

## 1.INTRODUCTION

A major problem in mobile search is that the interactions between the users and search engines are limited by the small form factors of the mobile devices. As a result, mobile users tend to submit shorter, hence, more ambiguous queries compared to their web search counterparts. In order to return highly relevant results to the users, mobile search engines must be able to profile the users' interests and personalize the search results according to the users' profiles. A practical approach to capturing a user's interests for personalization is to analyze the user's click through data Leung, et. al., developed a search engine personalization method based on users' concept preferences and showed that it is more effective than methods that are based on page preferences. However, most of the previous work assumed that all concepts are of the same type. Observing the need for different types of concepts, we present in this paper a personalized mobile search engine, PMSE, which represents different types of concepts in different ontologies. In particular, recognizing the importance of location information in mobile search, we separate concepts into location concepts and content concepts.

There are easy ways to support queries that combine spatial and text features. For example, for the above query, we could first fetch all the restaurants whose menus contain the set of keywords {steak, spaghetti, brandy}, and then from the retrieved restaurants, find the nearest one. Similarly, one could also do it reversely by targeting first the spatial conditions—browse all the restaurants in ascending order of their distances to the query point until encountering one whose menu has all the keywords. The major drawback of these straightforward approaches is that they will fail to provide real time answers on difficult inputs. A typical example is that the real nearest neighbor lies quite far away from the query point, while all the closer neighbors are missing at least one of the query keywords. Spatial queries with keywords have not been extensively explored. In the past years, the community has sparked enthusiasm in studying keyword search in relational databases. It is until recently that attention was diverted to multidimensional data. The best method to date for nearest neighbor search with keywords is due to Felipe et al. They nicely integrate two well-known concepts: R-tree, a popular spatial index, and signature file, an effective method for keyword-based document retrieval. By doing so they develop a structure called the IR2-tree, which has the strengths of both R-trees and signature files. Like R-trees, the IR2-tree preserves

objects' spatial proximity, which is the key to solving spatial queries efficiently. On the other hand, like signature files, the IR2-tree is able to filter a considerable portion of the objects that do not contain all the query keywords, thus significantly reducing the number of objects to be examined.

## 2.SYSTEM REQUIREMENTS

### HARDWARE INTERFACE

<b>Processor</b>	Pentium –III
<b>Speed</b>	1.1 Ghz
<b>RAM</b>	256 MB(min)
<b>Hard Disk</b>	20 GB

### SOFTWARE REQUIREMENTS

➤ Operating System	Windows
➤ Application Server	Tomcat5.0/6.X
➤ Front End	HTML, Java, Jsp
➤ Scripts	JavaScript.
➤ Server side Script	Java Server Pages.
➤ Database	MySql
➤ Database Connectivity	JDBC.

### DOCUMENT CONVENTIONS

The following are the conventions and the acronyms that are used in this document.

Styles and Formatting:

- Aligned At : Left of the page
- Text Font : Times New Roman, 12
- Heading Font : Times New Roman (Bold), 16
- Sub-heading Font : Times New Roman (Bold), 14
- Margins : Top 1inch,  
: Left 1.5 inch  
: Right 1inch  
: Bottom 1 inch

### **3.SOFTWARE OVERVIEW**

#### **3.1.JAVA**

Java is a programming language created by James Gosling from Sun Microsystems in 1991. The first publicly available version of Java (Java 1.0) was released in 1995. Over time new enhanced versions of Java have been released. The current version of Java is Java 1.7 which is also known as *Java 7*.

##### **3.1.1.DEVELOPMENT PROCESS WITH JAVA**

The programmer writes Java source code in a text editor which supports plain text. Normally the programmer uses an *Integrated Development Environment* (IDE) for programming. An IDE supports the programmer in the task of writing code, e.g. it provides auto-formatting of the source code, highlighting of the important keywords, etc.

At some point the programmer (or the IDE) calls the Java compiler (javac). The Java compiler creates the *byte code* instructions. . These instructions are stored in .class files and can be executed by the Java Virtual Machine.

##### **3.1.2.J2EE**

Java<sup>2</sup> Platform *Enterprise Edition*. J2EE is a platform-independent, Java-centric environment from Sun for developing, building and deploying Web-based enterprise applications online. The J2EE platform consists of a set of services, APIs, and protocols that provide the functionality for developing multitier, Web-based applications.

##### **KEY FEATURES AND SERVICES OF J2EE:**

- At the client tier, J2EE supports pure HTML, as well as Java applets or applications. It relies on Java Server Pages and servlet code to create HTML or other formatted data for the client.

- Enterprise JavaBeans (EJBs) provide another layer where the platform's logic is stored. An EJB server provides functions such as threading, concurrency, security and memory management. These services are transparent to the author.
- Java Database Connectivity (JDBC), which is the Java equivalent to ODBC, is the standard interface for Java databases.
- The Java servlet API enhances consistency for developers without requiring a graphical user interface.

### 3.1.3.JDBC

JDBC stands for **Java Database Connectivity**, which is a standard Java API for database-independent connectivity between the Java programming language and a wide range of databases.

The JDBC library includes APIs for each of the tasks commonly associated with database usage:

- Making a connection to a database
- Creating SQL or MySQL statements
- Executing that SQL or MySQL queries in the database
- Viewing & Modifying the resulting records

Fundamentally, JDBC is a specification that provides a complete set of interfaces that allows for portable access to an underlying database. Java can be used to write different types of executable code pieces, such as:

- Java Applications
- Java Applets
- Java Servlets
- Java ServerPages (JSPs)
- Enterprise JavaBeans (EJBs)

All of these different executable applications are able to use a JDBC driver to access a database and take advantage of the stored data. JDBC provides the same capabilities as ODBC, allowing Java programs to contain database-independent code.

### **JDBC ARCHITECTURE:**

The JDBC API supports both two-tier and three-tier processing models for database access but in general JDBC Architecture consists of two layers:

**JDBC API:** This provides the application-to-JDBC Manager connection.

### **JDBC DRIVER API:**

This supports the JDBC Manager-to-Driver Connection. The JDBC API uses a driver manager and database-specific drivers to provide transparent connectivity to heterogeneous databases. The JDBC driver manager ensures that the correct driver is used to access each data source. The driver manager is capable of supporting multiple concurrent drivers connected to multiple heterogeneous databases.

