For

Any CAB

Prepared by

<NAME1>

<NAME2…>

December 26, 2014

# Introduction

AnyCab is a taxi cab finder that combines the web and mobile android platform. AnyCab connects you directly to taxi drivers through the agency in the TAP of a button; you can find yourself in a taxi cab faster than ever.

## Purpose

AnyCab introduces a completely unique driver’s application, where drivers can receive push notifications of the passenger’s location. The driver’s application like the passenger app is completely Free and light. AnyCab is a great online / mobile platform for both taxi drivers and passengers. No more waving or whistling to stop a taxi, just “TAP for Taxi” and AnyCab will instantly connect you.

## Product Scope

Users can download an application on to their phones which can be used to locate cabs which are nearby to their current location by taking the GPS coordinates of the user. He/she can then call the cab agency to book a ride in that cab. The agency would forward the users coordinates on to the cabbies application and the cabbie can locate the user without any hassle.The cab agency can track each cab driver using the information from their mobile phones. Users can notify authorities if a cabbie misbehaves as the application provides users with information on the cab driver.

# Overall Description

## Product Perspective

Anycab provide a system where the entire cab system has a central control. Every cab agency or private cabbies can register at our site. Details of every cab driver would be stored inside the database. Each cabbie would have an android application on their mobiles. The application that calls the taxi cab driver near you. Let your cell phone with GPS to find the nearest taxi from his position. To order a taxi online with the phone simply select your starting point . You see your current location and taxis available nearby. Just use the button REQUEST TAXI . You can follow the driver in real time , will know which vehicle he will be arriving ( car model and plate) and receive information to be able to get in touch with him directly .

# External Interface Requirements

## Hardware Interfaces

Processor : Pentium Dual-Core

Memory : 2 GB RAM or higher

Processor Speed : 782 MHz

Storage : 320 GB HDD

Monitor : 1024 \* 768 Resolution Color Monitor

## Software Interfaces

The product will be comprised of interaction between the following software products:

1. **Java Standard Edition (JAVA SE)**

It includes a huge upgrade to the Java programming model and a coordinated evolution of the JVM, Java language, and libraries. Java 8 includes features for productivity, ease of use, improved polyglot programming, security and improved performance

1. **Android SDK**

Android software development is the process by which new applications are created for the Android operating system. Applications are usually developed in the Java programming language using the Android Software Development Kit.

1. **Eclipse Kepler**

Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Written mostly in Java, Eclipse can be used to develop android applications.

**FRONT END:**

**a) JAVA**

Java Soft, an operational company of sun Microsystems, spend years developing a high-powered programming language for the 90’s beyond. Java delivers on promise by being most robust, easy-to-use, and versatile programming language available today. It includes the best aspects of earlier programming languages such as C and C++, allows you to create powerful applications, has features such as built-in multimedia capabilities that make creating multimedia presentations easier than ever, and leaves out those things we all hated about C and C++ like multiple inheritance, operator overloading, and pointers.

Java is first and foremost an object-oriented programming language. Many programmers are surprised when they discover how easy it is to follow sound object-oriented design practices with Java. The following sectors give you a better understanding of what java offers.

**b) JAVA SERVER PAGES** (**JSP**)

It’s a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. Released in 1999 by Sun Microsystems,[3] JSP is similar to PHP, but it uses the Java programming language. To deploy and run JavaServerPages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required

JSP can be used independently or as the view component of a server-side model view controller design, normally with JavaBeans as the model and Java servlets (or a framework such as Apache Struts) as the controller. This is a type of Model 2 architecture. JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use Java byte code rather than a native software format. Like any other Java program, they must be executed within a Java virtual machine (JVM) that integrates with the server's host operating system to provide an abstract platform-neutral environment. JSPs are usually used to deliver HTML and XML documents, but through the use of Output Stream, they can deliver other types of data as well. The Web container creates JSP implicit objects like page Context, servletContext, session, request & response.

