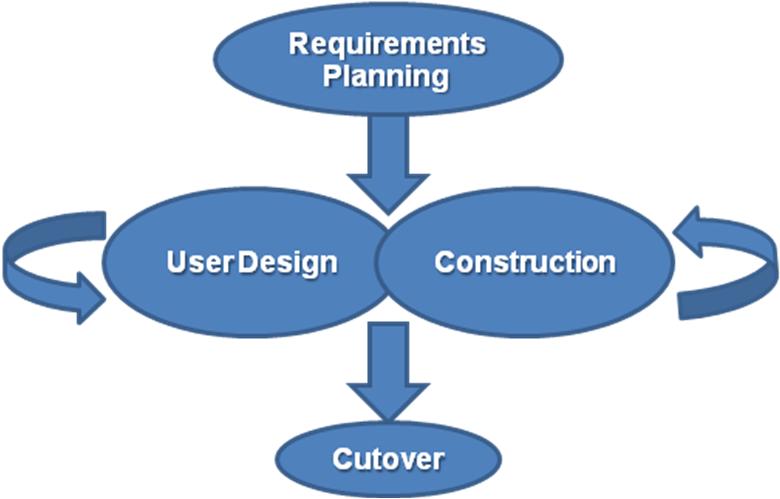
**3.4 SDLC Model:**

# RAPID APPLICATION DEVELOPMENT MODEL (RAD):

Rapid application development (RAD) is a [software development methodology](http://en.wikipedia.org/wiki/Software_development_methodology) that uses minimal planning in favor of rapid prototyping. The "planning" of software developed using RAD is interleaved with writing the software itself. The lack of extensive pre-planning generally allows software to be written much faster, and makes it easier to change requirements.

**Four Phases of RAD:**

1. **Requirements Planning phase** – combines elements of the system planning and systems analysis phases of the [Systems Development Life Cycle](http://en.wikipedia.org/wiki/Systems_Development_Life_Cycle) (SDLC). Users, managers, and IT staff members discuss and agree on business needs, project scope, constraints, and system requirements. It ends when the team agrees on the key issues and obtains management authorization to continue.
2. **User design phase** – during this phase, users interact with systems analysts and develop models and prototypes that represent all system processes, inputs, and outputs. The RAD groups or subgroups typically use a combination of Joint Application Development (JAD) techniques and CASE tools to translate user needs into working models. *User Design* is a continuous interactive process that allows users to understand, modify, and eventually approve a working model of the system that meets their needs.
3. **Construction phase** – focuses on program and application development task similar to the SDLC. In RAD, however, users continue to participate and can still suggest changes or improvements as actual screens or reports are developed. Its tasks are programming and application development, coding, unit-integration and system testing.
4. **Cutover phase** – resembles the final tasks in the SDLC implementation phase, including data conversion, testing, changeover to the new system, and user training. Compared with traditional methods, the entire process is compressed. As a result, the new system is built, delivered, and placed in operation much sooner.

****

**Fig 3.1 :** RAD Model

CHAPTER – 4

SOFTWARE REQUIREMENT SPECIFICATIONS

**4. SOFTWARE REQUIREMENT SPECIFICATIONS**

**4.1 SOFTWARE INTERFACES**

* **Environment:** Android operating system, Eclipse Galileo, Java Development Kit (JDK) 1.6 (or) 1.7
* **Database:** SQLite

**4.2 HARDWARE INTERFACES**

* **Device:** Android Mobile with version Gingerbread (2.2.3) (or) Higher
* **Processor:** Pentium-III (or) Higher
* **RAM:** 1GB (or) Higher
* **Hard Disk:** 40GB (or) Higher

**4.3 COMMUNICATION INTERFACES**

* Not Applicable
  1. **FUNCTIONAL REQUIREMENTS**
* Android Operating System more than Android 2.3.3
* Eclipse Galileo
* Java language
* Java Development Kit (JDK) 1.6
  1. **NON FUNCTIONAL REQUIREMENTS**
* Processor: Intel Pentium or more
* RAM: 256 MB or more
* Hard Disk: 40GB hard disk recommended for primary partition

CHAPTER – 5

LITERATURE SURVEY

**LITERATURE SURVEY**

**5.1 JAVA:**

**5.1.1. Introduction to JAVA**

Initially the language was called as “oak” but it was renamed as “Java” in 1995. The primary motivation of this language was the need for a platform-independent (i.e., architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

**Java is a programmer’s language.**

* Java is cohesive and consistent.
* Except for those constraints imposed by the Internet environment, Java gives the programmer, full control.
* Finally, Java is to Internet programming where C was to system programming**.**

**5.1.2. Features of Java Security**

Every time you that you download a “normal” program, you are risking a viral infection. Prior to Java, most users did not download executable programs frequently, and those who did scan them for viruses prior to execution. Most users still worried about the possibility of infecting their systems with a virus. In addition, another type of malicious program exists that must be guarded against. This type of program can gather private information, such as credit card numbers, bank account balances, and passwords. Java answers both these concerns by providing a “firewall” between a network application and your computer.

**5.1.3. Portability**

For programs to be dynamically downloaded to all the various types of platforms connected to the Internet, some means of generating portable executable code is needed .As you will see, the same mechanism that helps ensure security also helps create portability. Indeed, Java’s solution to these two problems is both elegant and efficient.

**5.1.4. The Byte code**

The key that allows the Java to solve the security and portability problems is that the output of Java compiler is Byte code. Byte code is a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine (JVM). That is, in its standard form, the JVM is an interpreter for byte code.

Translating a Java program into byte code helps makes it much easier to run a program in a wide variety of environments. The reason is, once the run-time package exists for a given system, any Java program can run on it.

Although Java was designed for interpretation, there is technically nothing about Java that prevents on-the-fly compilation of byte code into native code. Sun has just completed its Just In Time (JIT) compiler for byte code. When the JIT compiler is a part of JVM, it compiles byte code into executable code in real time, on a piece-by-piece, demand basis. It is not possible to compile an entire Java program into executable code all at once, because Java performs various run-time checks that can be done only at run time. The JIT compiles code, as it is needed, during execution.

**5.1.5. Java Virtual Machine (JVM)**

Beyond the language, there is the Java virtual machine. The Java virtual machine is an important element of the Java technology. The virtual machine can be embedded within a web browser or an operating system. Once a piece of Java code is loaded onto a machine, it is verified. As part of the loading process, a class loader is invoked and does byte code verification makes sure that the code that’s has been generated by the compiler will not corrupt the machine that it’s loaded on. Byte code verification takes place at the end of the compilation process to make sure that is all accurate and correct. So byte code verification is integral to the compiling and executing of Java code.

**5.1.6. Java Architecture**

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

**5.1.7. Compilation of code**

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed to execute the byte code. The JVM is created for overcoming the issue of portability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be a Intel Pentium Windows 95 or SunSARC station running Solaris or Apple Macintosh running system and all could receive code from any computer through Internet and run the Applets.