Following is the architectural diagram, which shows the location of the driver manager with respect to the JDBC drivers and the Java application

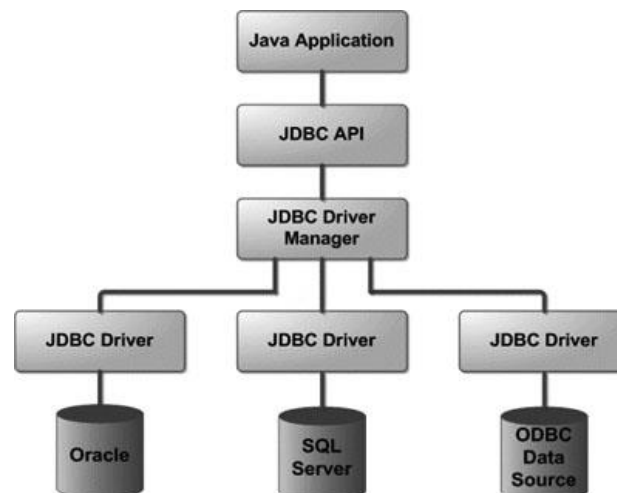


Fig:3.1.3(JDBC architecture)

## COMMON JDBC COMPONENTS:

The JDBC API provides the following interfaces and classes:

- **Driver Manager:** This class manages a list of database drivers. Matches connection requests from the java application with the proper database driver using communication sub-protocol. The first driver that recognizes a certain sub-protocol under JDBC will be used to establish a database Connection.
- **Driver:** This interface handles the communications with the database server. You will interact directly with Driver objects very rarely. Instead, you use Driver Manager objects, which Manages objects of this type. It also abstracts the details associated with working with Driver objects.
- **Connection:** This interface with all methods for contacting a database. The connection object represents communication context, i.e., all communication with database is through connection object only.
- **Statement:** You use objects created from this interface to submit the SQL statements to the database. Some derived interfaces accept parameters in addition to executing stored procedures.
- **Result Set:** These objects hold data retrieved from a database after you execute an SQL query using Statement objects. It acts as an iterator to allow you to move through its data.
- **SQL Exception:** This class handles any errors that occur in a database application.

## WHAT IS JDBC DRIVER?

JDBC drivers implement the defined interfaces in the JDBC API for interacting with your database server. For example, using JDBC drivers enable you to open database connections and to interact with it by sending SQL or database commands then receiving results with Java.

The *Java.sql* package that ships with JDK contains various classes with their behaviors defined and their actual implementations are done in third-party drivers. A third party vendor implements the *java.sql.Driver* interface in their database driver.

### 3.1.4.JDBC DRIVERS TYPES:

JDBC driver implementations vary because of the wide variety of operating systems and hardware platforms in which Java operates. Sun has divided the implementation types into four categories, Types 1, 2, 3, and 4, which is explained below:

#### TYPE 1: JDBC-ODBC BRIDGE DRIVER:

In a Type 1 driver, a JDBC bridge is used to access ODBC drivers installed on each client machine. Using ODBC requires configuring on your system a Data Source Name (DSN) that represents the target database.

When Java first came out, this was a useful driver because most databases only supported ODBC access but now this type of driver is recommended only for experimental use or when no other alternative is available.

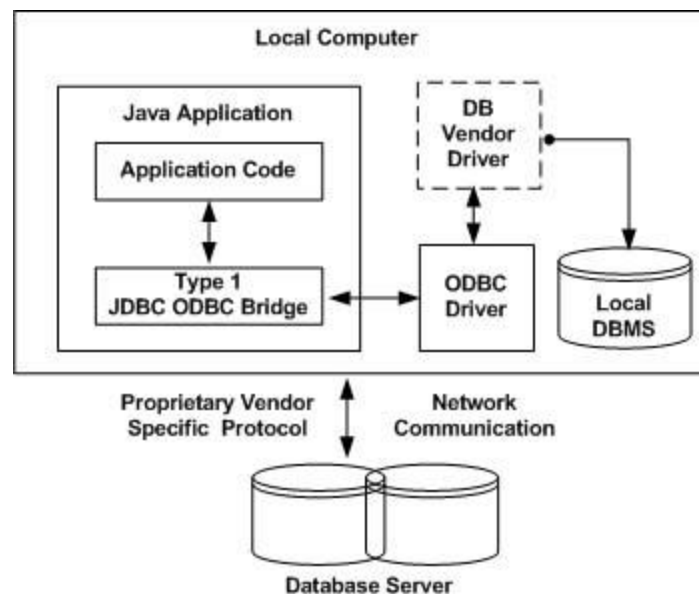


Fig:3.1.4(JDBC-ODBC bridge driver)

The JDBC-ODBC bridge that comes with JDK 1.2 is a good example of this kind of driver.



**TYPE 2: JDBC-NATIVE API:**

In a Type 2 driver, JDBC API calls are converted into native C/C++ API calls which are unique to the database. These drivers typically provided by the database vendors and used in the same manner as the JDBC-ODBC Bridge, the vendor-specific driver must be installed on each client machine.

If we change the Database we have to change the native API as it is specific to a database and they are mostly obsolete now but you may realize some speed increase with a Type 2 driver, because it eliminates ODBC's overhead.

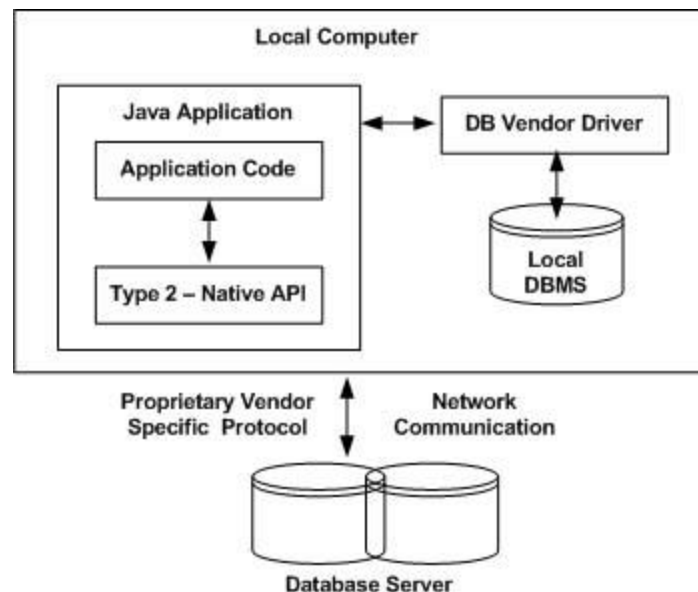


Fig:3.1.4(JDBC-Native API)

**TYPE 3: JDBC-NET PURE JAVA:**

In a Type 3 driver, a three-tier approach is used to accessing databases. The JDBC clients use standard network sockets to communicate with an middleware application server. The socket information is then translated by the middleware application server into the call format required by the DBMS, and forwarded to the database server.