**BACK END:**

**a) CLOUD DATABASE**

* A cloud database is a database that typically runs on a cloud computing, such as Amazon EC2, GoGrid, Salesforce and Rackspace. There are two common deployment models: users can run databases on the cloud independently, using a virtual machine image, or they can purchase access to a database service, maintained by a cloud database provider. Of the databases available on the cloud, some are SQL -based and some use a No SQL data model. Most database services offer web-based consoles, which the end user can use to provision and configure database instances. For example, the Amazon Web Services web console enables users to launch database instances, create snapshots (similar to backups) of databases, and monitor database statistics.
* Database services consist of a database manager component, which controls the underlying database instances using a service API. The service API is exposed to the end user, and permits users to perform maintenance and scaling operations on their database instances. For example, the Amazon Relational Database Service's service API enables creating a database instance, modifying the resources available to a database instance, deleting a database instance, creating a snapshot (similar to a backup) of a database, and restoring a database from a snapshot.
* Database services make the underlying software stack transparent to the user - the stack typically includes the operating system, the database and third-party software used by the database. The service provider is responsible for installing, patching and updating the underlying software stack.
* Database services take care of scalability and high availability of the database. Scalability features differ between vendors - some offer auto-scaling, others enable the user to scale up using an API, but do not scale automatically. There is typically a commitment for a certain level of high availability (e.g. 99.9% or 99.99%).

**3.4 SQLite**

SQLite is a relational database management system contained in a C programming library. 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-compliant and implements most of the SQL standard, using a dynamically and weakly typed SQL syntax that does not guarantee the domain integrity.

SQLite is a popular choice as embedded database for local/client storage in application software such as web browsers. It is arguably the most widely deployed database engine, as it is used today by several widespread browsing, operating systems, and embedded systems, among others. SQLite has many bindings to programming languages. The source code for SQLite is in the public domain.

Unlike client-server database management systems, the SQLite engine has no standalone processes with which the application program communicates. Instead, the SQLite library is linked in and thus becomes an integral part of the application program. The library can also be called dynamically. The application program uses SQLite's functionality through simple function calls, which reduce latency  in database access: function calls within a single process are more efficient than inter-process communication. SQLite stores the entire database as a single cross-platform file on a host machine. SQLite read operations can be multitasked, though writes can only be performed sequentially.

# Other Requirements

## Performance Requirements

• The overall system should be fast and error free.

• It should have built in error checking and correction facilities.

• The system should be able to handle large amount of data comfortably.

## Software Quality Attributes

Reliability

In order to ensure reliability, this system is being designed using software that is established to be stable and easy to use.

Availability

This system is designed to run 24/7 and be readily available to the user.

**SYSTEM DESIGN**

**Input Design**

Depending upon the output required the inputs hat needed to produce these outputs are determined. The next step is to decide how the data is captured; one-way is to fill the data on forms. The next step is to convert the data into machine-readable form; this is termed as data entry. The final is for a computer programs to rad this data into memory in order to process it: this last step is called Data input.

There are many ways of capturing, entering or inputting into the computer. Most systems capture data on forms as transactions are made. These data then goes through the data entry and data input stages. The method selected by weighing the following factors against each other.

* Cost

If special equipment is required to implement a particular choice of method, it is verified whether extra cost can be justified by expected benefits.

* Accuracy

Verification is made whether it is critical that the needed to be 100% accurate.

* Time

The time Constraints for data entry considered.

* Controls

Controls are ensured that the data is accurate and complete. Controls usually take the form of verification by people and validation by software.

**Output Design**

The output design is files updating. This involves adding new records, modifying or deleting exiting records. The problem of input design is to think a way of getting source data into the computer’s memory, where it can be processed. We must be careful to include controls along the way to ensure that, as far as possible, the data that ends up in memory is accurate. Inaccurate data cannot be processed to produce reliable or valid results.