**5.1.8. Object-Oriented**

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank slate. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

**5.1.9. Robust**

The multi-platform environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and run time.

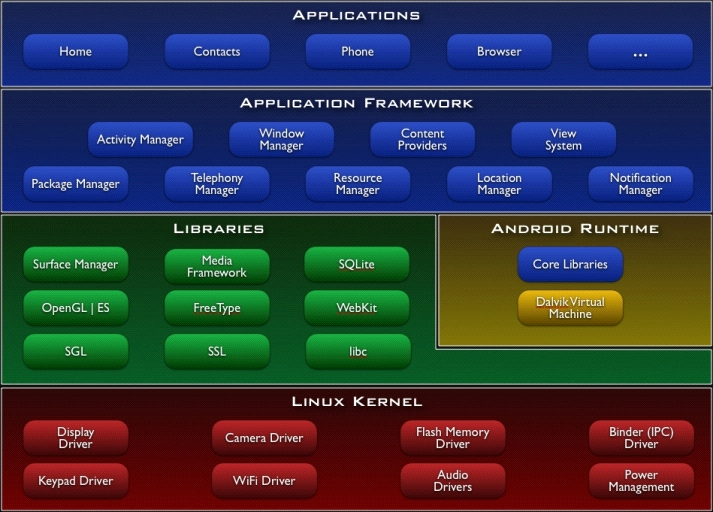
Java virtually eliminates the problems of memory management and de-allocation, which is completely automatic. In a well-written Java program, all run time errors can –and should –be managed by your program.

**5.2 ECLIPSE GALILEO (3.5)**

**Eclipse** is a commercially available Java EE and AJAX IDE created and maintained by the company Genuitec, a founding member of the Eclipse Foundation. **Eclipse** is a multi-language [software development environment](http://en.wikipedia.org/wiki/Software_development_environment) comprising an [integrated development environment](http://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) and an extensible [plug-in](http://en.wikipedia.org/wiki/Plug-in_(computing)) system. It is written mostly in [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) and can be used to develop applications in Java. Eclipse employs plug-ins in order to provide all of its functionality on top of (and including) the runtime system, in contrast to some other applications where functionality is typically [hard coded](http://en.wikipedia.org/wiki/Hard_code). The runtime system of Eclipse is based on [Equinox](http://en.wikipedia.org/wiki/Equinox_(OSGi)), an [OSGi](http://en.wikipedia.org/wiki/OSGi) standard compliant implementation. This plug-in mechanism is a lightweight [software componentry](http://en.wikipedia.org/wiki/Software_componentry) framework. In addition to allowing Eclipse to be extended using other [programming languages](http://en.wikipedia.org/wiki/Programming_language) such as [C](http://en.wikipedia.org/wiki/C_(programming_language)) and [Python](http://en.wikipedia.org/wiki/Python_(programming_language)), the plug-in framework allows Eclipse to work with typesetting languages like [LaTeX](http://en.wikipedia.org/wiki/LaTeX), networking applications such as [telnet](http://en.wikipedia.org/wiki/Telnet), and [database management systems](http://en.wikipedia.org/wiki/Database_management_system).

**5.3 ANDROID:**

Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.Google Inc. purchased the initial developer of the software, Android Inc., in 2005. Android's mobile operating system is based upon a modified version of the Linux kernel.

**5.3.1 Android Architecture:**

**Fig 5.1:** Android Architecture

**5.3.1.1 Applications**

Android will ship with a set of core applications including an email client, SMS program, calendar, maps, browser, contacts, and others. All applications are written using the Java programming language.

**5.3.1.2 Application Framework**

By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including:

* A rich and extensible set of Views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser
* Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data
* A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files
* A Notification Manager that enables all applications to display custom alerts in the status bar
* An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack

**5.3.1.3 Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

* System C library - a BSD-derived implementation of the standard C system library (libc), tuned for embedded Linux-based devices
* Media Libraries - based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG, and PNG
* Surface Manager - manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications
* LibWebCore - a modern web browser engine which powers both the Android browser and an embeddable web view
* SGL - the underlying 2D graphics engine
* 3D libraries - an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
* FreeType - bitmap and vector font rendering

**5.3.1.4 Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language.

Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool.

The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management

**5.3.1.5 Linux Kernel**

Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

**5.4 ORACLE 10G DATABASE SERVER**

Oracle Database 10g is Oracle’s revolutionary new grid offering for database administrators, which include breakthrough self-management, storage management, and clustering features. It is designed to relieve system administrators from repetitive, low-value administrative work and provide them with an opportunity to increase their value and expand their technical skills. The new Oracle University curriculum tracks for Oracle Database 10g are tailored to the scale and needs of customers’ data centers. These new courses target current and new Oracle Database Administrators and provide training in half the time and half the cost.

**5.4.1 FEATURES**

There are too many Oracle 10g SQL new features to be able to cover them all here so we've just picked out a few of the more interesting ones as well as a few SQL\*Plus new features.

In brief, the SQL new features and enhancements we'll be looking at are:

* [writeable external tables](http://www.asktheoracle.net/oracle-10g-sql-new-features.html#external_table)
* [restore points and flashback queries](http://www.asktheoracle.net/oracle-10g-sql-new-features.html#Restore_Points_and_Flashback_Query)
* [regular expression support](http://www.asktheoracle.net/oracle-10g-sql-new-features.html#Regular_Expression_Support)

The Oracle 10g new features can be organized into 5 broad categories:

1. Manageability/automation - in other words decreasing the amount of time spent on routine tasks to leave more time for strategic tasks

* Common Manageability Infrastructure
* Storage Enhancements
* Space Management
* Other Manageability Enhancements
* Human Error Correction (including [flashback database](http://www.asktheoracle.net/oracle-10g-new-features-flashback-db.html))

1. Availability

* Recovery through reset logs
* Backup and recovery
* Data Guard
* Online redefinition
* Log Miner

1. Performance

* Resource Manager
* Wait interface enhancements
* End to end application testing
* SQL tuning advisor

1. Business Intelligence/Data Warehousing

* Materialized view refresh and query rewrite
* Asynchronous change data capture
* Materialized view refresh and query rewrite

1. Application Development

* [SQL and SQL\*PLUS](http://www.asktheoracle.net/oracle-10g-sql-new-features.html)
* [PLSQL](http://www.asktheoracle.net/oracle-10g-plsql-new-features.html)

**5.5 DATAMNIC DEZIGN DATABASE:**

DeZign for Databases is an intuitive database design and modeling tool for developers and DBA's that can help you model, create and maintain databases. The software uses entity relationship diagrams (ERDs, data models) to graphically design databases and automatically generates the most popular SQL and desktop databases.

DeZign for Databases offers a sophisticated visual data modeling environment for database application development that makes your database development process much easier. The tool reduces faults in database development and improves your productivity. You can visualize database structures to understand your database, create new databases or reverse-engineer existing databases to modify, document, analyze, and optimize.