This kind of driver is extremely flexible, since it requires no code installed on the client and a single driver can actually provide access to multiple databases.

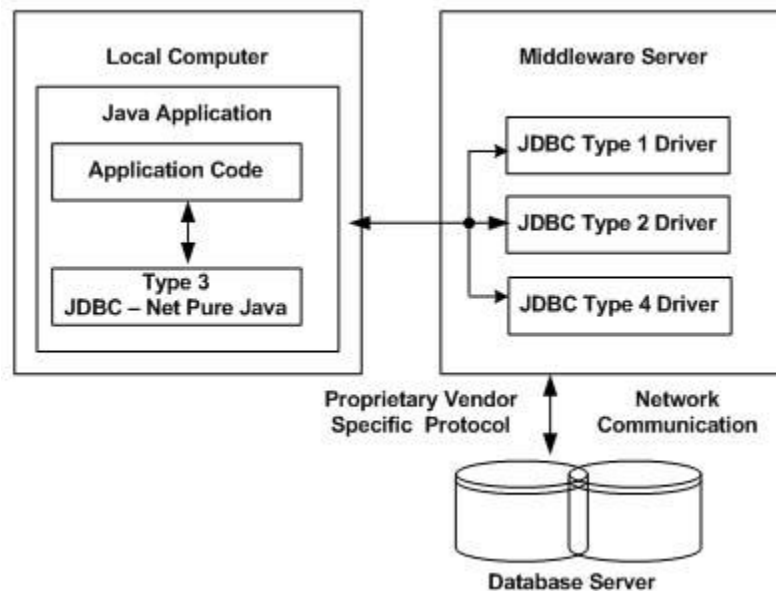


Fig:3.1.4(JDBC-net pure java)

You can think of the application server as a JDBC "proxy," meaning that it makes calls for the client application. As a result, you need some knowledge of the application server's configuration in order to effectively use this driver type.

Your application server might use a Type 1, 2, or 4 driver to communicate with the database, understanding the nuances will prove helpful.

#### **TYPE 4: 100% PURE JAVA:**

In a Type 4 driver, a pure Java-based driver that communicates directly with vendor's database through socket connection. This is the highest performance driver available for the database and is usually provided by the vendor itself. This kind of driver is extremely flexible, you don't need to install special software on the client or server. Further, these drivers can be downloaded dynamically.

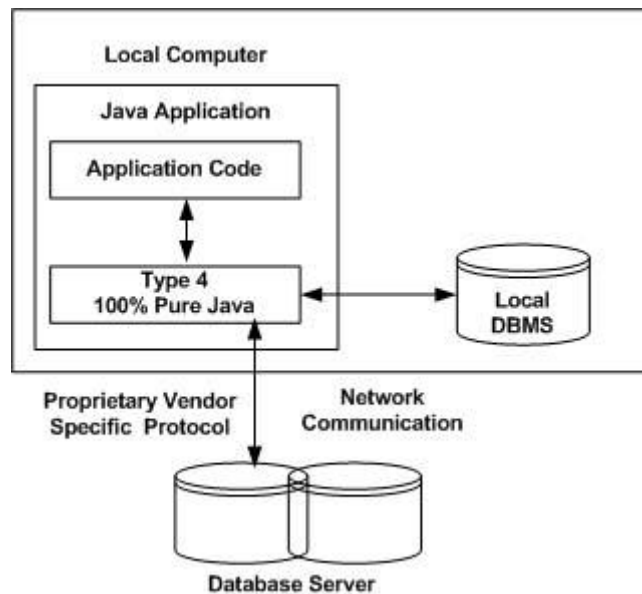


Fig:3.1.4(100% Pure java)

MySQL's Connector/J driver is a Type 4 driver. Because of the proprietary nature of their network protocols, database vendors usually supply type 4 drivers.

## JDBC DATABASE CONNECTIONS

After you've installed the appropriate driver, it's time to establish a database connection using JDBC.

The programming involved to establish a JDBC connection is fairly simple. Here are these simple four steps:

- **Import JDBC Packages:** Add **import** statements to your Java program to import required classes in your Java code.
- **Register JDBC Driver:** This step causes the JVM to load the desired driver implementation into memory so it can fulfill your JDBC requests.
- **Database URL Formulation:** This is to create a properly formatted address that points to the database to which you wish to connect.
- **Create Connection Object:** Finally, code a call to the *DriverManager* object's *getConnection( )* method to establish actual database connection.

### IMPORT JDBC PACKAGES:

The Importstatements tell the Java compiler where to find the classes you reference in your code and are placed at the very beginning of your source code.

To use the standard JDBC package, which allows you to select, insert, update, and delete data in SQL tables, add the following *imports* to your source code:

```
import java.sql.* ; // for standard JDBC programs  
import java.math.* ; // for BigDecimal and BigInteger support
```

### Register JDBC Driver:

You must register the your driver in your program before you use it. Registering the driver is the process by which the Oracle driver's class file is loaded into memory so it can be utilized as an implementation of the JDBC interfaces.

You need to do this registration only once in your program. You can register a driver in one of two ways.

### Approach- DriverManager.registerDriver():

The second approach you can use to register a driver is to use the static **DriverManager.registerDriver()**method.

You should use the registerDriver() method if you are using a non-JDK compliant JVM, such as the one provided by Microsoft.

The following example uses registerDriver() to register the Oracle driver:

```
try {  
  
    Driver myDriver = new oracle.jdbc.driver.OracleDriver();  
  
    DriverManager.registerDriver(myDriver );  
  
}  
  
catch(ClassNotFoundException ex) {  
  
    System.out.println("Error: unable to load driver class!");  
  
    System.exit(1);  
  
}
```

### **3.1.5.Database URL Formulation:**

After you've loaded the driver, you can establish a connection using the **DriverManager.getConnection()** method. For easy reference, let me list the three overloaded **DriverManager.getConnection()** methods:

- `getConnection(String url)`
- `getConnection(String url, Properties prop)`
- `getConnection(String url, String user, String password)`

Here each form requires a database **URL**. A database URL is an address that points to your database.

Formulating a database URL is where most of the problems associated with establishing a connection occur.

Following table lists down popular JDBC driver names and database URL.

RDBMS	JDBC driver name	URL format
MySQL	com.mysql.jdbc.Driver	<b>jdbc:mysql://</b> hostname/ databaseName
ORACLE	oracle.jdbc.driver.OracleDriver	<b>jdbc:oracle:thin:</b> @hostname:port Number:databaseName
DB2	COM.ibm.db2.jdbc.net.DB2Driver	<b>jdbc:db2:</b> hostname:port Number/databaseName
Sybase	com.sybase.jdbc.SybDriver	<b>jdbc:sybase:Tds:</b> hostname: port Number/databaseName

Table:3.1.5(Database url formulation)

All the highlighted part in URL format is static and you need to change only remaining part as per your database setup.

