**TABLE OF CONTENTS**

**CHAPTER TITLE PAGE NO**

**ABSTRACT iii**

**LIST OF TABLES vii**

**LIST OF FIGURES viii**

**LIST OF ABBREVATIONS ix**

1. **INTRODUCTION 1**
2. **LITERATURE SURVEY 3** 
   1. GREEDY PERIMETER STATELESS 3

ROUTING

* 1. TERMINODE REMOTE ROUTING 4
  2. LOCATION AIDED ROUTING 4

1. **SYSTEM ANALYSIS 6**
   1. EXISTING SYSTEM 6
   2. PROPOSED SYSTEM 7
2. **SYSTEM DESIGN 8**
   1. OBJECTIVE 8
   2. MODULES 8
   3. MODULES DESCRIPTION 8
      1. User Interface design 8
      2. Server design 8
      3. Tracking the location 9
      4. Viewing the tracked location 9
   4. SYSTEM ARCHITECTURE 9
   5. UML DIAGRAMS 10
   6. SPECIFICATION 15
3. **SYSTEM IMPLEMENTATION 16**
   1. IMPLEMENTATION 16
   2. SOFTWARE DESCRIPTION 16
      1. Android 16
      2. SQL yog 27
   3. INPUT DESIGN 28
   4. DATABASE DESIGN 29
   5. OUTPUT DESIGN 30
4. **SYTEM STUDY 33**
   1. FEASIBILITY STUDY 33
      1. Economical feasibility 34
      2. Technical feasibility 35
      3. Operational feasibility 35
5. **SYSTEM TESTING 37**
   1. INTRODUCTION 37
   2. TYPES OF TESTING 37
      1. Unit testing 37
      2. Integration testing 39
      3. Acceptance testing 40
      4. Security testing 41
6. **MAINTENANCE AND DEVELOPMENT 42**
   1. MAINTENANCE 42
   2. DEVELOPMENT 43
7. **CONCLUSION AND FUTURE 44**

**ENHANCEMENTS**

* 1. CONCLUSION 44
  2. FUTURE ENHANCEMENTS 44

**REFERENCES 45**

**APPENDIX 46 LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **TABLE NO** | **TABLE NAME** | **PAGE NO** |
| 1 | Unit testing | 38 |
| 2 | Integration testing | 39 |
| 3 | Acceptance testing | 40 |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **FIGURE NAME** | **PAGE NO** |
| 1 | System Architecture | 9 |
| 2 | Use case diagram | 10 |
| 3 | Class diagram | 11 |
| 4 | Sequence diagram | 12 |
| 5 | Collaboration diagram | 14 |
| 6 | Android Architecture | 19 |
| 7 | Input data | 29 |
| 8 | Values in database | 30 |
| 9 | View Location | 31 |
| 10  11 | View current location  View distance | 31  32 |
|  |  |  |

**LIST OF ABBREVATIONS**

|  |  |
| --- | --- |
| **SYMBOL** | **EXPANSION** |
| SDK | Software Development Kit |
| APK | Android Application Package |
| SQL | Structured Query Language |
| GPS | Global Positioning System |
| GLONASS | Global Navigation Satellite System |
| GPRS | General Packet Radio Services |
| RTLS | Real Time Locating System |

**CHAPTER 1**

**INTRODUCTION**

* 1. **Objective**

The idea of the proposed system is to create an android application which can be downloaded and kept in our mobile phones to view the location of our respective bus. Students, staff and other users can use this application to locate the position of bus. The bus-in-charge of the college can use this application to monitor the status of all the buses.

* 1. **Motivation**

Android applications are familiar due to its user friendliness. This application helps the user to locate the bus and view a map which provides the distance and time along with it.

* 1. **Scope**

Automatic updating of bus locations helps us to identify the respective status of buses without using complex embedded systems and costly techniques. It will be helpful for everyone who is waiting for bus.

* 1. **Problem definition**

At present there are many locating devices in general but they are not client specific. The tracking devices require embedded systems which are quite costlier. This system focuses in finding the information and updating it on the application. Thus it does not require any additional devices for support except mobile phones.

* 1. **Overview of the project**

The bus locator combines the installation of purpose designed computer software at one operational base to enable the owner or a third party to track the vehicle's location, collecting data in the process from the field and delivers it to the base of operation.

* 1. **Project impact**

Once the application is installed in the mobile it enables the user to make a registration with their mobile number which provides a login feature for future use. After signing up they can view the map using “view bus” option. It aims to minimize the problems facing in waiting for bus and calling others to find where the bus is currently present.

* 1. **Project outcome**

The project has two applications, one running on the user side and the other on the sender (bus) side. The location of the bus will be updated in the server whenever the position of bus changes. The application on the user side enables us to view the current status via a map. This application also provides a feature by which the estimated distance and time between the sender and receiver can be found.

**CHAPTER 2**

**LITERATURE SURVEY**

The highly interconnected present world is becoming smarter and easier because of the smart phones and the related gadgets. When a particular problem is taken into consideration, there are many different solutions to it in the current world. Likewise there are many different android applications for the same task. The outcome of those applications might be the same but the way it works may differ and may have certain cons in it which can be neglected using new techniques. The proposed project has overcome those cons using new techniques. A survey has been done on different methods to find out the techniques used and the pros and cons of it.

**2.1** **Greedy Perimeter Stateless Routing**

The concept used here is Greedy Perimeter Stateless Routing (GPSR), a novel routing protocol for wireless datagram networks that uses the positions of routers and a packet's destination to make packet forwarding decisions. GPSR makes greedy forwarding decisions using only information about a router's immediate neighbours in the network topology. When a packet reaches a region where greedy forwarding is impossible, the algorithm recovers by routing around the perimeter of the region. By keeping state only about the local topology, GPSR scales better in per-router state than shortest-path and ad-hoc routing protocols as the number of network destinations increases. Under mobility's frequent topology changes, GPSR can use local topology information and correct new routes quickly. It describes the GPSR protocol, and use extensive simulation of mobile wireless networks to compare its performance with that of Dynamic Source Routing. This simulation demonstrates GPSR's scalability on densely deployed wireless networks [1].

**2.2 Terminode remote routing**

This stated the theory as “Using location information to help routing is often proposed as a means to achieve scalability in large mobile ad hoc networks. However, location-based routing is difficult when there are holes in the network topology and nodes are mobile or frequently disconnected to save battery. Terminode routing, presented here, addresses these issues. It uses a combination of location-based routing (terminode remote routing, TRR), used when the destination is far, and link state-routing (terminode local routing, TLR), used when the destination is close. TRR uses anchored paths, a list of geographic points (not nodes) used as loose source routing information. Anchored paths are discovered and managed by sources, using one of two low overhead protocols: friend assisted path discovery and geographical map-based path discovery. This simulation results show that terminode routing performs well in networks of various sizes. In smaller networks; the performance is comparable to MANET routing protocols. In larger networks that are not uniformly populated with nodes, terminode routing outperforms, existing location-based or MANET routing protocols” [2].