The most important thing about any system s what produces is. A system is judged to be a success or failure depending on whether it’s produced is useful or not. So it is critical that we first specify what is required from the system. Once this has been done, we can concentrate on what is required to produce this output.

On decision, this needs to be made, which medium to use for a particular output. The main media are;

* Print, Used for reports and for a permanent listing of the file contents.
* Video display used for temporary output, usually responses to queries.
* Disk, used for storing data files, these files normally used for output and input.

Other factors consider to designing output are usage, quality and cost. These three factors are closely related. Output is sent can be divided into two ways internal and external. Internal usage refers to use by employers within the organization whereas external output is designed for people outside the organization.

**SYSTEM TESTING**

**INTRODUCTION TO SYSTEM TESTING**

Software testing is a critical element of software quality assurance and represents the ultimate reviews of specification, design and coding. Testing presents an interesting anomaly for the software. Testing is vital to the success of the system. Errors can be injected at any stage during development. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved.

During testing, the program to be tested is executed with set of test data and the output of the program for the test data is evaluated to determine if the program is performing as expected. A series of testing are performed for the proposed system before the system is ready for user acceptance testing. Another reason for system testing is its utility as a user-oriented vehicle before implementation.

The types of testing techniques are:

**BLACK BOX TESTING**

Black box testing methods focus on the functional requirements of the software. That is, Black box testing enables the software engineer to derive sets of input conditions that willfully exercise all functional requirements for a program. Black box testing attempts to find errors in the following categories

* Incorrect or missing function.
* Interface errors.
* Errors in data structures or external database access.
* Behavior or performance errors.
* Initialization and termination errors.

The various testing performed in “Desert Eagle :An Intelligent Tracking Mechanism For Kitkat” are unit testing and integration testing.

**WHITE BOX TESTING**

White box testing is a test case design method that causes the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that

* Guarantee that all independent paths within a module have been exercised atleast once.
* Exercise all logical decisions on their true and false sides.
* Execute all loops at their boundaries and within their operational bounds.
* Exercise internal data structures to ensure their validity.

**UNIT TESTING**

Unit testing focuses verification effort on the smallest unit of the software design, the module this is known as module testing. Since the proposed system has modules the testing is individually performed on each module.

Unit testing first focuses first on the modules in the proposed system to locate errors. This enables to detect errors in the coding and logic that are contained within that module alone. In unit testing step each module has to be checked separately. In our system we had done this testing in different modules such as student details, course details, internal grades etc.

**INTEGRATION TESTING**

Data can be test across an interface, one module can have adverse effect on another, sub function when combined may not produced the desired function. Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated within the interface.

The objective is to take unit tested modules and built a program structure that has been dictated by design. All modules are combined in this testing step. The entire program is tested as a whole. Correction is difficult at this stage because the isolation of causes is complicated by the vast expense of the program.

**IMPLEMENTATION AND MAINTENANCE**

Implementation includes all those activities that take place to convert from the old system to the new. The old system consists of manual operations, which is operated in a very different manner from the proposed new system. A proper implementation is essential to provide a reliable system to meet the requirements of the organization. An improper installation may affect the success of the computerized system.

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users, that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation, of change over methods. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out, discussions made regarding the equipment and the additional equipment has to be acquired to implement the new system.

A software implementation method is a systematically structured approach to effectively integrate software based service or component into the workflow of an organizational structure or an individual end-user. This entry focuses on the process modeling (Process Modeling), a process model is a description of a process at the type level, side of the implementation of large product software, using the implementation of Enterprise Resource Planning systems as the main example to elaborate on. A product software implementation method is a blueprint to get users and/or organizations running with a specific software product. The method is a set of rules and views to cope with the most common issues that occur when implementing a software product: business alignment from the organizational view and acceptance from the human view.

The implementation phase of the software development is concerned with translating design specification into source code. The user tests the developed system and changes are made according to their needs. Our system has been successfully implemented. Before implementation several tests have been conducted to ensure that no errors are encountered during the operation. The implementation phase ends with an evaluation of the system after placing into the operation for a period of time. Implementation is the third phase of the system process.