The tool is extremely easy to use. Whether you are a beginner or an expert database modeler, you will find your way in the tool very easily. DeZign for Databases provides all the features you expect in a professional database modeling and design tool.

**5.6 APACHE TOMCAT 6.0:**

**Apache Tomcat** (or **Jakarta Tomcat** or simply **Tomcat**) is an [open source](http://en.wikipedia.org/wiki/Open_source) [servlet container](http://en.wikipedia.org/wiki/Java_Servlet) developed by the [Apache Software Foundation](http://en.wikipedia.org/wiki/Apache_Software_Foundation) (ASF). Tomcat implements the [Java Servlet](http://en.wikipedia.org/wiki/Java_Servlet) and the [JavaServerPages](http://en.wikipedia.org/wiki/JavaServer_Pages) (JSP) specifications from [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems), and provides a "pure [Java](http://en.wikipedia.org/wiki/Java_(programming_language))" [HTTP](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [web server](http://en.wikipedia.org/wiki/Web_server) environment for [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) code to run.

**Tomcat 5.x has the following features:**

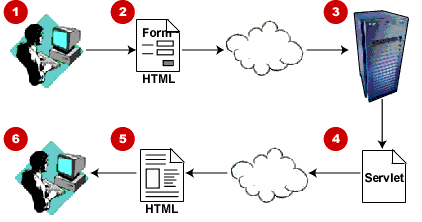
* Implements the Servlet 2.4 and JSP 2.0 specifications
* Reduced garbage collection, improved performance and scalability
* Native Windows and Unix wrappers for platform integration
* Faster JSP parsing

**5.6.1 Servlet:**

Servlet is what you get when you cross an applet with a CGI script. From a programmatic point of view, a servlet resembles an applet; it is an executable that's written in the Java programming language, and it usually (though not always) is executed in response to an invocation from an HTML page.

Figure shows one of the most common ways of using a servlet.

A user (1) requests some information by filling out a form containing a link to a servlet and clicking the Submit button (2). The server (3) locates the requested servlet (4). The servlet then gathers the information needed to satisfy the user's request and constructs a Web page (5) containing the information. That Web page is then displayed on the user's browser (6).



**Fig 5.2:** Servlet

**5.7 JDBC**

JDBC stands for "Java Data Base Connectivity". It is an API (Application Programming Interface) which consists of a set of Java classes, interfaces and exceptions and a specification to which both JDBC driver vendors and JDBC developers adhere when developing applications.

JDBC is a very popular data access standard. RDBMS (Relational Database Management Systems) or third-party vendors develop drivers which adhere to the JDBC specification. Other developers use these drivers to develop applications which access those databases e.g. you'll use Connector J JDBC driver to access My SQL database. Since the drivers adhered to JDBC specification, the JDBC application developers can replace one driver for their application with another better one without having to rewrite their application. If they had used some proprietary API provided by some RDBMS vendor, they will not have been able to change the driver and/or database without having to rewrite the complete application.

**5.7.1 Advantages of JDBC Technology**

* **Leverage Existing Enterprise Data-**With JDBC technology, businesses are not locked in any proprietary architecture, and can continue to use their installed databases and access information easily -- even if it is stored on different database management systems.
* **Simplified Enterprise Development**-The combination of the Java API and the JDBC API makes application development easy and economical. JDBC hides the complexity of many data access tasks, doing most of the "heavy lifting” for the programmer behind the scenes. The JDBC API is simple to learn, easy to deploy, and inexpensive to maintain.
* **Zero Configuration for Network Computers**-With the JDBC API, no configuration is required on the client side. With a driver written in the Java programming language, all the information needed to make a connection is completely defined by the JDBC URL or by a Data Source object registered with a Java Naming and Directory Interface (JNDI) naming service. Zero configuration for clients supports the network computing paradigm and centralizes software maintenance.

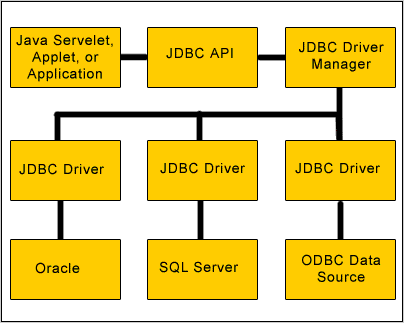
**5.7.2 Key Features**

* **Full Access to Metadata-**The JDBC API provides metadata access that enables the development of sophisticated applications that need to understand the underlying facilities and capabilities of a specific database connection.
* **No Installation** -A pure JDBC technology-based driver does not require special installation; it is automatically downloaded as part of the applet that makes the JDBC calls.
* **Database Connection Identified by URL**- JDBC technology exploits the advantages of Internet-standard URLs to identify database connections. The JDBC API includes an even better way to identify and connect to a data source, using a Data Source object that makes code even more portable and easier to maintain. In addition to this important advantage, Data Source objects can provide connection pooling and distributed transactions, essential for enterprise database computing. This functionality is provided transparently to the programmer.
* **Included in the Java Platform*-***As a core part of the Java 2 Platform, the JDBC API is available anywhere that the platform is. This means that your applications can truly write database applications once and access data anywhere. The JDBC API is included in both the Java 2 Platform, Standard Edition (J2SE) and the Java 2 Platform, Enterprise Edition (J2EE), providing server-side functionality for industrial strength scalability.

**5.7.3 JDBC has four Components:**

1. The JDBC API.  
2. The JDBC Driver Manager.  
3. The JDBC Test Suite.  
4. The JDBC-ODBC Bridge.

A **JDBC** driver translates standard JDBC calls into a network or database protocol or into a database library API call that facilitates communication with database. This translation layer provides JDBC applications with database independence. If back-end database changes, only JDBC driver need be replaced with few code modifications required.

****

**Fig 5.3 :** JDBC Architecture

* 1. **SQLITE:**

**SQLite** is a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) contained in a [C](http://en.wikipedia.org/wiki/C_(programming_language)) programming [library](http://en.wikipedia.org/wiki/Library_(computer_science)). In contrast to other database management systems, SQLite is not a separate process that is accessed from the client application, but an integral part of it.

SQLite is [ACID](http://en.wikipedia.org/wiki/Atomicity,_consistency,_isolation,_durability)-compliant and implements most of the [SQL](http://en.wikipedia.org/wiki/SQL) standard, using a dynamically and weakly typed SQL [syntax](http://en.wikipedia.org/wiki/Syntax) that does not guarantee the [domain integrity](http://en.wikipedia.org/wiki/Integrity_constraints).

SQLite is a popular choice as [embedded database](http://en.wikipedia.org/wiki/Embedded_database) for local/client storage in [application software](http://en.wikipedia.org/wiki/Application_software) such as [web browsers](http://en.wikipedia.org/wiki/Web_browser). It is arguably the most widely deployed [database engine](http://en.wikipedia.org/wiki/Database_engine), as it is used today by several widespread browsers, [operating systems](http://en.wikipedia.org/wiki/Operating_system), and [embedded systems](http://en.wikipedia.org/wiki/Embedded_system), among others. SQLite has many [bindings](http://en.wikipedia.org/wiki/Language_binding) to programming languages.