**2.3 Location-Aided Routing**

This stated the concept as “A mobile ad hoc network consists of wireless hosts that may move often. Movement of hosts results in a change in routes, requiring some mechanism for determining new routes. Several routing protocols have already been proposed for ad hoc networks. This paper suggests an approach to utilize location information (for instance, obtained using the global positioning system) to improve performance of routing protocols for ad hoc networks. By using location information, the proposed Location-Aided Routing (LAR) protocols limit the search for a new route to a smaller “request zone” of the ad hoc network. This results in a significant reduction in the number of routing messages. They present two algorithms to determine the request zone, and also suggest potential optimizations to their algorithms” [3].Normalized advance (NADV) for geographic routing in multi hop wireless networks. NADV selects neighbours with the optimal trade-off between proximity and link cost. Coupled with the local next hop decision in geographic routing, NADV enables an adaptive and efficient cost-aware routing strategy. Depending on the objective or message priority, applications can use the NADV framework to minimize various types of link cost. They present efficient methods for link cost estimation and perform detailed simulations in diverse scenarios. Their results show that NADV outperforms current schemes in many aspects: for example, in high noise environments with frequent packet losses, the use of NADV leads to 81% higher delivery ratio. When compared to the centralized routing under certain settings, geographic routing using NADV paths whose cost is close to the optimum [4].

**CHAPTER 3**

**SYSTEM ANALYSIS**

**3.1 Existing system**

There are many applications which focus on tracking the locations of the vehicles. The tracking application that already exists makes use of the automatic vehicle location in individual vehicles. This is done with the help of the software that collects these data and gives the information. Modern vehicle tracking systems commonly use GLONASS (Global Navigation Satellite System ) technology for locating the vehicle. GLONASS is a space-based satellite navigation system analogous to the GPS but operated by the Russian Aerospace Defense Forces. It is widely considered to be an alternative to the GPS system – Global positioning system released for public use by the American Army. In spite of the now obvious redundancy, the GLONASS system was created as an alternative to the American GPS system.  Vehicle information can be viewed on electronic maps via the Internet or specialized software. But those embedded systems are costlier and not a client-specific product. Here client specific product means a system that functions and gives the output to the user or the client when a particular task is specified by the user.

**Drawbacks:**

* The existing system makes use of software to collect the data and give the information.
* Modern systems do not rely on software but make use of Global Navigation Satellite System.
* Those systems are not client specific.
* It makes use of the embedded systems.
* The embedded systems are costly and require additional maintenance.

**3.2 Proposed system**

The proposed system uses the mobile phone which is available with the users and also to bus-in-charge or the system embedded in bus. In this project embedded system is not used. Therefore the application that is developed is made available to the students, staffs and as well as the person in charge of the bus. A simple interface (application) in our mobile helps us to monitor the status of the bus. A server is designed to store the details of the users using it and locations are updated as soon as the users send the request. On the other side bus in charge is given an additional application which should be started by the in charge of the bus once the bus is started. As the bus gets started the server gets periodically updated with the current location. The current location is given as the output to the user when requested. Further enhancements can be added by getting alerts in application in case the bus will be delayed or will not arrive etc. This system is a very simple application which offers reliable and secured usage for tracking a bus. This system makes use of the built in GPS of the mobile phones. The drawbacks of the existing system are taken into consideration and the proposed system is developed neglecting all those drawbacks.

**Advantages:**

* It is a client specific application.
* It is a useful application for the students as well as all the passengers to view the exact location of the bus.
* It eliminates the people from being in a confused state of mind think whether the bus has crossed the stop or not
* It is user friendly.
* Easily affordable.

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 Objective**

This project performs the task of developing an application in android that enables the students and faculty to find the location of the bus easily. The main purpose of this application is to make the students and the faculty aware of the distance between the bus location and the users’ current location and also make the users aware of how long the bus might take to reach the stop.

**4.2 Modules**

This project consists of four major modules

* User Interface design
* Server design
* Tracking the location
* Viewing the tracked location

**4.3 Module Description**

**4.3.1 User Interface design**

In this Module, the process is done by using mobile to perform the user interactions such as login and registration. User details are handled in backend common database. A login is the process by which individual access to a mobile system is provided with controlled access.

**4.3.2 Server design**

In this module the server is designed in such a way that the details which the users give during registration is stored and updated if any changes. The location of the bus is periodically updated. User sends the registration request to the application and creates a valid user-Id. The User can view the bus location only the registration is successful and the user can view the application.

**4.3.3 Tracking the location**

In this module, Real-time locating systems (RTLS) are used to automatically identify and track the location of people in real time. Wireless RTLS tags are used in most RTLS, fixed reference points receive wireless signals from tags to determine viewer’s location.

**4.3.4 Viewing the tracked location**

In this module, Android Google API, GPS (for getting user current location) and GPRS (for internet connectivity between mobile and server) these technologies are used to display the output on the student app. The current location is displayed in a Google map in the app used by the students. It also show the distance between the current location of the student and the location of the bus in the G map.

* 1. **System Architecture**

REGISTRATION

UPDATES THE LOCATION

DRIVER

RUN THE APP

STUDENT

VIEW THE LOCATION OF THE BUS

REQUESTS THE BUS LOCATION

Figure1. System Architecture

* 1. **UML Diagrams**

**4.5.1 Use case diagram**

This diagram depicts the interaction between the actors’ bus device, user and the respective operations.

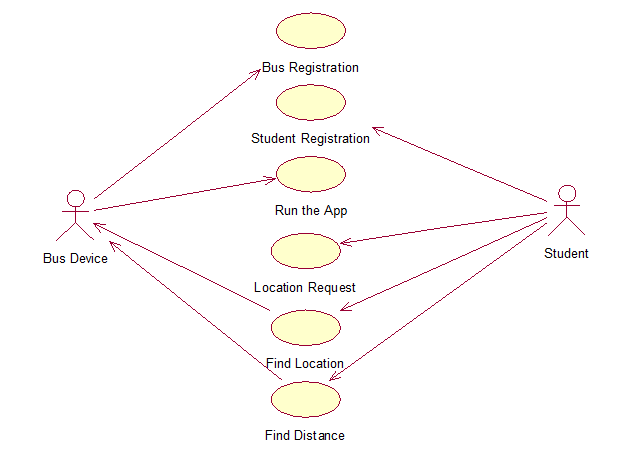


Figure2. Use case diagram

**4.5.2 Class diagram**

The class diagram denotes the attributes and the operations of the each class function.

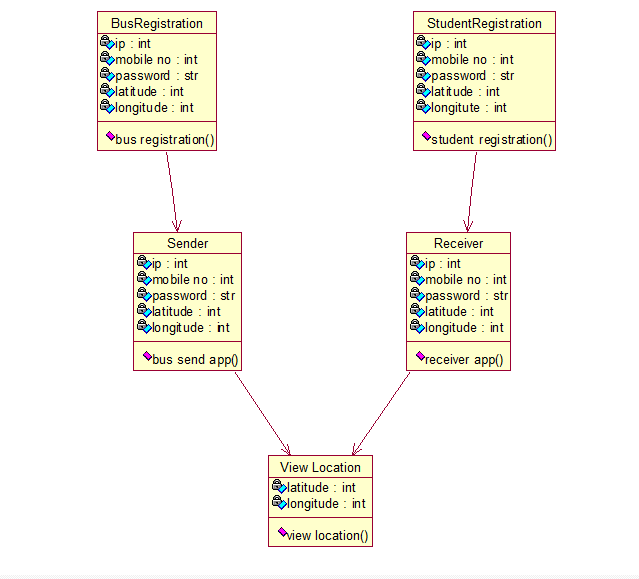
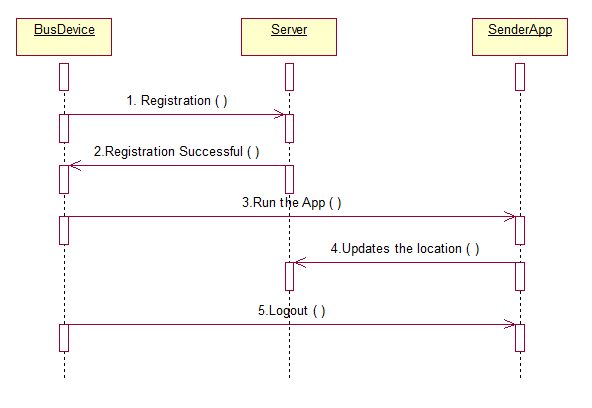