### Create Connection Object:

### Using a database URL with a username and password:

I listed down three forms of [DriverManager.getConnection\(\)](#) method to create a connection object. The most commonly used form of getConnection() requires you to pass a database URL, a *username*, and a *password*:

Assuming you are using Oracle's **thin** driver, you'll specify a host:port:databaseName value for the database portion of the URL.

If you have a host at TCP/IP address 192.0.0.1 with a host name of amrood, and your Oracle listener is configured to listen on port 1521, and your database name is EMP, then complete database URL would be:

```
jdbc:oracle:thin:@amrood:1521:EMP
```

Now you have to call getConnection() method with appropriate username and password to get a [Connection](#) object as follows:

```
String URL = "jdbc:oracle:thin:@amrood:1521:EMP";
```

```
String USER = "username";
```

```
String PASS = "password"
```

```
Connection conn = DriverManager.getConnection(URL, USER, PASS);
```

### **Closing JDBC connections:**

At the end of your JDBC program, it is required explicitly close all the connections to the database to end each database session. However, if you forget, Java's garbage collector will close the connection when it cleans up stale objects.

Relying on garbage collection, especially in database programming, is very poor programming practice. You should make a habit of always closing the connection with the close() method associated with connection object.

To ensure that a connection is closed, you could provide a finally block in your code. A *finally* block always executes, regardless if an exception occurs or not.

To close above opened connection you should call close() method as follows:

```
conn.close();
```

Explicitly closing a connection conserves DBMS resources, which will make your database administrator happy.

## **3.2. HYPER TEXT MARKUP LANGUAGE**

HTML stands for "Hypertext Markup Language". HTML is a SGML (Standard Generalized Markup Language) application widely used to create web pages. It is basically a formatting language and not a programming language. HTML is a language that is easy to write, easy to understand and highly portable. HTML is not a compiled language and is directly interpreted by a browser. HTML is the set of instructions. Each instruction is called as an element

or Markup. It is used to structure and format documents for presentation on the web. HTML enhances ASCII files with markup tags that permit the display of a variety of fonts, images, and highlighting options. It also designates structural elements such as headers, lists, and paragraphs, and provides hypertext links to other documents on the Internet.

In plain English, for those of us who are not programming wizards, it is the language our web browser understands, so that it can display what the author of the page wanted us to see... and we can do some really neat stuff with it.

### **3.2.1.JAVA SCRIPT**

Java Script is Netscape's cross-platform, object-based scripting language for client server application. JavaScript is mainly used as a client side scripting language. This means that JavaScript code is written into an HTML page. When a user requests an HTML page with JavaScript in it, the script is sent to the browser and it's up to the browser to do something with it. JavaScript can be used in other contexts than a Web browser. Netscape created server-side JavaScript as a CGI-language that can do roughly the same as Perl or ASP.

Fortunately most browsers can handle JavaScript nowadays, but of course some browsers do not support some bits of script.

### **APACHE TOMCAT SERVER**

Apache Tomcat is a web container developed at the Apache Software Foundation (ASF). Tomcat implements the servlet and the Java Server Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are normally XML-formatted. Tomcat includes its own HTTP server internally.

### **3.2.2.JAVA SERVER PAGES**



## **INTRODUCTION:**

Jsp technology enables you to mix regular static html with dynamically generated content from servlets. Separating the static html from the dynamic content provides a number of benefits over servlets alone.

## **WHY USE JSP:**

Jsp is easy to learn and allows developers to quickly produce web sites and application in an open and standard way. Jsp is based on java, an object-oriented language. Jsp offers a robust platform for web development.

Main reasons to Jsp:

- a. Multi platform
- b. Component reuse by using java beans and Ejb
- c. Advantages if java

We can take one Jsp file and move it to another platform, web server or Jsp servlet engine.

## **3.2.3.JSP COMPARED TO ASP:**

Jsp and Asp are fairly similar in the functionality that they provide. Jsp may have slightly higher learning curve. Both allow embedded code in an html page, session variables Platform i.e., NT, JSP can operate on any platform that conforms to the J2EE specification. Jspallow component reuse by using JavaBeans and Ejbs. Asp provides the use of Com/activeX controls.

## **3.2.4.JSP COMPARED TO SERVLETS:**

A servlet is java class that provides special server side service. It is hard to write HTML code in servletsIn servlets you need to have lots of println statement to generate HTML.

## **3.3.SERVLETS**

### **3.3.1.INTRODUCTION:**

The Java web server is Java Soft own web Server. The Java web server is just a part of a larger framework, intended to provide you not just with a web server, but also with tools. To build customized network servers for any Internet or Intranet client/server system. Servlets are to a web server, how applets are to the browser.

### **3.3.2. ABOUT SERVLETS:**

Servlets provide a Java-based solution used to address the problems currently associated with doing server-side programming, including inextensible scripting solutions, platform-specific APIs, and incomplete interfaces. Servlets are objects that conform to a specific interface that can be plugged into a Java-based server.

Servlets are to the server-side what applets are to the client-side - object byte codes that can be dynamically loaded off the net. They differ from applets in that they are faceless objects (without graphics or a GUI component). They serve as platform independent, dynamically loadable, plug able helper byte code objects on the server side that can be used to dynamically extend server-side functionality. For example, an HTTP Servlet can be used to generate dynamic HTML content. When you use Servlets to do dynamic content you get the following advantages:

- They are faster and cleaner than CGI scripts
- They use a standard API (the Servlet API)
- They provide all the advantages of Java (run on a variety of servers without needing to be rewritten).

### **3.3.3.ATTRACTIVENESS OF SERVLETS:**

There are many features of Servlets that make them easy and attractive to use. These include:

- Easily configured using the GUI-based Admin tool
- Can be loaded and invoked from a local disk or remotely across the network.
- Can be linked together, or chained, so that one Servlet can call another Servlet or several Servlets in sequence.

- Can be called dynamically from within HTML pages, using server-side include tags.
- Are secure - even when downloading across the network, the Servlet security model and Servlet sandbox protect your system from unfriendly behavior.