The [source code](http://en.wikipedia.org/wiki/Source_code) for SQLite is in the [public domain](http://en.wikipedia.org/wiki/Public_domain).

**5.8.1. Design:**

Unlike [client–server](http://en.wikipedia.org/wiki/Client%E2%80%93server) database management systems, the SQLite engine has no standalone [processes](http://en.wikipedia.org/wiki/Process_(computing)) with which the application program communicates. Instead, the SQLite [library](http://en.wikipedia.org/wiki/Library_(computing)) is [linked](http://en.wikipedia.org/wiki/Linker_(computing)) in and thus becomes an integral part of the application program. (In this, SQLite follows the precedent of [Informix SE](http://en.wikipedia.org/wiki/IBM_Informix#Other_Products) of [c. 1984](http://www.iiug.org/faqs/informix-faq/ifaq01.htm.1#1.2)) The library can also be called dynamically. The application program uses SQLite's functionality through simple [function calls](http://en.wikipedia.org/wiki/Subroutine), which reduce [latency](http://en.wikipedia.org/wiki/Latency_(engineering)) in database access: function calls within a single process are more efficient than [inter-process communication](http://en.wikipedia.org/wiki/Inter-process_communication). SQLite stores the entire database (definitions, tables, indices, and the data itself) as a single cross-platform [file](http://en.wikipedia.org/wiki/Computer_file) on a host machine. It implements this simple design by [locking](http://en.wikipedia.org/wiki/Lock_(computer_science)) the entire database file during writing. SQLite read operations can be multitasked, though writes can only be performed sequentially.

**5.8.2. Features:**

SQLite implements most of the [SQL-92](http://en.wikipedia.org/wiki/SQL-92) standard for [SQL](http://en.wikipedia.org/wiki/SQL) but it lacks some features. For example it has partial support for [triggers](http://en.wikipedia.org/wiki/Database_trigger), and it can't write to [views](http://en.wikipedia.org/wiki/View_(database)) (however it supports INSTEAD OF triggers that provide this functionality). While it supports complex queries, it still has limited [ALTER TABLE](http://en.wikipedia.org/wiki/Data_Definition_Language#ALTER_statements) support, as it can't modify or delete columns.

SQLite uses an unusual [type system](http://en.wikipedia.org/wiki/Type_system) for an SQL-compatible DBMS; instead of assigning a type to a column as in most SQL database systems, types are assigned to individual values; in language terms it is dynamically typed. Moreover, it is weakly typed in some of the same ways that [Perl](http://en.wikipedia.org/wiki/Perl) is: one can insert a [string](http://en.wikipedia.org/wiki/String_(computer_science)) into an [integer](http://en.wikipedia.org/wiki/Integer) column (although SQLite will try to convert the string to an integer first, if the column’s preferred type is integer). This adds flexibility to columns, especially when bound to a dynamically typed scripting language. However, the technique is not portable to other SQL products. A common criticism is that SQLite's type system lacks the data integrity mechanism provided by statically typed columns in other products. The SQLite web site describes a "strict affinity" mode, but this feature has not yet been added.

Several [computer processes](http://en.wikipedia.org/wiki/Computer_process) or [threads](http://en.wikipedia.org/wiki/Thread_(computer_science)) may access the same database concurrently. Several read accesses can be satisfied in parallel. A write access can only be satisfied if no other accesses are currently being serviced. Otherwise, the write access fails with an [error code](http://en.wikipedia.org/wiki/Error_code) (or can automatically be retried until a configurable timeout expires). This concurrent access situation would change when dealing with temporary tables. This restriction is relaxed in version 3.7 when [WAL](http://en.wikipedia.org/wiki/Write-ahead_logging) is turned on enabling concurrent reads and writes.

A standalone program called sqlite3 is provided that can be used to create a database, define tables within it, insert and change rows, run queries and manage an SQLite database file. This program is a single executable file on the host machine. It also serves as an example for writing applications that use the SQLite library. SQLite full [Unicode](http://en.wikipedia.org/wiki/Unicode) support is optional. SQLite also has [bindings](http://en.wikipedia.org/wiki/Language_binding) for a large number of [programming languages](http://en.wikipedia.org/wiki/Programming_language), including [BASIC](http://en.wikipedia.org/wiki/BASIC), [C](http://en.wikipedia.org/wiki/C_(programming_language)), [C++](http://en.wikipedia.org/wiki/C%2B%2B), etc.

CHAPTER – 6

SYSTEM DESIGN

1. **SYSTEM DESIGN**
   1. **Introduction:**

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer’s goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

**6.2 Architecture:**

The three-tier architecture was comes into existence to improve management of code and contents and to improve the performance of the web based applications. There are mainly three layers in three-tier architecture.

These are defined as follows:

1. Presentation.
2. Business Logic.
3. Database.

1.  **Presentation:** First layer Presentation contains mainly the interface code, and this is shown to user. This code could contain any technology that can be used on the client side like HTML, JavaScript or VBScript etc.

2. **Business Logic** : Second layer is Business Logic which contains all the code of the server-side .This layer have code to interact with database and to query, manipulate, pass data to user interface and handle any input from the UI as well.

3. **Database** : Third layer Data represents the data store like MS Access, SQL Server, an XML file, an Excel file or even a text file containing data also some additional database are also added to that layers.

**Database**

**Presentation Layer**

**Business Logic**

**Fig 6.1:** Architecture

**6.3 Entity-Relationship diagrams:**

Basic objects in the logical model are:

* Entities
* Attributes
* Relationships

**Entity:** An entity is a logical object that represents a person, place or thing about which an organization maintains information

**Attributes:** Attributes collect information about an entity. Logical attributes usually correspond to physical columns in a table.

**Relationships:** A very important object in a data model is the relationship, which is represented by the solid or dashed line that connects two entities or two tables. A relationship line connects a parent and child entity or a table.

Basic objects in the physical model are:

**Table**: An entity in the logical model usually corresponds to a table in a physical data model. In the physical model, a graphic box represents a table in which data is stored in the database.

**Columns:** Columns collect information about a table.

**Views:** In a physical model, you can create a view, which is really a SQL query that is permanently stored in the database. Typically, a view is used to present specific database information for target audience.

**User:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| Name |  | VARCHAR2 |
| Id | NOT NULL | VARCHAR2 |
| Password |  | VARCHAR2 |
| date\_of\_birth |  | INTEGER |
| mobile\_number |  | INTEGER |
| Latitude |  | VARCHAR2 |
| Longitude |  | VARCHAR2 |
| security\_ans |  | VARCHAR2 |
| ques\_id |  | VARCHAR2 |
| Radius |  | NUMBER |

**Security\_ques:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| ques\_id | NOT NULL | VARCHAR2 |
| Question |  | VARCHAR2 |

**Friend\_list:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| Id | NOT NULL | VARCHAR2 |
| friend\_id | NOT NULL | VARCHAR2 |
| Status |  | VARCHAR2 |

**6.4 UML Design:**

**Introduction:**

Modeling is an activity that has been carried out over the years in software development. When writing applications by using the simplest languages to the most powerful and complex languages, you still need to model. Modeling can be as straightforward as drawing a flowchart listing the steps carried out by an application.