Figure3. Class diagram

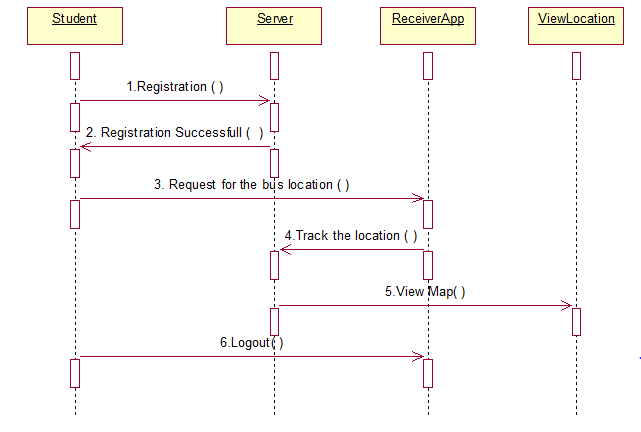
**4.5.3 Sequence diagram**

**** This diagram depicts the interaction between the actors’ bus device, user and the respective operations.

**Sequence of events in bus device**

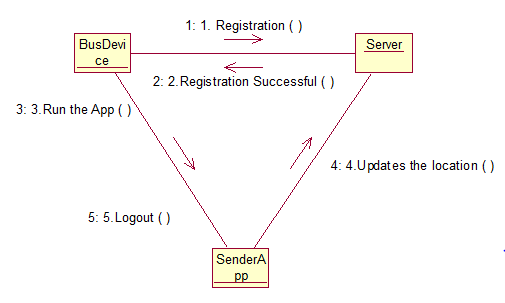
Figure 4.1 Sequence diagram for bus device

**Sequence of events in student application**

Figure 4.2 Sequence diagram for student

**4.5.4Collaboration diagram**

This diagram depicts the interaction between the actors’ bus device, user and the respective operations using sequence diagram. The collaboration diagram will be generated by the sequence diagram.



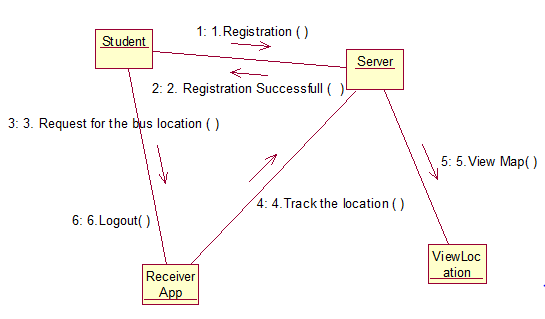
Figure5.1. Collaboration diagram for bus device

Figure5.2. Collaboration diagram for student device

* 1. **Specification**

**4.6.1 Software Specification**

* Operating system : Windows Family
* Database : SQLyog server
* Front end language : XML
* Back end language : Java
* Tool : Android SDK

**4.6.2 Hardware Specification**

* Mobile model : Android mobiles
* Processor : Pentium IV 2.4 GHz
* RAM : 512 MB
* Hard disk : 40 GB

**CHAPTER 5**

**SYSTEM IMPLEMENTATION**

**5.1 Implementation**

Implementation is the most crucial stage in achieving a successful system and giving the user confidence that the new system is effective. Implementation of this project refers to the installation of the package in its real environment to the full satisfaction of the users and operations of the system. In short, implementation constitutes of all activities that are required to put an already tested and completed package into operation. The success of any information system lies in its successful implementation.

System implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The project execution is checked with live environment and the user requirements are satisfied. Proper implementation is essential to provide reliable system to meet the organization requirements.

**5.2 Software Description**

**5.2.1 Android**

Android is a [mobile operating system](https://en.wikipedia.org/wiki/Mobile_operating_system) (OS) currently developed by [Google](https://en.wikipedia.org/wiki/Google), based on the [Linux kernel](https://en.wikipedia.org/wiki/Linux_kernel) and designed primarily for [touch screen](https://en.wikipedia.org/wiki/Touchscreen) mobile devices such as [smart phones](https://en.wikipedia.org/wiki/Smartphone) and [tablets](https://en.wikipedia.org/wiki/Tablet_computer). Android's [user interface](https://en.wikipedia.org/wiki/User_interface) is mainly based on [direct manipulation](https://en.wikipedia.org/wiki/Direct_manipulation_interface), using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a [virtual keyboard](https://en.wikipedia.org/wiki/Virtual_keyboard) for text input. In addition to touch screen devices, Google has further developed [Android TV](https://en.wikipedia.org/wiki/Android_TV) for televisions, [Android Auto](https://en.wikipedia.org/wiki/Android_Auto) for cars and [Android Wear](https://en.wikipedia.org/wiki/Android_Wear) for wrist watches, each with a specialized user interface. Variants of Android are also used on [notebooks](https://en.wikipedia.org/wiki/Laptop), [game consoles](https://en.wikipedia.org/wiki/Video_game_console), [digital cameras](https://en.wikipedia.org/wiki/Digital_camera), and other electronics.

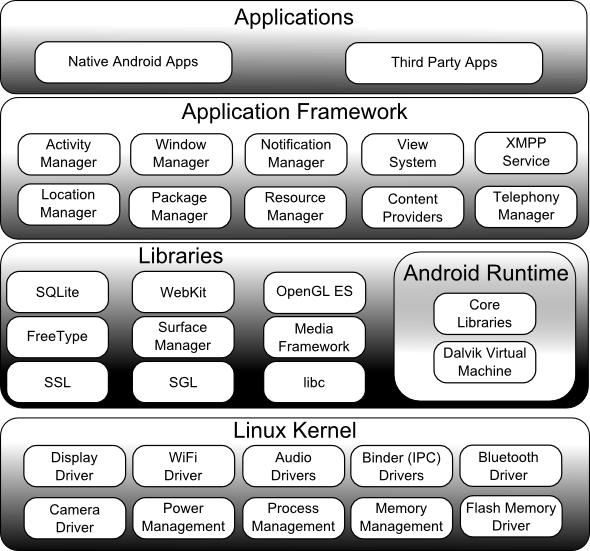
Android's default user interface is mainly based on [direct manipulation](https://en.wikipedia.org/wiki/Direct_manipulation_interface), using touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, along with a [virtual keyboard](https://en.wikipedia.org/wiki/Virtual_keyboard). [Game controllers](https://en.wikipedia.org/wiki/Game_controller) and full-size physical [keyboards](https://en.wikipedia.org/wiki/Computer_keyboard) are supported via [Bluetooth](https://en.wikipedia.org/wiki/Bluetooth) or [USB](https://en.wikipedia.org/wiki/USB). The response to user input is designed to be immediate and provides a fluid touch interface, often using the vibration capabilities of the device to provide [hap tic feedback](https://en.wikipedia.org/wiki/Haptic_technology) to the user. Internal hardware, such as [accelerometers](https://en.wikipedia.org/wiki/Accelerometer), [gyroscopes](https://en.wikipedia.org/wiki/Gyroscope) and [proximity sensors](https://en.wikipedia.org/wiki/Proximity_sensor)  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, or allowing the user to steer a vehicle in a racing game by rotating the device, simulating control of a steering.