### **3.3.4. ADVANTAGES OF THE SERVLET API:**

One of the great advantages of the Servlet API is protocol independence. It assumes nothing about:

- The protocol being used to transmit on the net
- How it is loaded
- The server environment it will be running in

These qualities are important, because it allows the Servlet API to be embedded in many different kinds of servers. There are other advantages to the Servlet API as well. These include:

- a. It's extensible - you can inherit all your functionality from the base classes made available to you.
- b. It's simple, small, and easy to use.

### **3.3.5.FEATURES OF SERVLETS:**

Servlets are persistent. Servlet are loaded only by the web server and can maintain services between requests.

- Servlets are fast. Since Servlets only need to be loaded once, they offer much better performance over their CGI counterparts.
- Servlets are platform independent.
- Servlets are extensible. Java is a robust, object-oriented programming language, which easily can be extended to suit your needs
- Servlets are secure.

### **3.3.6.LOADING SERVLETS:**

Servlets can be loaded from three places:

From a directory that is on the CLASSPATH. The CLASSPATH of the Java Web Server includes service root/classes/which is where the system classes reside. From the <SERVICE\_ROOT/Servlets/directory> this is \*not\* in the server's class path. A class loader is used to create Servlets from this directory. New Servlets can be added - existing Servlets can be recompiled and the server will notice these changes from a remote location. For this a code base like http://nine.eng/classes/foo/ is required in addition to the Servlet class name. Refer to the admin GUI docs on Servlet section to see how to set this up.

Loading Remote SERVLETS:

Remote Servlets can be loaded by:

- Configuring the Admin Tool to setup automatic loading of remote Servlets
- Setting up server side include tags in .html files
- Defining a filter chain configuration.

### **3.3.7.INVOKING SERVLETS:**

A Servlet invoker is a Servlet that invokes the "service" method on a named Servlet. If the Servlet is not loaded in the server, then the invoker first loads the Servlet (either from local disk or from the network) and then invokes the "service" method. Also like applets, local Servlets in the server can be identified by just the class name. In other words, if a Servlet name is not absolute, it is treated as local.

A client can invoke Servlets in the following ways:

- The client can ask for a document that is served by the Servlet.
- The client (browser) can invoke the Servlet directly using a URL, once it has been mapped using the Servlet aliases section of the admin GUI.
- The Servlet can be invoked through server side include tags.
- The Servlet can be invoked by placing it in the Servlets / directory.

## **4.SYSTEM ANALYSIS**

### **4.1.THE STUDY OF THE SYSTEM:**

- To conduct studies and analyses of an operational and technological nature, and
- To promote the exchange and development of methods and tools for operational analysis as applied to defense problems.

#### **4.1.1.ARCHITECTURE FLOW:**

Below architecture diagram represents mainly flow of request from the users to database through servers. In this scenario overall system is designed in three tiers separately using three layers called presentation layer, business layer, data link layer. This project was developed using 3-tier architecture.

#### **4.1.2. 3-TIER ARCHITECTURE:**

The three-tier software architecture (a three layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user.

#### **4.1.3. ADVANTAGES OF THREE-TIER:**

- Separates functionality from presentation.
- Clear separation - better understanding.

- Changes limited to well define components.
- Can be running on WWW

### **4.3.FEASIBILITY STUDY:**

Feasibility study is conducted once the problem is clearly understood. The feasibility study which is a high-level capsule version of the entire system analysis and design process. The objective is to determine whether the proposed system is feasible or not and it helps us to the minimum expense of how to solve the problem and to determine, if the Problem is worth solving. The following are the three important tests that have been carried out for feasibility study.

#### **4.3.1.TECHNICAL FEASIBILITY:**

In the technical feasibility study, one has to test whether the proposed system can be developed using existing technology or not. It is planned to implement the proposed system in JSP. The project entitled is technically feasible because of the following reasons.

- All necessary technology exists to develop the system.
- The existing system is so flexible that it can be developed further.

#### **4.3.2.ECONOMIC FEASIBILITY:**

As a part of this, the costs and benefits associated with the proposed systems are to be compared. The project is economically feasible only if tangible and intangible benefits outweigh the cost. We can say the proposed system is feasible based on the following grounds.

- The cost of developing the full system is reasonable.
- The cost of hardware and software for the application is less.

#### **4.3.3.OPERATIONAL FEASIBILITY:**

The project is operationally feasible because there is sufficient support from the project management and the users of the proposed system .Proposed system definitely does not harm and will not produce the bad results and no problem will arise after implementation of the system.

**USER-FRIENDLY:**

Customer will use the forms for their various transaction i.e. for adding new routes, viewing the routes details. Also the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

**RELIABILITY:**

The package will pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

**SECURITY:**

The web server and database server should be protected from hacking, virus etc.

**AVAILABILITY:**

This software will be available always.