Implementation is the final and the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if it is found to be working according to the specification.

**Implementation Plan**

The implementation plan includes a description of all the activities that must occur to implement the new system and to put it into operation. It identifies the personnel responsible for the activities and prepares s a time chart for implementing the system. The implementation plan consists of the following steps:

* List all files required for implementation
* Identify all data required to build new files during the implementation.
* List all new documents and procedures that go into the new system.

The implementation plan should anticipate possible problems and must able to deal with them. The usual problems may be missing documents, mixed data formats between current and files, errors in data translation, missing data etc.

**Maintenance**

Maintanance involves the software industry captive, typing up system resources. It means restoring something to its original condition. Maintanance follows conversion to extend that changes are necessary to maintain satisfactory operations relative to changes in the user’s environment. Maintanance often includes minor enhancements or corrections to problems that surface in the systems operation.Maintanance is also done based on fixing the problems reported, changing the interface with other software or hardware enhancing the software. Maintenance error, updating documentation and test data, and upgrading user support. Maintenance is continued till the product is reengineered or deployed to another platform reported, changing the interface with other software or hardware enhancing the software.

Any system developed should be secured and protected against possible hazards. Security measures are provided to prevent unauthorized access of the database at various levels. Password protection and simple procedures to prevent the unauthorized access are provided to the users. The system allows the user to enter the system only through proper username and password. System maintenance is the modification of a software product after delivery to correct faults, to improve performance or other attributes. This section describes the six software maintenance processes as:

* The implementation processes contains software preparation and transition activities, such as the conception and creation of the maintenance plan, the preparation for handling problems identified during development, and the follow-up on product configuration management.
* The problem and modification analysis process, which is executed once the application has become the responsibility of the maintenance group. The maintenance programmer must analyze each request, confirm it (by reproducing the situation) and check its validity, investigate it and propose a solution, document the request and the solution proposal, and, finally, obtain all the required authorizations to apply the modifications.
* The process considering the implementation of the modification itself.
* The process acceptance of the modification, by confirming the modified work with the individual who submitted the request in order to make sure the modification provided a solution.
* The migration process is exceptional, and is not part of daily maintenance tasks. If the software must be ported to another platform without any change in functionality, this process will be used and a maintenance project team is likely to be assigned to this task.
* Finally, the last maintenance process, also an event which does not occur on a daily basis, is the retirement of a piece of software.

**DATA FLOW DIAGRAMS**

**LEVEL 0**

AnyCab

Request Response

User

Driver

Response Request

Request Response

Admin

Level 1

Customer

Any Cab

Login

validation

Registration

User

Cabbie Details

Feedback

**LEVEL 2**

**Level 2.1**

Registration

Any Cab

Login

validation

Registration

Customer

Cabbie Details

Feedback

**Level 2.2**

Registration

Any Cab

Login

validation

Registration

Cabbie

Cabbie Details

**Level 2.3**

View Cabbies

Any Cab

login

validation

Registration

Admin

Cabbie Details

Feedback

**5.TABLE DESIGN**

**REGISTRATION:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | **Constraints** |
| Name | varchar | Name |  |
| User name | varchar | User name |  |
| Password | varchar | Pass word |  |
| Mob no: | Int | Mobile number |  |
| e-mail | varchar | E-mail address |  |

**Cabbie**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | **Constraints** |
| Id | int | Auto increment | Primary key |
| Vehicle Number | Int | Mobile number |  |
| Name | varchar |  |  |
| Seat capacity | int |  |  |
| Mobile No | int |  |  |
| status | boolean | Available or not |  |

**Feed Backs**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Type** | **Description** | **Constraints** |
| slno | int | Auto increment | Primary key |
| date | date | optional |  |
| details | varchar | Remainder detail |  |