Applications ("[apps](https://en.wikipedia.org/wiki/Mobile_app)"), which extend the functionality of devices, are written using the [Android software development](https://en.wikipedia.org/wiki/Android_software_development) kit (SDK) and, often, the [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) programming language that has complete access to the Android APIs. Java may be combined with [C](https://en.wikipedia.org/wiki/C_(programming_language))/[C++](https://en.wikipedia.org/wiki/C%2B%2B), together with a choice of non-default runtimes that allow better C++ support; the [Go](https://en.wikipedia.org/wiki/Go_(programming_language)) programming language is also supported since its version 1.4, which can also be used exclusively although with a restricted set of Android APIs. The SDK includes a comprehensive set of development tools which includes source codes, a simple [debugger](https://en.wikipedia.org/wiki/Debugger), [software libraries](https://en.wikipedia.org/wiki/Software_library), a handset [emulator](https://en.wikipedia.org/wiki/Emulator) based on [QEMU](https://en.wikipedia.org/wiki/QEMU), documentation, several sample codes, and tutorials. Initially, Google's supported [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) was [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software)) using the Android Development Tools (ADT) plug in, in December 2014, Google released [Android Studio](https://en.wikipedia.org/wiki/Android_Studio), based on Interlay, as its primary IDE for Android application development. Other development tools are available, including a [native development kit](https://en.wikipedia.org/wiki/Native_development_kit) (NDK) for applications or extensions in C or C++, [Google App Inventor](https://en.wikipedia.org/wiki/Google_App_Inventor), a visual environment for novice programmers, and various [cross platform mobile web applications frameworks](https://en.wikipedia.org/wiki/Multiple_phone_web_based_application_framework). In January 2014, Google unveiled an framework based on [Apache Cordova](https://en.wikipedia.org/wiki/Apache_Cordova) for porting [Chrome](https://en.wikipedia.org/wiki/Google_Chrome) [HTML 5](https://en.wikipedia.org/wiki/HTML_5) [web applications](https://en.wikipedia.org/wiki/Web_app) to Android, wrapped in a native application shell.

Android has a growing selection of third-party applications, which can be acquired by users by downloading and installing the application's [APK](https://en.wikipedia.org/wiki/APK_(file_format)) (Android application package) file, or by downloading them using an [application store](https://en.wikipedia.org/wiki/Application_store) program that allows users to [install, update, and remove applications](https://en.wikipedia.org/wiki/Package_manager) from their devices. [Google Play Store](https://en.wikipedia.org/wiki/Google_Play_Store) is the primary application store installed on Android devices that comply with Google's compatibility requirements and license the Google Mobile Services software. Google Play Store allows users to browse, download and update applications published by Google and third-party developers; as of July 2013, there are more than one million applications available for Android in Play Store. As of July 2013, 50 billion applications have been installed. Some carriers offer direct carrier billing for Google Play application purchases, where the cost of the application is added to the user's monthly bill.

Due to the open nature of Android, a number of third-party application marketplaces also exist for Android, either to provide a substitute for devices that are not allowed to ship with Google Play Store, provide applications that cannot be offered on Google Play Store due to policy violations, or for other reasons. Examples of these third-party stores have included the [Amazon Appstore](https://en.wikipedia.org/wiki/Amazon_Appstore), [GetJar](https://en.wikipedia.org/wiki/GetJar), and SlideMe. [F-Droid](https://en.wikipedia.org/wiki/F-Droid), another alternative marketplace, seeks to only provide applications that are distributed under [free and open source](https://en.wikipedia.org/wiki/Free_and_open_source) [licenses](https://en.wikipedia.org/wiki/Free_software_license).

Android devices incorporate many optional hardware components, including still or video cameras, [GPS](https://en.wikipedia.org/wiki/GPS), [orientation sensors](https://en.wikipedia.org/wiki/Orientation_sensing), dedicated gaming controls, [accelerometers](https://en.wikipedia.org/wiki/Accelerometer), [gyroscopes](https://en.wikipedia.org/wiki/Gyroscope), barometers, [magnetometers](https://en.wikipedia.org/wiki/Magnetometer), [proximity sensors](https://en.wikipedia.org/wiki/Proximity_sensor), [pressure sensors](https://en.wikipedia.org/wiki/Pressure_sensor), thermometers, and [touch screens](https://en.wikipedia.org/wiki/Touchscreen). Some hardware components are not required, but became standard in certain classes of devices, such as smart phones, and additional requirements apply if they are present. Some other hardware was initially required, but those requirements have been relaxed or eliminated altogether.

Figure 6. Android Architecture

For example, as Android was developed initially as a phone OS, hardware such as microphones were required, while over time the phone function became optional. Android used to require an [autofocus](https://en.wikipedia.org/wiki/Autofocus) camera, which was relaxed to a [fixed-focus](https://en.wikipedia.org/wiki/Fixed-focus_lens) camera if present at all, since the camera was dropped as a requirement entirely when Android started to be used on [set-top boxes](https://en.wikipedia.org/wiki/Set-top_box).

i) **Application Framework**

The Application Framework is a set of services that collectively form the environment in which Android applications run and are managed. This framework implements the concept that Android applications are constructed from reusable, interchangeable and replaceable components. This concept is taken a step further in that an application is also able to publish its capabilities along with any corresponding data so that they can be found and reused by other applications.

The Android framework includes the following key services:

* Activity Manager – Controls all aspects of the application lifecycle and activity stack.
* Content Providers – Allows applications to publish and share data with other applications.
* Resource Manager – Provides access to non-code embedded resources such as strings, color settings and user interface layouts.
* Notifications Manager – Allows applications to display alerts and notifications to the user.
* View System – An extensible set of views used to create application user interfaces.
* Package Manager – The system by which applications are able to find out information about other applications currently installed on the device.
* Telephony Manager – Provides information to the application about the telephony services available on the device such as status and subscriber information.
* Location Manager – Provides access to the location services allowing an application to receive updates about location changes.

**ii)Android Libraries**

This category encompasses those Java-based libraries that are specific to Android development. Examples of libraries in this category include the application framework libraries in addition to those that facilitate user interface building, graphics drawing and database access.

A summary of some key core Android libraries available to the Android developer is as follows:

* android.app – Provides access to the application model and is the cornerstone of all Android applications.
* android.content – Facilitates content access, publishing and messaging between applications and application components.
* android.database – Used to access data published by content providers and includes SQLite database management classes.
* android.graphics – A low-level 2D graphics drawing API including colors, points, filters, rectangles and canvases.
* android.hardware – Presents an API providing access to hardware such as the accelerometer and light sensor.
* android.opengl – A Java interface to the OpenGL ES 3D graphics rendering API.
* android.os – Provides applications with access to standard operating system services including messages, system services and inter-process communication.
* android.media – Provides classes to enable playback of audio and video.
* android.net – A set of APIs providing access to the network stack. Includes android.net.wifi, which provides access to the device’s wireless stack.
* android.provider – A set of convenience classes that provide access to standard Android content provider databases such as those maintained by the calendar and contact applications.
* android.text – Used to render and manipulate text on a device display.
* android.util – A set of utility classes for performing tasks such as string and number conversion, XML handling and date and time manipulation.
* android.view – The fundamental building blocks of application user interfaces.
* android.widget - A rich collection of pre-built user interface components such as buttons, labels, list views, layout managers, radio buttons etc.
* android.webkit – A set of classes intended to allow web-browsing capabilities to be built into applications.