**4.4. SDLC (Spiral Model):**

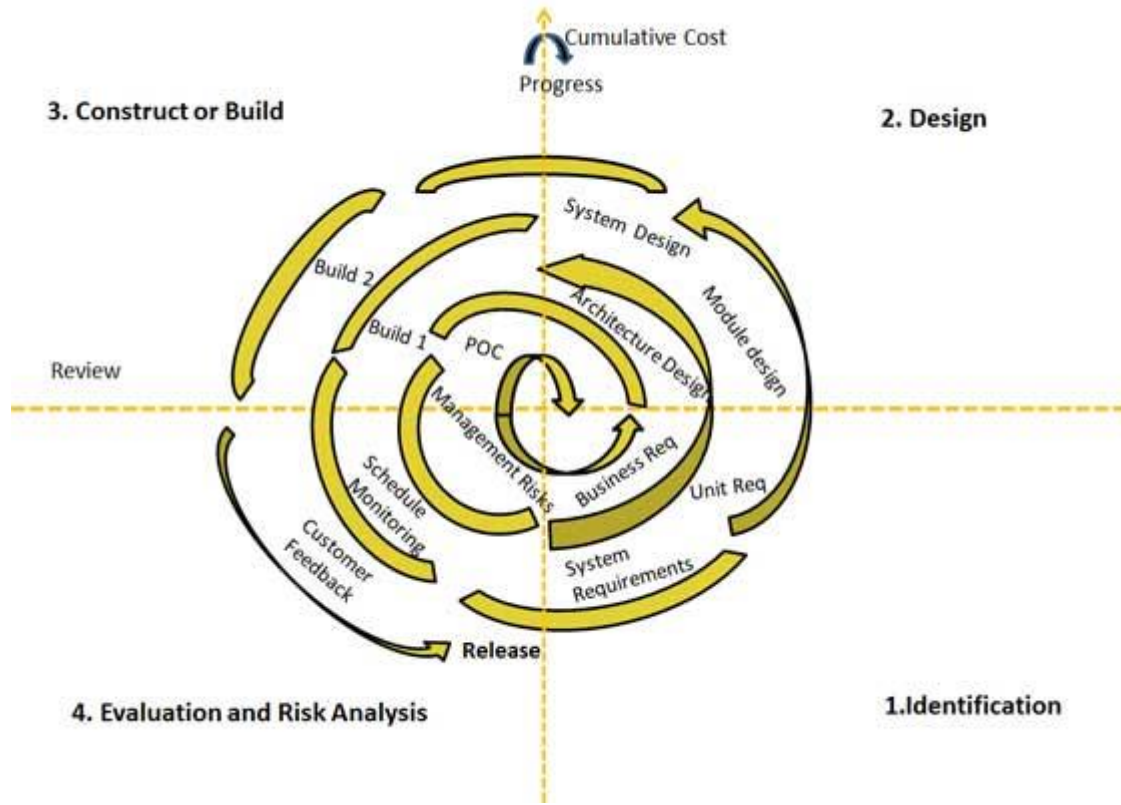


Fig:4.4(SDLC)

#### 4.4.1. STAGES OF SDLC:

- Requirement Gathering and Analysis
- Designing
- Coding
- Testing
- Deployment



#### **4.4.2. REQUIREMENTS DEFINITION STAGE AND ANALYSIS:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual Descriptionription.

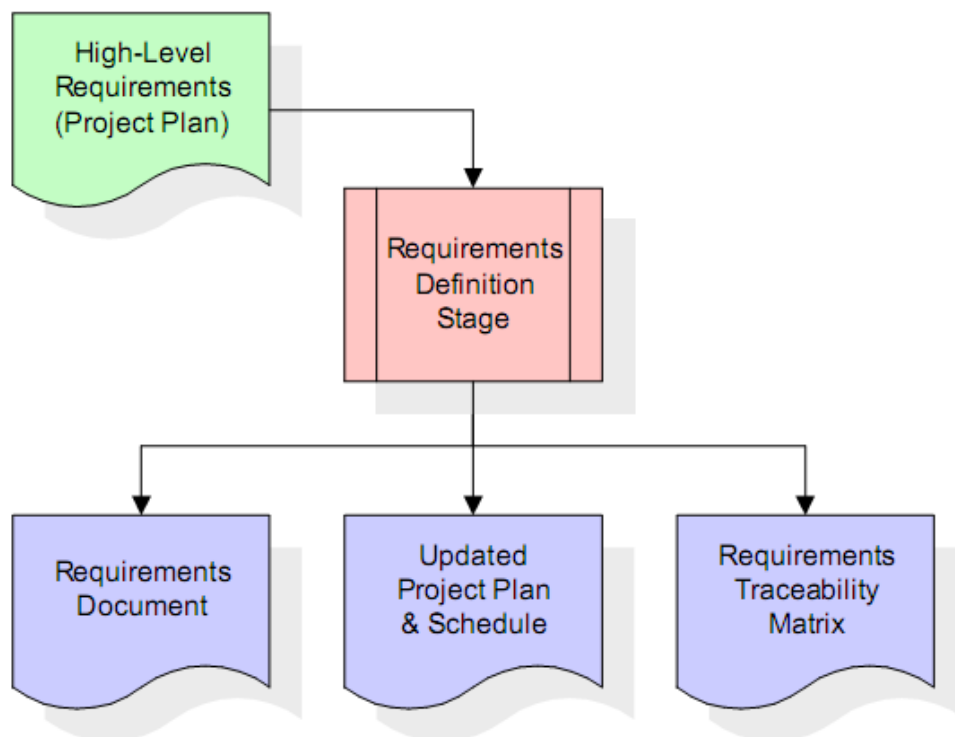


Fig:4.2.2(Requirements Definition Stage and Analysis)

These requirements are fully Descriptionribed in the primary deliverables for this stage. The Requirements Document and the Requirements Traceability Matrix (RTM). the requirements

document contains complete Descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document. The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term *requirements traceability*. The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

#### **4.4.3. DESIGN STAGE:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements Describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to Describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

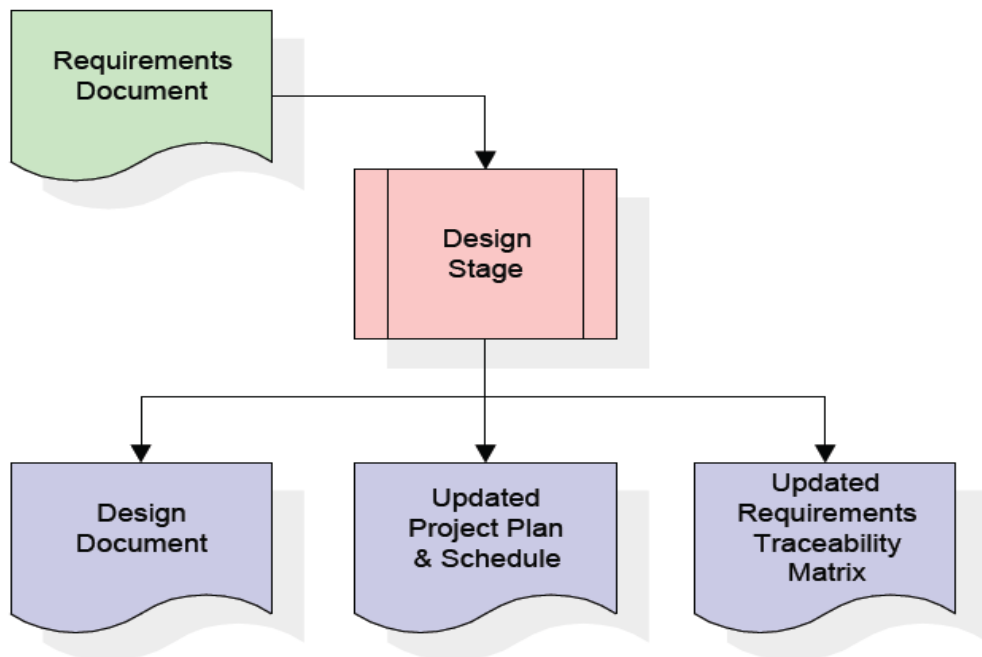


Fig:4.4.3(Design Stage)

When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

#### **4.4.4. DEVELOPMENT STAGE:**

The development stage takes as its primary input the design elements Descriptionribed in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