**Why do we use modeling?**

Defining a model makes it easier to break up a complex or a huge system into simple, discrete pieces that can be individually studied. We can focus more easily on the smaller parts of a system and then understand the “big picture”. Hence, the reasons behind modeling can be summed up in two words:

* Readability
* Reusability

**Readability** brings clarity- ease of understanding. Understanding a system is the first step in either building or enhancing a system. This involves knowing what a system is made up of, how it behaves, and so forth. Modeling a system ensures system to make it readable involves capturing the structure of a system and the behavior of the system.

**Reusability** is the byproduct of making the system readable. After a system has been modeled to make it easy to understand, we tend to identify similarities or redundancy, be they in terms of functionality, features, or structure. The Unified Modeling Language or UML, as it is popularly known by its TLA (Three Letter Acronym), is the language that can be used to model systems and make them readable. This essentially means that UML provides the ability to capture the characteristics of a system by using notations. UML provides a wide array of simple, easy to understand notations for documenting systems based on the object-oriented design principles. These notations are called the nine diagrams of UML.

Different languages have been used for depicting systems using object-oriented methodology. The prominent among these were the Rumbaing methodology, the Brooch methodology and the Jacobson methodology. The problem was that, although each methodology had its advantages, they were essentially disparate. Hence, if you had to work on different projects that use any of these methodologies, you had to be well versed with each of these methodologies. A very tall order indeed! The Unified Modeling Language is just that. It “unifies” the design principles of each of these methodologies into a single, standard, language that can be easily applied across the board for all object-oriented systems. But, unlike the different methodologies that tended more to design and detailed design system of system, UML spans the realm of requirement, analysis, and design and uniquely implementation as well. The beauty of UML lies in the fact that any of the nine diagrams can be used on an incremental basis as need arises. Considering all these reasons, it is no wonder that UML is considered “the” language of choice.

UML does not have any dependencies with respect to any technologies or languages. That implies that you can use UML to model applications and systems based on either of the current hot technologies; for example J2EE and .NET. Every effort has been made to keep UML as a clear and concise modeling language without being tied down to any technologies.

**6.4.1 Introduction to UML:**

The Unified Modeling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object-oriented software and the software development projects. Using the UML helps projects teams communicate, explore potential designs, and validate the architectural design of the software.

**Goals of UML:**

The primary goals in the design of the UML were:

* Provides users with a ready-to-use, expensive visual modeling language so they can develop and exchange meaningful models.
* Provide extensibility and specialization mechanisms to extend the core concepts.
* Be independent of particular programming languages and development processes.
* Provide a formal basis for understanding the modeling language.
* Encouraging the growth of object oriented tools market.
* Support higher-level development concepts such as collaborations, frameworks, patterns and components.
* Ingrate best practices.

**Why do we use UML?**

As the strategic value of software increases for many companies, the industry looks for techniques to automate the production of software and to improve quality and reduce cost and time-to-market. These techniques include component technology, visual programing, patterns, and frameworks. Businesses also seek techniques to manage the complexity of systems as they increase in scope and scale. In particular, they recognize the need to solve recurring architectural problems, such as physical distribution, concurrency, replication, security, load balancing and fault tolerance. Additionally, the development for the World Wide Web, while making some things simpler, has exhausted these architectural problems. The Unified Modeling Language (UML) was designed to respond to these needs.

**UML Diagrams:**

The underlying premise of UML is that no one diagram can capture the different elements of a system in its entirety. Hence, UML is made up of nine diagrams that can be used to model a system at different points of time in the software lifecycle of a system.

The nine UML diagrams are:

1. **Use Case Diagram:** The use case diagram is used to identify the primary elements and processes that form the system. The primary elements are termed as “actors” and the processes are called “use cases.” The use case diagram shows which actors interact with each use case.
2. **Class Diagram:** The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an “is-a” or “has-a” relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed “methods” of the class. Apart from this, each class may have certain “attributes” that uniquely identify the class.
3. **Object Diagram:** The object diagram is a special kind of class diagram. An object is the instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point of time.
4. **State Diagram:** A State diagram, as the name given, represents the different states that objects in the system undergo during their life cycle. Objects in the system change states in response to events. In addition to this, a state diagram also captures the transition of the object’s state from an initial state to a final state in response to events affecting the system.
5. **Activity Diagram:** The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.
6. **Sequence Diagram:** A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing “message”.
7. **Collaboration Diagram:** A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.
8. **Component Diagram:** The component diagram represents the high-level parts that make up the system. This diagram depicts, at a high level, what components form part of the system and how they are interrelated. A component diagram depicts the components called after the system has undergone the development or construction phase.
9. **Deployment Diagram:** The deployment diagram captures the configuration of the runtime elements of the application. This diagram is by far most useful when a system is built and ready to be deployed.

Now that we have an idea of the different UML diagrams, let us see if we can somehow group together these diagrams to enable us to further understand how to use them.

**UML Diagram Classification-Static, Dynamic, and Implementation:**

A software system can be said to have two distinct characteristics: a structural” static” part and a behavioral, “dynamic” part. In addition to these two characteristics, an additional characteristic that a software system possesses is related to implementation. Before we categorize UML diagrams into each of these three characteristics, let us take a quick look at exactly what these characteristics are:

1. **Static:** The static characteristics of a system are essentially the structural aspect of the system. The static characteristics define what parts the system is made up of.
2. **Dynamic:** The behavioral features of a system; for example, the ways a system behaves in response to certain events or actions are the dynamic characteristics of a system.
3. **Implementation:** The implementation characteristics of a system is entirely new feature that describes the different elements required for deploying a system.

The UML diagrams that fall under each of these categories are:

1. **Static:**

* Use Case Diagram
* Class diagram

1. **Dynamic:**

* Object diagram
* State diagram
* Activity diagram
* Sequence diagram
* Collaboration diagram

1. Implementation

* Component diagram
* Deployment diagram

Finally, let us take a look at the 4+1 view of UML diagrams.

**Views of UML Diagrams:**

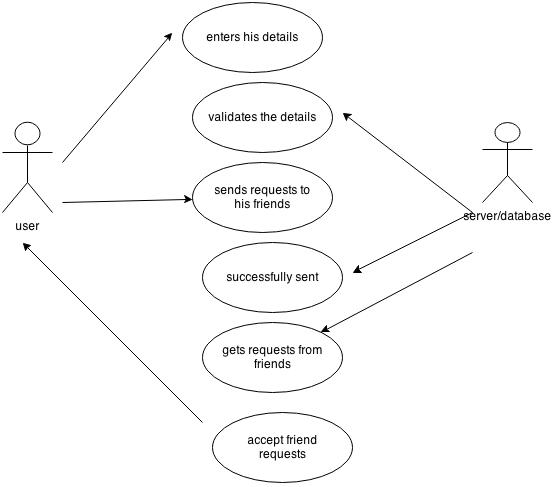
Considering that the UML diagrams can be used in different stages in the life cycle of a system, let us take a look at the “4+1 view” of UML diagrams. The 4+1 view offers a different perspective to classify and apply UML diagrams. The 4+1 view is essentially how a system can be modeled. This will enable us to understand where exactly the UML diagrams fit in and their applicability.