Having covered the Java-based core libraries in the Android runtime, it is now time to turn our attention to the C/C++ based libraries contained in this layer of the Android software stack.

**iii)Linux kernel**

Positioned at the bottom of the Android software stack, the Linux Kernel provides a level of abstraction between the device hardware and the upper layers of the Android software stack. Based on Linux version 2.6, the kernel provides preemptive multitasking, low-level core system services such as memory, process and power management in addition to providing a network stack and device drivers for hardware such as the device display, Wi-Fi and audio.

The original Linux kernel was developed in 1991 by Linus Torvalds and was combined with a set of tools, utilities and compilers developed by Richard Stallman at the Free Software Foundation to create a full operating system referred to as GNU/Linux. Various Linux distributions have been derived from these basic underpinnings such as Ubuntu and Red Hat Enterprise Linux.

It is important to note, however, that Android only uses the Linux kernel. That said, it is worth noting that the Linux kernel was originally developed for use in traditional computers in the form of desktops and servers. In fact, Linux is now most widely deployed in mission critical enterprise server environments. It is a testament to both the power of today’s mobile devices and the efficiency and performance of the Linux kernel that we find this software at the heart of the Android software stack.

**iv)Android Runtime**

The Linux kernel provides a multitasking execution environment allowing multiple processes to execute concurrently. It would be easy to assume, therefore, that each Android application simply runs as a process directly on the Linux kernel. In fact, each application running on an Android device does so within its own instance of the Dalvik virtual machine (VM).

Running applications in virtual machines provides a number of advantages. Firstly, applications are essentially sandboxed, in that they cannot detrimentally interfere (intentionally or otherwise) with the operating system or other applications, nor can they directly access the device hardware. Secondly, this enforced level of abstraction makes applications platform neutral in that they are never tied to any specific hardware.

The Dalvik virtual machine was developed by Google and relies on the underlying Linux kernel for low-level functionality. It is more efficient than the standard Java VM in terms of memory usage, and specifically designed to allow multiple instances to run efficiently within the resource constraints of a mobile device.

In order to execute within a Dalvik VM, application code must be transformed from standard Java class files to the Dalvik executable (.dex) format, which has a 50% smaller memory footprint than standard Java bytecode. Standard Java class files can usually (though not always) be converted to Dex format using the dx tool included with the Android SDK.

**v)Android components**

**a) Activity -**An activity in Android represents a single screen with user interface. An android application typically consists of several activities.  An activity interacts with the user to do one thing, only ONE, such as Unlock screen, dial a phone, view home etc. An application consists of multiple activities that are loosely bound together. But only one activity can be specified as main activity which is displayed while launching the application. Every time an activity starts previous activity is stopped but the data is preserved. For instance an email application will have one activity to display list of new emails, another activity to compose email, another activity for reading emails and so on.

**b)View -** An activity contains views and view groups. Views in Andriod are UI components that could be used to build user interface. Examples are buttons, label, and text boxes. One or more views can be grouped together into a View Group. Types of views in android are,

**Basic Views**

* + List Views
  + Picker Views
  + Menus
  + Display Views

View Group provides a layout where you can order the appearance of views. Android support the following view groups.

      .          Linear Layout

·         Table Layout

·         Absolute Layout

·         Frame Layout

·         Relative Layout

·         Scroll View

**c) Services**

 Services are something that does not require user interface. It performs operation without user interaction in the background but does not get initiated without user invocation. Another component such as an activity should invoke. To be more technical, service does not have an independent thread, they make use of the main thread from hosting process. For instance you can download something while you get involved in some other application, where downloading is a better and a worthy service running in background.

**d) Content Provider**

  Android is embedded with SQLite Database where all your data gets stored. Content providers in Android manage data that is being shared by more than one application. Consider a case where your Contacts are stored in centralized repository, so here many applications may require access to it or may even require modifying it. In such cases these applications need to query the centralized repository through content providers, so an application with proper permissions can read or modify the Contacts based on permissions. Content provider is concept evolved to manage common data based on permissions.

**e) Broadcast Receivers**

   Broadcast receiver is also a component where you can register for system or application events. Once registered, you will be notified about the events. Broadcast originates from the system as well as applications. Instance for broadcast originating from the system is ‘low battery notification’. Application level is, like when you download an mp3 file, Mp3 player gets notified about it, and gets added to player list. For this action to take place, mp3 player has to register for this event.

**f) Intent**

Intents are messages that allow components to request activities from other components. Consider an activity requires another activity to perform some functionality in such cases communication between activities occurs via Intent.

**vi) Android SDK**

While the Android SDK provides a great starting point for an individual developer of Android code, it is missing features that facilitate the collaboration and coordination needed when a team is developing an Android application. By integrating the device-specific, native platform SDK with a compatible commercial development solution, agile teams can achieve tremendous efficiencies and higher-quality results. The [Android](http://tech.opensystemsmedia.com/android/) [Software Development](http://embedded-computing.com/topics/software-development) Kit (SDK), which Google provides for free, is a great starting point for developing an Android-based smart device application. The SDK contains a variety of useful materials for developers, including extensive documentation, tutorials, samples, best practice guidance, and an array of tools for numerous development purposes. The SDK’s set of Java APIs gives [application developers](http://embedded-computing.com/topics/application-developers) access to native functions that Android-based devices support, such as 2D and 3D graphics, multimedia codec’s, telephony features, and location services. A device emulator in the SDK allows developers to try out their code directly from the development environment without requiring a physical device. And the SDK has an Eclipse plug-in that exposes the Android APIs and SDK tools in a rich Integrated Development Environment (IDE).For an individual developer of Android code, the SDK is valuable and is becoming more so as it is being extended with new features all the time. However, it is missing features that facilitate the collaboration and coordination needed when a development team is creating the application. By integrating the device-specific, native platform SDK from Google with a compatible commercial development solution, [agile](http://embedded-computing.com/topics/agile) teams can achieve tremendous efficiencies and better results. Integrating the native [Android SDK](http://tech.opensystemsmedia.com/android/#android%20sdk) with a commercial development environment opens the door to seamless source control, iterative application planning, effortless work item management, and a host of enterprise-quality development capabilities for an Android application.

**5.2.2 SQLyog**

SQLyog is a [GUI](https://en.wikipedia.org/wiki/Graphical_user_interface) tool for the [RDBMS](https://en.wikipedia.org/wiki/RDBMS) [MySQL](https://en.wikipedia.org/wiki/MySQL). It is developed by Webyog, Inc. based out of [Bangalore, India](https://en.wikipedia.org/wiki/Bangalore,_India) and [Santa Clara, California](https://en.wikipedia.org/wiki/Santa_Clara,_California). SQLyog is being used by more than 30,000 customers worldwide and has been downloaded more than 2,000,000 times.