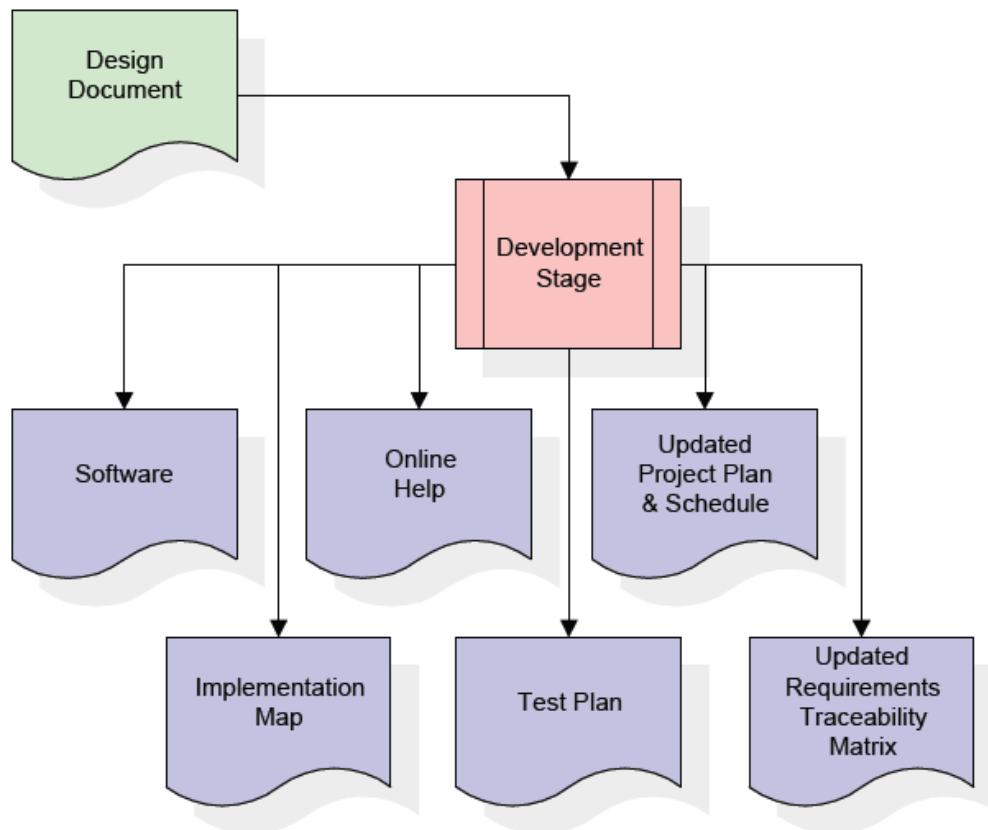


Fig:4.4.4(development stage)

The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that Descriptionribes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that Descriptionribes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

#### **4.4.5.INTEGRATION & TEST STAGE:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability.

During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

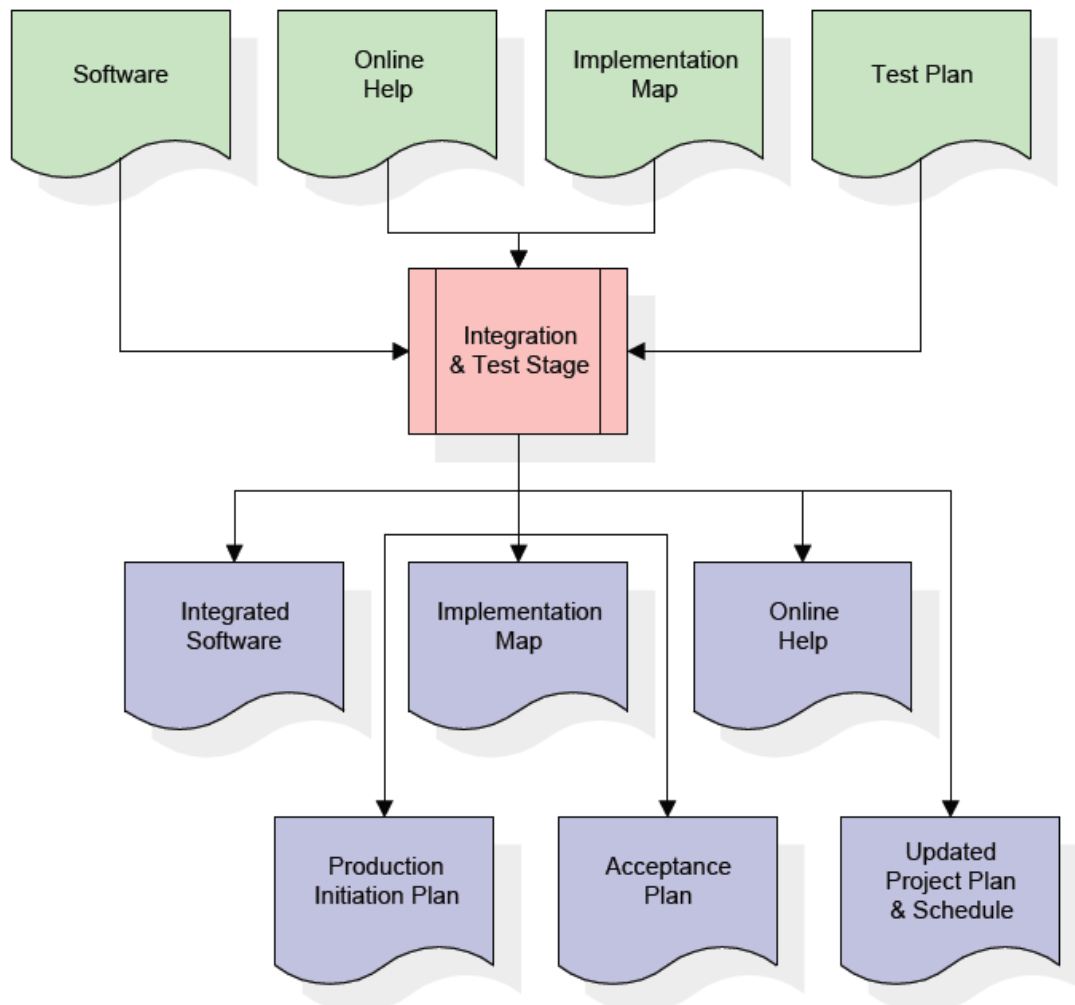


Fig:4.4.5(Integration & Test Stage)