The different views are:

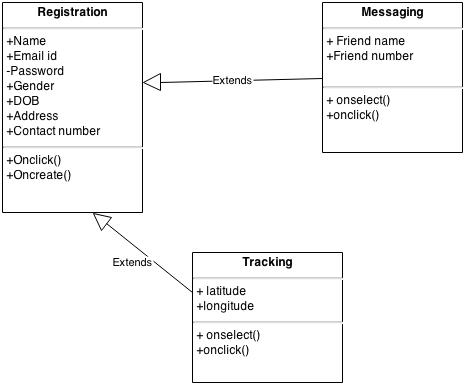
* **Design View:** The design view of a system is the structural view of the system. This gives an idea of what a given view of the system.
* **Process View:** The dynamic behavior of a system can be seen using the process view. The different diagrams such as the state diagram, activity diagram, sequence diagram, and collaboration diagram are used in this view.
* **Component View:** Component view shows the grouped modules of a given system modeled using the component diagram.
* **Deployment View:** The deployment diagram of UML is used to identify the deployment modules for a given system.
* **Use Case View:** Finally, we have the use case view. Use case diagrams of UML are used to view a system from this perspective as a set of discrete activities or transactions.

**UML DIAGRAMS:**

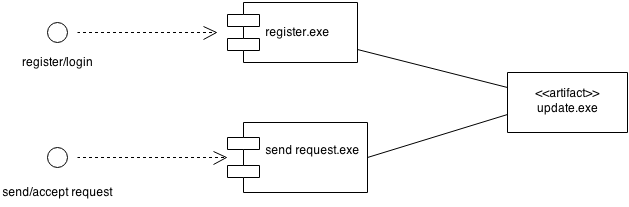
1. **Use Case Diagram:**
2. **Profile management Usecase diagram:**

****

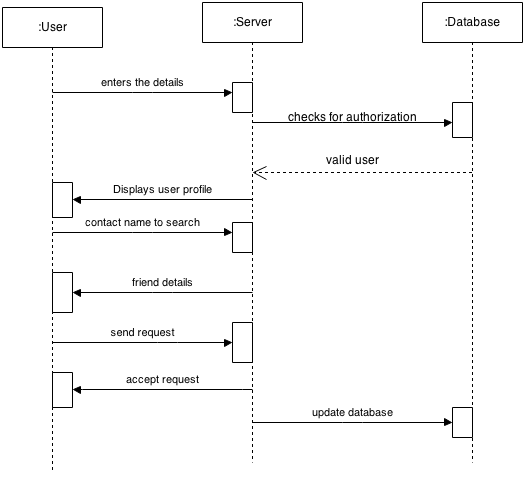
1. **Class Diagram:**

****

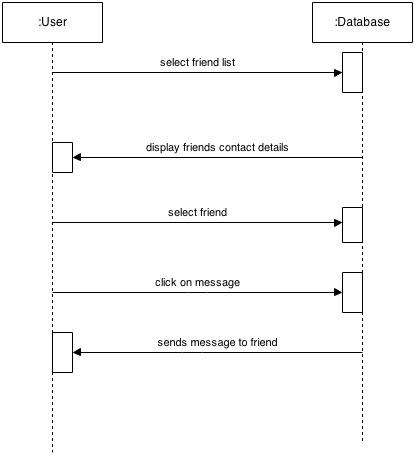
1. **Component Diagram:**

****

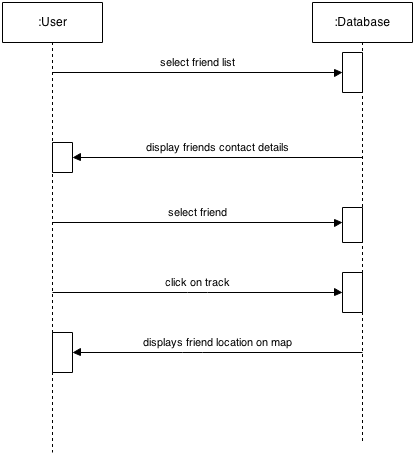
1. **Sequence Diagram:**
2. **Profile management sequence diagram:**

****

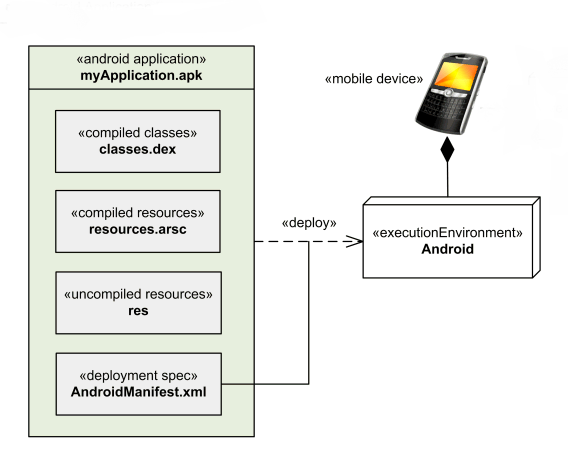
1. **Messaging sequence diagram:**

****

1. **Tracking sequence diagram:**

****

1. **Deployment Diagram:**

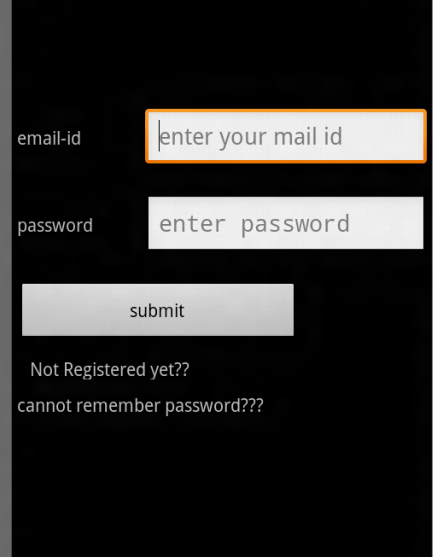
****

CHAPTER – 7

USER INTERFACES OR OUTPUT SCREENS

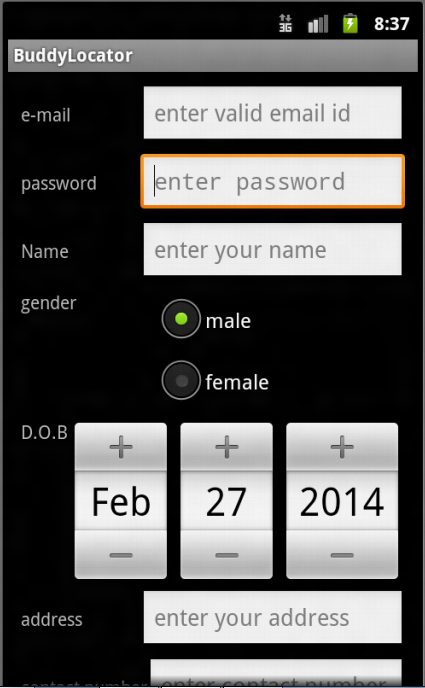
1. **User Interfaces or Output Screens**