**Features**

* 64 bit binaries are available from version 11.0.
* Editor with syntax highlighting and various automatic formatting options.
* Intelligent Code Completion.
* Data manipulations (INSERT, UPDATE, DELETE) may be done from a spreadsheet-like interface. Both raw table data and a result set from a query can be manipulated.
* Visual Schema Designer.
* Visual Query Builder.
* Query Formatter.
* Wizard driven Tool for import of data from [ODBC](https://en.wikipedia.org/wiki/ODBC)-databases
* Backup Tool for performing unattended backups. Backups may be compressed and optionally stored as a file-per-table as well as identified with a timestamp.
* 'SQL Scheduler and Reporting Tool' - a tool for scheduling and automating execution of any sequence of SQL statements. Result of queries may be sent as [HTML](https://en.wikipedia.org/wiki/HTML)-formatted reports.
* Schema/Structure Synchronization and Data Synchronization.
* Query Profiler and Redundant Index Finder.
* All automated jobs have mail alerting and reporting options.
* Full character set/[Unicode](https://en.wikipedia.org/wiki/Unicode) support.
* A 'Data Search' feature using a Google-type search syntax translated transparently for user to SQL.
* Form view to display one row at a time - a great way to enter/edit data.
* Foreign key lookup.
* Visual Data Compare.

SQLyog works on the [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) platform starting from [Windows XP](https://en.wikipedia.org/wiki/Windows_XP)/[Windows 2003](https://en.wikipedia.org/wiki/Windows_2003) to [Windows 8](https://en.wikipedia.org/wiki/Windows_8)/[Server 2008 R2](https://en.wikipedia.org/wiki/Windows_Server_2008_R2) (Windows 9x/ME support was removed in version 5.0 primarily because of lack of [Unicode](https://en.wikipedia.org/wiki/Unicode) support in those early Windows versions and Windows 2000 support stopped with version 8.6). It has also been made to work under [Linux](https://en.wikipedia.org/wiki/Linux) and various Unix's (including Mac [OS X](https://en.wikipedia.org/wiki/OS_X)) using the [Wine](https://en.wikipedia.org/wiki/Wine_(software)) environment. Further a subset of SQLyog Enterprise/Ultimate functionalities are available with the free SJA (SQLyog Job Agent) for Linux as a native Linux utility. This makes it possible to specify and test 'scheduled jobs' on a Windows environment and port execution parameters seamlessly to a Linux environment.

**5.3 Input design**

Input design is the process of converting user-originated inputs to a computer-based format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. The input design requirements are user friendliness and consistent format in the project, the input design is made as an application with buttons for “sign-up” and “view bus”.

In this system the input designs are

* Application home page
* Sign up page
* View page.

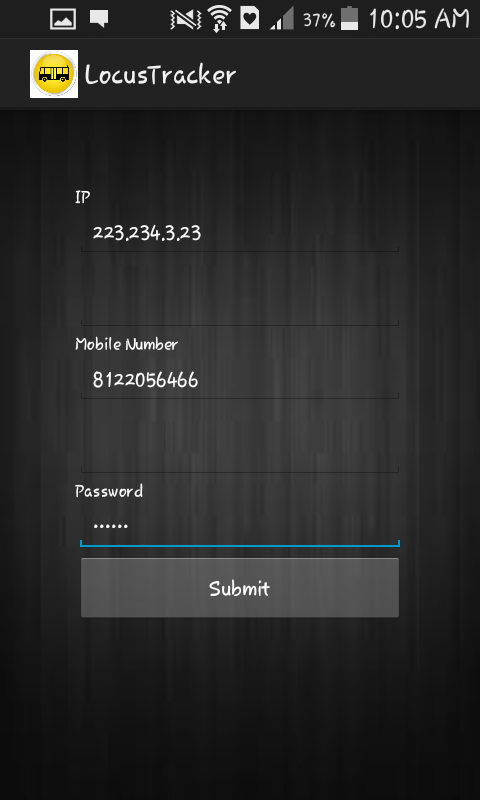


Figure 7. Input data

**5.4 Database design**

Database is a collection of data which is stored in the form of tables. In this system the database table is for storing the latitude, longitude, mobile number of the user and password details.

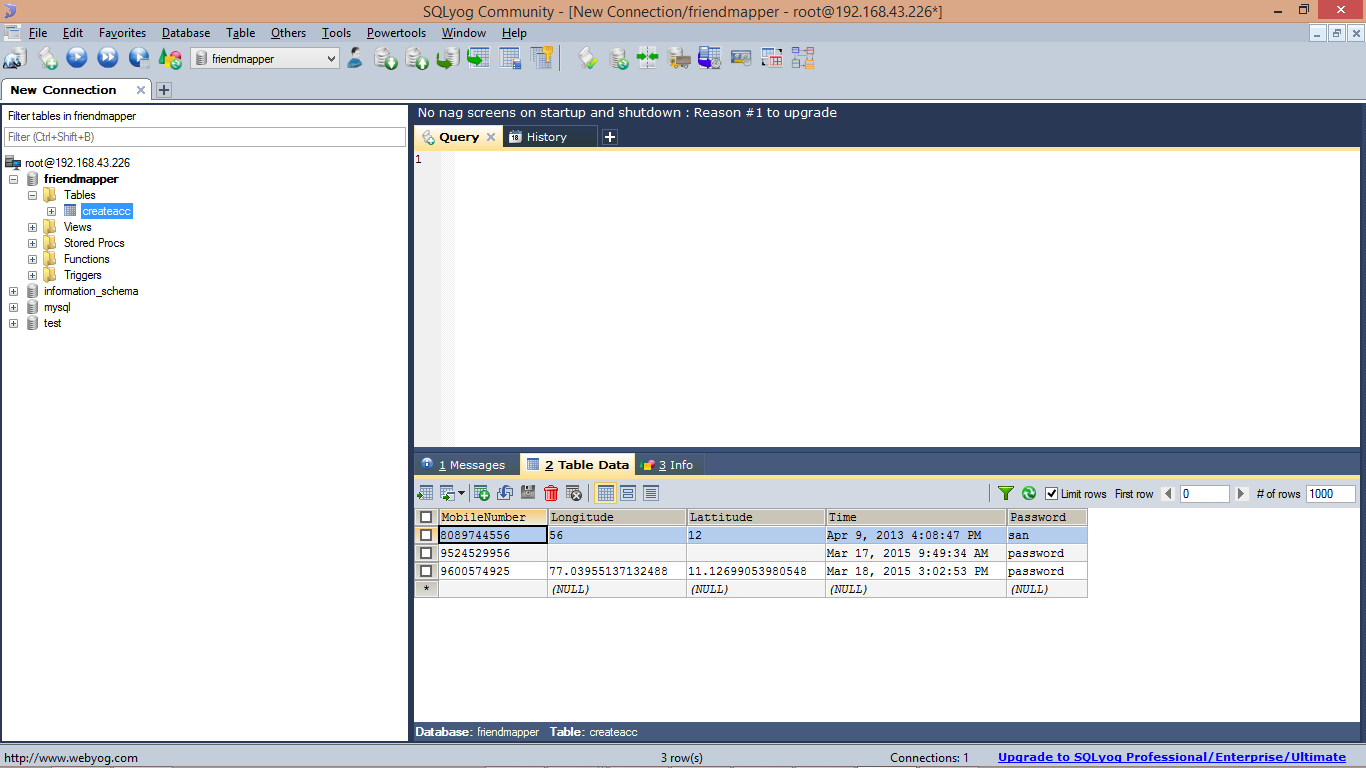


Figure 8. values stored in database

**5.5 Output design**

Output design generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application. The output design specifies the results. The output design is one that provides a customized page to the end user. The output design acts as a medium of communication to the user by providing the desired page that may be either used for storing data or fetching from the database. A quality output is one, which meets the requirements of the end-user and presents the information clearly. In any system the results of processing are communicated to the user and to other systems through outputs. In our system the final output is the map which provides a visual presentation about the bus location.