**7.1 Login Page:**

****

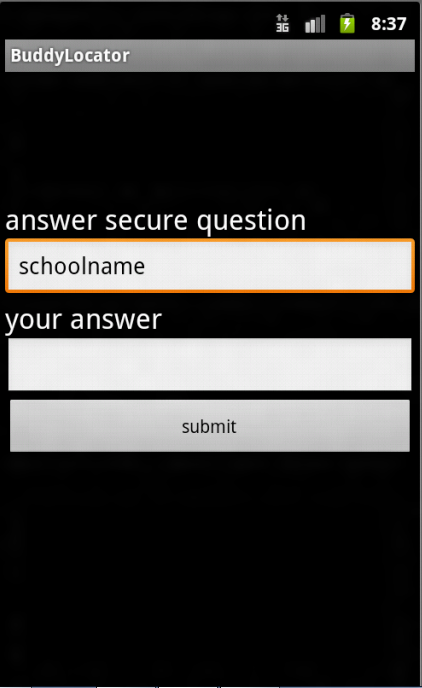
This is the welcome page. If the user is an existing one then he can login with his mail id and password. If the user is new he can click on not yet register to register with his details.

**7.2 Registration Page**

****

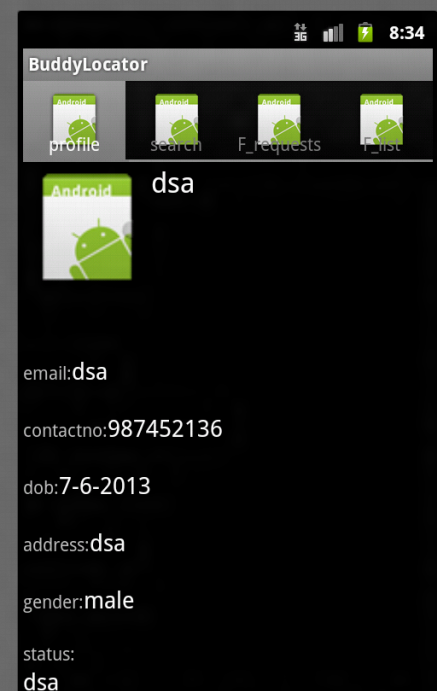
In this registration page, user enters all his details like mail id, gender, date of birth etc and gets registered with the app.

**7.3 Forgot Password Page**

****

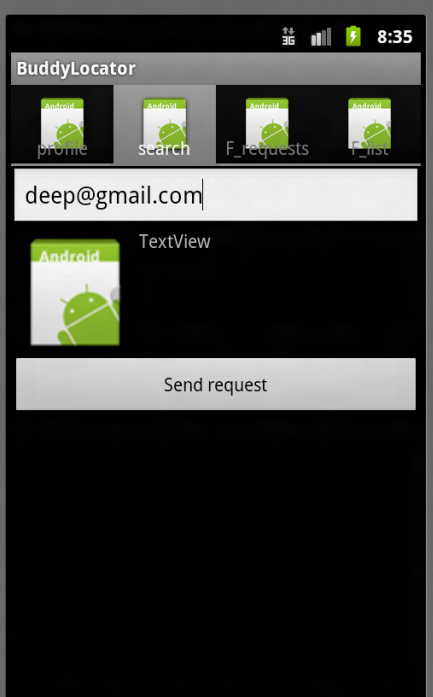
If the user forgets the password, he can click on forgot password button. This will redirect to forgot password screen. The user gets the password once he answers the security question.

**7.4 Profile Page**

****

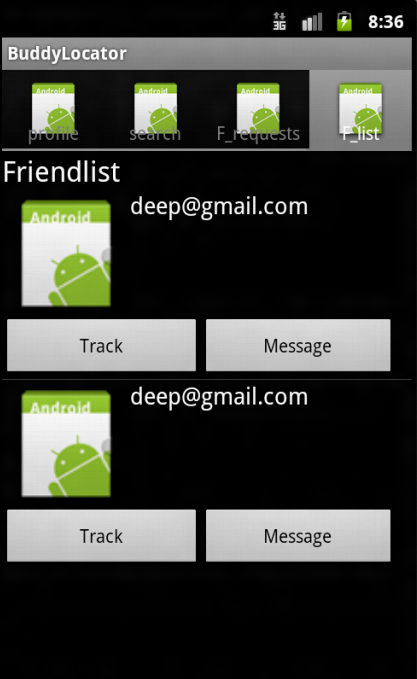
This screen displays the profile of the user which includes mail id, date of birth, status etc.

**7.5 Search Page**

****

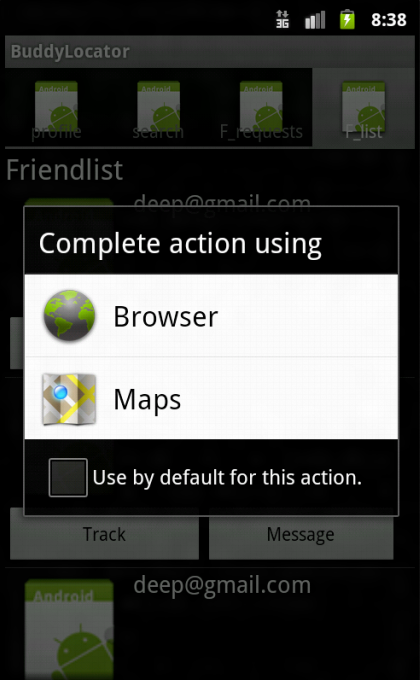
In search page, the user can search for his friends using this application and can send them request.

**7.6 Friend List Page**

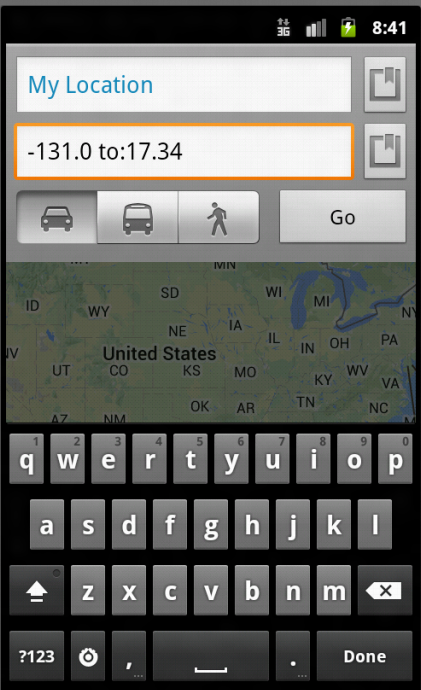
****

Friendlist page displays the list of user’s friends. Along with their name there are two options called track and message. The user can track and message them.