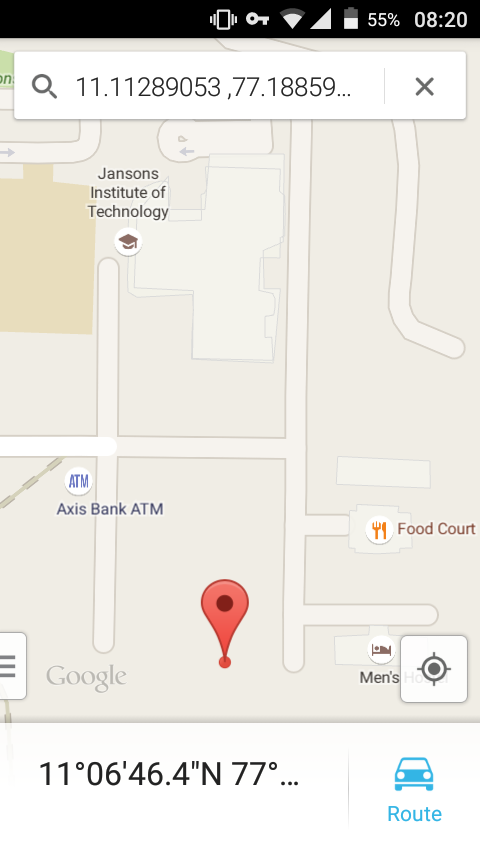
****

Figure 9. View location

The next picture depicts to view the current location of our bus using Google map.

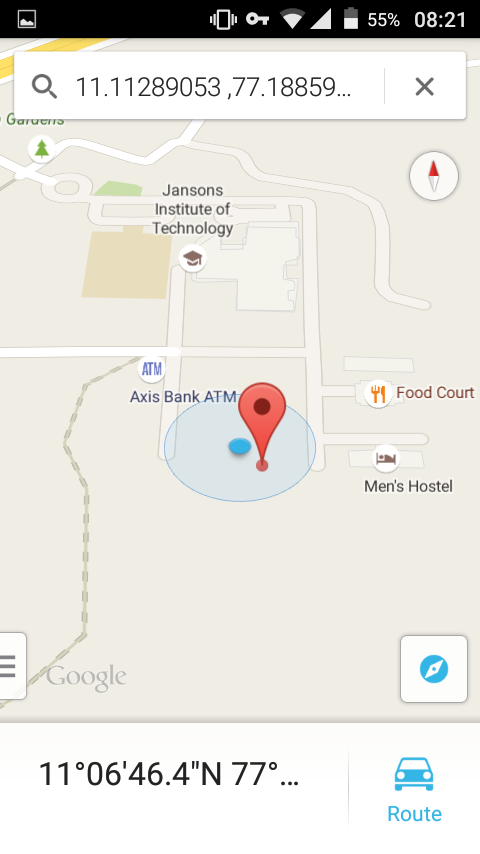


Figure 10.view current location

This picture depicts to find the distance between our current location and the bus location. Then also find the time taken to reach our destination.

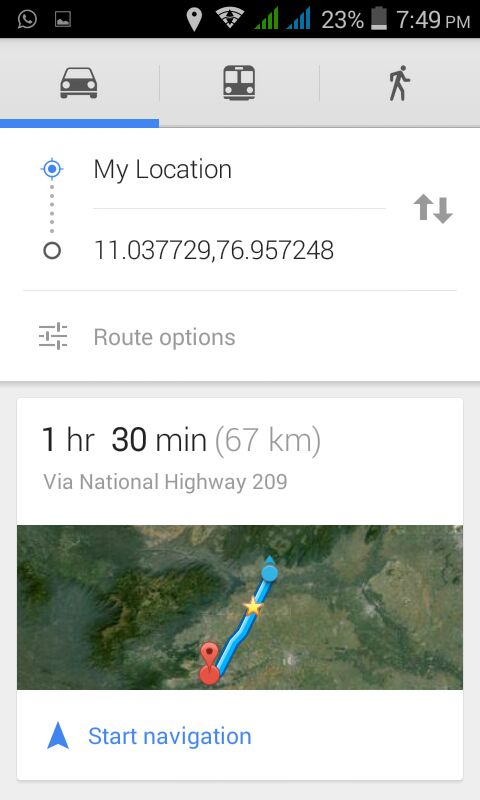


Figure11.View distance

**CHAPTER 6**

**SYSTEM STUDY**

**6.1 Feasibility study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

The feasibility study investigates the problem and the information needs of the stake holders. It seeks to determine the resources required to provide an information system solution, the cost and benefits of such a solution, and the feasibility of such a solution. The analyst conducting the study gathers information using a variety of methods, the most popular of which are:

* Interviewing users, employees, managers, and customers.
* Developing and administering questionnaires to interested stakeholders, such as potential users of the information system.
* Observing or monitoring users of the current system to determine their needs as well as their satisfaction and dissatisfaction with the current system.
* Collecting, examining and analyzing documents, reports, layouts, procedures, manuals, and any other documentation relating to the operations of the current system.
* Modeling, observing and simulating the work activities of the current system.
* An analysis of the ability to complete a project successfully, taking into account legal, economic, technological, scheduling and other factors. Rather than just diving into a project and hoping for the best, a feasibility study allows project managers to investigate the possible negative and positive outcomes of a project before investing too much time and money.

The goal of the feasibility study is to consider alternative information system solutions, evaluate their feasibility, and propose the alternative most suitable to the organization. The components are:

* **Economical feasibility**
* **Technical feasibility**
* **Operational feasibility**

**6.1.1 Economical feasibility**

It refers to the benefits or Outcomes we are deriving from the product as compared to the total cost we are spending for developing the product. If the benefits are more or less the same as the older system, then it is not feasible to develop the product.

In the present system, the development of new product greatly enhances the accuracy of the system and cuts short the delay in the processing of Birth and Death application. The errors can be greatly reduced and at the same time providing a great level of security. Here we don’t need any additional equipment except memory of required capacity.

No need for spending money on client for maintenance because the database used is web enabled database.

**6.1.2 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 the kind of technology to be deployed) etc. A number of issues have to be considered while doing a technical analysis.

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.

Find out whether the organization currently possesses the required technologies:

* Is the required technology available with the organization
* If so is the capacity sufficient
* For instance – “Will the current printer be able to handle the new reports and forms required for the new system” These possibilities are discussed.

**6.1.3 Operational feasibility**

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 are 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.

Since the proposed system aims to reduce the hardships encountered in the existing manual system, the new system was considered to be operational feasible.

**CHAPTER 7**

**SYSTEM TESTING**

**7.1 Introduction**

The most important phase in system development life cycle is system testing. The number and nature of errors in a newly designed system depends on the system specifications and the time frame given for the design.

A newly designed system should have all the subsystems working together, but in reality each subsystems work independently. During this phase, all the subsystems are gathered into one pool and tested to determine whether it meets the user requirements.

Testing is done at two levels – Testing of individual modules and testing the entire system. During the system testing, the system is used experimentally to ensure that the software will run according to the specifications and in the way the user expects. Each test case is designed with the intent of finding errors in the way the system will process it.

Testing plays a very crucial role in determining the reliability and efficiency of software and hence is a very important stage in software development. Software testing is done at different levels. They are the unit testing and system testing which comprises of integration testing and acceptance testing.

**7.2 Types of testing**

The following tests are performed on depending on our testing requirements.

**7.2.1 Unit Testing**

Instead of testing the system as a whole, unit testing focuses on the modules that make up the system. Each module is taken up individually and testing for the correctness in the coding and logic. Errors resulting from interaction of modules are initially avoided.

The advantages of unit testing are:

* Size of a module is quite small and errors can easily be located.
* Confusing interaction of multiple errors in widely different parts of software eliminated.
* Module level testing can be exhaustive.

The obvious assumption made when unit test is pursued is that, individual modules can be isolated from the system module interacts with other modules in the system and, to isolate module, the analyst must stimulate these interactions.

That is the analyst must create driver modules to call the procedures in the module and stub functions for the module to call. The cost involved in creation of this stimulated environment, may or at times is prohibited.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test scenario | Test case | Expected o/p | Actual o/p | Result |
| Sign up | Student registration  Get the latitude and longitude | Mobile number  and password  Application should get by itself | Valid login  Automatic retrieval of  values | PASS  PASS |
| Output | Check whether bus location is retrieved | Current bus location should be obtained in the map. | Current bus location is obtained in the map. | PASS |

Table 1 Unit Testing

**7.2.2 Integration Testing**

Integration testing in the software testing model comes before system testing and after the unit testing has been done. The way that the integration testing works is, by getting the individual modules that have been through the unit testing phase and integrating each module into a group. The integration testing phase will make sure when the modules are being integrated together that any problems, for example errors or bugs, caused due to the integration of the modules are eliminated.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test scenario | Test case | Expected o/p | Actual o/p | Result |
| Registration | Sender side registration. | Location will be updated in the database. | Location updated | PASS |
| Sign up | Student  Registration | Storage of Mobile number and password. | User data are updated in the database | PASS |
| View | Location retrieval. | Sender data should be viewed on receiver side. | Bus location is viewed by the user. | PASS |

Table 2. Integration testing

Integration testing does not deal with the integration of the whole system but deals with the integration of a process in the system. In the integration testing stage there are three things that are created, to ensure that the integration of the modules is successful and that it runs successfully as well, a test plan, test cases and test data is produced to effectively test that the integration is successful.

**7.2.3 Acceptance Testing**

It tests for the errors resulting from integration modules. One specific target of integration testing is the interface: whether parameters match on both sides as to type, permissible ranges and meaning. Analyst tries to find areas where modules have been designed with different specifications.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test scenario | Test case | Expected o/p | Actual o/p | Result |
| Application execution | Run sender side application | Location updater form will be appeared. | Location updater form is viewed. | PASS |
| Warning messages | Alerts | Alerts will be sent from the sender. | Alerts delivered to the user. | PASS |

Table 3.Acceptance testing

This is the top-level testing. In this all modules tested separately would be put together and tested for producing the ultimate result of the system. The main emphasis during this testing will be on the interface between the modules. By applying various business rules generated as a part of test cases, we were able to ease certain design level complexities.

**7.2.4 Security testing**

It is basically a type of software testing that is done to check whether the application or the product is secured or not. It checks to see if the application is vulnerable to attacks, if anyone hack the system or login to the application without any authorization.

**CHAPTER 8**

**MAINTENANCE AND DEVELOPMENT**

**8.1 Maintenance**

The objectives of the maintenance work are to make sure that the system gets into work all time without any bug. Provision must be for environmental changes which may affect the computer or the software system. This is called the maintenance of the system. Nowadays, there is rapid change in the software world. Due to this rapid change, the system must be capable of adapting these changes. In this project, the process can be added without affecting other parts of the system.

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

Implementation is the process of converting a new system design into operation. It is a phase that focuses on user training, site preparation and file conversion for installing a candidate system. The important factor that should be considered here is that the conversion should not disrupt the functioning of the organization.

The application is implemented in the Internet Information Services web server under the windows Professional and accessed from various clients.

**8.2 Development**

Top – down approaches emphasize planning and complete understanding of the system. It is inherent that no coding can begin until a sufficient level of detail has been reached in the design of at least some part of the system. Programming, actually writing software code is just one part of the process, which is why it is prefer to be called developers rather than merely programmers.

Top – down programming is a programming style, the mainstay of traditional procedural languages, in which design by specifying complex task and then dividing them into separate modules. Eventually, the components are specific enough to be coded and the program is written.

The waterfall model is a sequential design process, often used in software development processes, in which progress is seen as flowing steadily downwards through the phases of Initiation, Analysis, Design, Code, Testing, Production/ implementation and Maintenance

**CHAPTER 9**

**CONCLUSION AND FUTURE ENHANCEMENT**

**9.1 Conclusion**

This project is an android application that will be available in user's mobile from which our college students or staff can view their respective bus location. From this information, they can predict when the bus will arrive in their stop. Also the bus-in-charge of the college can view the status all the buses and monitor them periodically.

It is concluded that the application works well and good. The application is tested very well and errors are properly debugged. The application is simultaneously accessed from more than one user. Simultaneous login from more than one place is tested. This system is user friendly so everyone can use it easily. Proper documentation is provided. The end user can easily understand how the whole system is implemented by going through the documentation.

**9.2 Future Enhancement**

There is scope for future enhancement in this project. The world of computer fields is not static; it is always subject to be dynamic. The technology that is famous today becomes outdated the very next day. To keep abstract of technical improvements, the system may be further refined. So, it is not concluded. Yet it will improve with further enhancements. Enhancements can be done in an efficient manner. The application can be extended to send alerts if the bus is nearby.

* Vehicle tracking systems can be used for tracking, routing, dispatch, on-board information and security**.**
* Asset tracking*:* Companies are in need to track valuable assets with safety transport.
* Mobile sales professionals can access real-time locations.

**REFERENCES**

[1] B. Karp and H.T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” Proc. ACM MobiCom, pp. 243- 254, Aug. 2000.

[2] L. Blazevic, S. Giordano, and J.-Y. LeBoudec, “A Location Based Routing Method for Mobile Ad Hoc Networks,” IEEE Trans. Mobile Computing, vol.4, no. 2, pp. 97-110, Mar. 2005.

[3] Y. Ko and N.H. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks,” ACM/Baltzer Wireless Networks, vol. 6, no. 4, pp. 307-321, Sept. 2002.

[4] S. Lee, B. Bhattacharjee, and S. Banerjee, “Efficient Geographic Routing in Multihop Wireless Networks,” Proc. ACM MobiHoc, pp. 230-241, May 2005.

[5] James, “Android App Stats”. Archived from the orginal on January 2,2011. Retrived December 31,2010.

[6] en.wikipedia.org/wiki/GPS \_ tracking \_unit

[7] en.wikipedia.org/wiki/Vehicle\_tracking\_system