**7.7 Tracking Page**

****

Once the user clicks on track button this page is displayed. This page displays options through which the user can track his friend. The user can select anyone of the option.

****

This page displays the location of the friends. The user can choose his mode of travel to that particular location and calculate the time to reach that particular location.

CHAPTER – 8

TESTING

1. **TESTING**

**8.1 Introduction**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation.

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

* meets the requirements that guided its design and development,
* works as expected,
* can be implemented with the same characteristics,
* and satisfies the needs of stakeholders.

**Levels of Testing:**

In order to uncover the errors present in the different phases we have the concept of levels of testing. The basic levels of testing are as shown below:

Client Needs

Acceptance Testing

Requirements

System Testing

Design

Integration Testing

Unit Testing

Code

**Fig 8.1:** Levels of Testing

**Testing Methods**

* Unit Testing
* Integration Testing
* Validation Testing
* White Box Testing
* Black Box Testing

**8.2 Unit Testing**

Unit testing is a software verification and validation method in which a programmer tests if individual units of 0 are fit for use. A unit is the smallest testable part of an application. In procedural programming a unit may be an individual function or procedure. Ideally, each test case is independent from the others: substitutes like method stubs, objects, fakes and test harnesses can be used to assist testing a module in isolation.

**8.3 Integration Testing**

Integration testing is sometimes called Integration and Testing. Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before system testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates and delivers as its output the integrated system ready for system testing.

**8.4 Validation Testing**

Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software function in a manner that can reasonably expected by a customer. After validation test has been conducted, one of the following two possible conditions exists. The functions or performance characteristics confirm to specification and are accepted.

* In the user module, all the fields must be filled.
* In the user module, phone number must be numeric.
* Confirm password and the given password must match.

**8.5 White Box Testing**

White box sometimes called “Glass Box Testing” is a test case design uses the control structure of the procedural design to drive the test case. Using white box testing methods, the following tests were made on the system.

* All independent paths within a module have been exercised once. In our system, ensuring that case was selected and executed checked all case structures. The bugs that were prevailing in some part of the code where fixed.
* All logical decisions were checked for the truth and falsity of the values.

**8.6 Black Box Testing**

Black box testing focuses on the functional requirements of the software. This is black box testing enables the software engineering to derive a set of input conditions that will fully exercise all functional requirements for a program. Black box testing is not an alternative to white box testing rather it is complementary approach that is likely to uncover o different class of errors that white box methods like.

* Interface errors
* Performance in data structures
* Performance errors
* Initializing and termination errors

**8.7 Test Cases**

Test cases are designed for finding undiscovered error. All the modules are tested against the specification produced during the design for the modules. These modules were tested as a part of unit testing. Test cases are also able to identify inter module impacts.

**Registration Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| Should allow the user to enter his details and click submit button | It redirects the user to the login screen of the application. | It redirects user to the login screen of the application. | Success | The button allows the user to create new account |
| If user doesn’t enter details in any one of the field and click submit button | It gives an alert saying all fields are mandatory. | It gives an alert saying all fields are mandatory. | Success | All the fields are mandatory. |

**Login Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| User enters valid id and password. | It redirects the user to the profile screen of the application. | It redirects user to the profile screen of the application. | Success | The button allows the user to login into his account. |
| If user enters invalid details in any one of the field and click submit button | It gives an alert saying invalid id or password. | It gives an alert saying invalid id or password. | Success | All the fields are mandatory. |
| If user enters id and clicks on forgot password. | It redirects to forgot password page | It redirects to forgot password page | Success |  |

**Search Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| Should allow the user to search for his friends and then click send request | It gives an alert saying friend request sent. | It gives an alert saying friend request sent. | Success | The button allows the user to search for other friends. |

**Friend Request Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| It allows the user to accept his friends. | Once the request is accepted, it displays the friend name in friends list. | Once the request is accepted, it displays the friend name in friends list. | Success | The button allows the user to add more friends. |

**Friend List Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| It displays the user’s friends list. | It displays the track and message options beside the friend name. | It displays the track and message options beside the friend name. | Success | The button allows the user to track or message his friend. |

**Track Page**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FUNCTION** | **EXPECTED RESULTS** | **ACTUAL RESULTS** | **REMARKS** | **COMMENTS** |
| It should display the location of the friend. | The location of the friend is shown on a map. | The location of the friend is shown on a map. | Success | The button allows the user to track his other friends. |
| If the GPS is off friend’s location cannot be tracked. | It gives an alert saying to turn on GPS. | It gives an alert saying to turn on GPS. | Success | GPS should be turned on to track friends on map. |

CHAPTER – 10

CONCLUSIONS AND FUTURE ENHANCEMENT

1. **CONCLUSIONS AND FUTURE ENHANCEMENTS**

**CONCLUSION**

Friend location alert is a mobile application which is used to give an alert when our contacts using this app are in nearby location. The user can instantly call or message them. Using this application, we can locate our friends on the map i.e., we can keep a track on our near and dear ones especially during the times of distress. Hence this feature makes it different from other products in the market to be unique and distinguishable.

**FUTURE ENHANCEMENTS**

Inthe present version of Advanced Profile Manager Application, the messages being sent are charged from the users’ main balance or internal balance. In the near future this feature can be enhanced to a new level by assigning a set of messages per day to the users’ through the message gateway services.

We are also planning to add certain new features like getting caller’s location along with caller name and number. The other feature which can be added is sharing our current location with a particular contact.

**BIBLIOGRAPHY**

The following books were referred during the analysis and execution phase of the project:

* **Software Engineering**, By Roger.S.Pressman.

### Programming Android: [Java Programming for the New Generation of Mobile Devices](http://erclk.about.com/?zi=0/3U2J), By [Zigurd Mednieks](http://it-ebooks.info/author/1476/), [Laird Dornin](http://it-ebooks.info/author/870/), [G. Blake Meike](http://it-ebooks.info/author/516/), [Masumi Nakamura](http://it-ebooks.info/author/949/).

* **Object Oriented Analysis And Design using UML**, By Ali Bahrami 2nd  Edition,Tata McGraw-Hill.
* **Java 2 Programmers Reference**, By Herbert Schildt, 1st  edition, By MCGraw -Hill companies

### Learning Android: Develop mobile Apps with Java and Eclipse, By Marko Gargenta & Masumi Nakamura.

**REFERENCES**

### Eclipse Galileo Usage from <http://www.eclipse.org/documentation/>

* Android Specifications from <http://www.android.com/>
* Android Basics from <https://developer.android.com/training/basics/firstapp/index.html?hl=cn>
* Android Source Code from <http://developer.android.com/index.html/>
* SQLite Database Usage from <http://www.sqlite.org/>
* **Wikipedia** for further documentation.