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that Descriptionribes reference

data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

#### **4.4.6. INSTALLATION & ACCEPTANCE STAGE:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

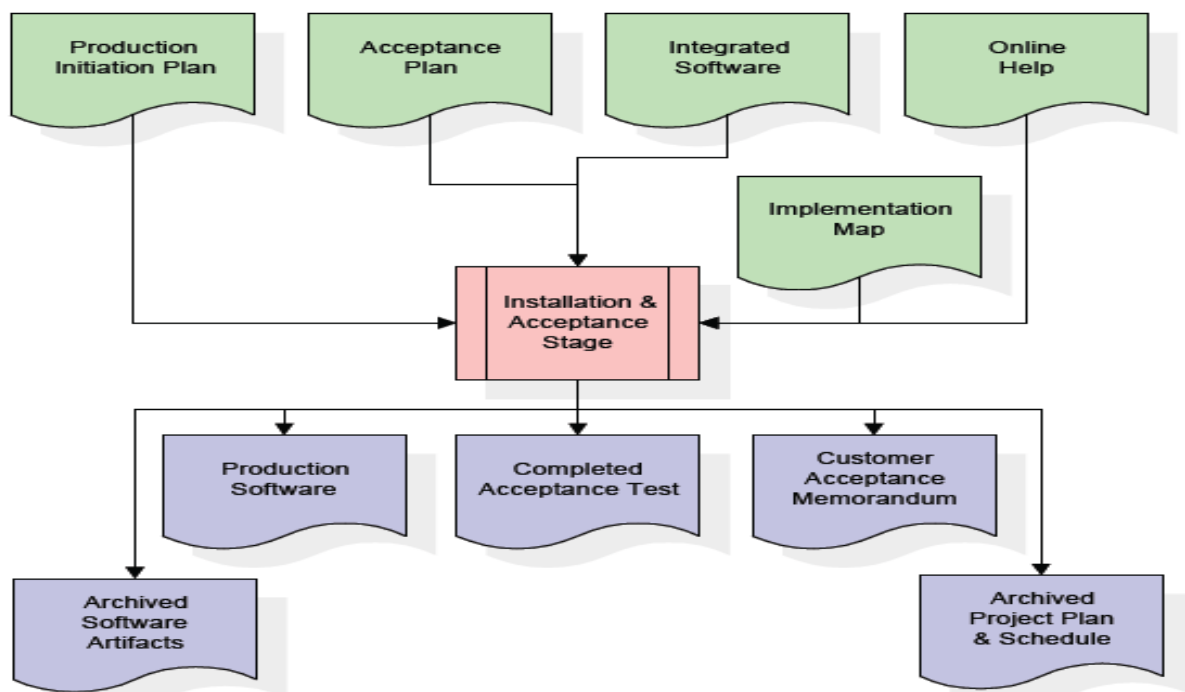


Fig:4.4.6(Installation & Acceptance Stage)

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the

software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

## 5.DESIGN

## **5.1.SYSTEM DESIGN:**

The System Design Document Descriptionribes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

## **5.2. INPUT DESIGN:**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

### **5.2.1. OBJECTIVES:**

Input Design is the process of converting a user-oriented Descriptionription of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.



When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

### **5.3. OUTPUT DESIGN:**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

- a. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
- b. Select methods for presenting information.
- c. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

## **6.SYSTEAM ARCHITECTURE**

### **6.1.UML DIAGRAMS:**

### 6.1.1.GLOBAL USE CASE DIAGRAMS:

Identification of actors:

**Actor:** Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:

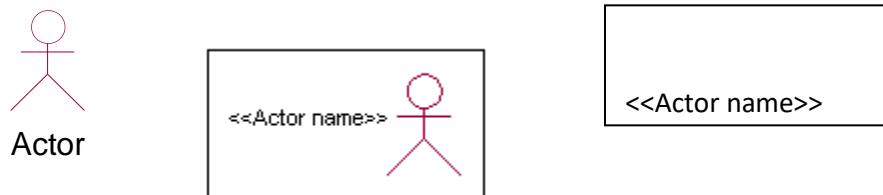


Fig:6.1.1(Global use case diagram)

An actor is someone or something that:

Interacts with or uses the system.

- Provides input to and receives information from the system.
- Is external to the system and has no control over the use cases.

Actors are discovered by examining:

- Who directly uses the system?
- Who is responsible for maintaining the system?
- External hardware used by the system.
- Other systems that need to interact with the system.

Questions to identify actors:

- Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
- Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
- Which external hardware or systems (if any) use the system to perform tasks?
- What problems does this application solve (that is, for whom)?
- And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

- SYSTEM ADMINISTRATOR**
- CUSTOMER**
- CUSTOMER CARE**

Identification of usecases:

**USECASE:** A use case can be Descriptionribed as a specific way of using the system from a user's (actor's) perspective.

### **GRAPHICAL REPRESENTATION:**



A more detailed Descriptionription might characterize a use case as:

- Pattern of behavior the system exhibits
- A sequence of related transactions performed by an actor and the system
- Delivering something of value to the actor

Use cases provide a means to:

- capture system requirements
- communicate with the end users and domain experts
- test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

- For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
- Name the use cases.
- Descriptionribe the use cases briefly by applying terms with which the user is familiar.

This makes the Descriptionription less ambiguous.

Questions to identify use cases:

- What are the tasks of each actor?
- Will any actor create, store, change, remove or read information in the system?
- What use case will store, change, remove or read this information?
- Will any actor need to inform the system about sudden external changes?
- Does any actor need to inform about certain occurrences in the system?
- What usecases will support and maintains the system?
- 

### **6.1.2. FLOW OF EVENTS:**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the

system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

- When and how the use case starts and ends
- Use case/actor interactions
- Data needed by the use case
- Normal sequence of events for the use case
- Alternate or exceptional flows

Construction of Usecase diagrams:

Use-case diagrams graphically depict system behavior (use cases). These diagrams present a high level view of how the system is used as viewed from an outsider's (actor's) perspective. A use-case diagram may depict all or some of the use cases of a system.

A use-case diagram can contain:

- actors ("things" outside the system)
- use cases (system boundaries identifying what the system should do)
- Interactions or relationships between actors and use cases in the system including the associations, dependencies, and generalizations.

## **1. COMMUNICATION:**

The communication relationship of an actor in a usecase is shown by connecting the actor symbol to the usecase symbol with a solid path. The actor is said to communicate with the usecase.

## **2. USES:**

A Uses relationship between the usecases is shown by generalization arrow from the usecase.

## **3. EXTENDS:**

The extend relationship is used when we have one usecase that is similar to another usecase but does a bit more. In essence it is like subclass.

## **6.1.3.SEQUENCE DIAGRAMS:**

A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next. Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.

There are two main differences between sequence and collaboration diagrams: sequence diagrams show time-based object interaction while collaboration diagrams show how objects associate with each other. A sequence diagram has two dimensions: typically, vertical placement represents time and horizontal placement represents different objects.

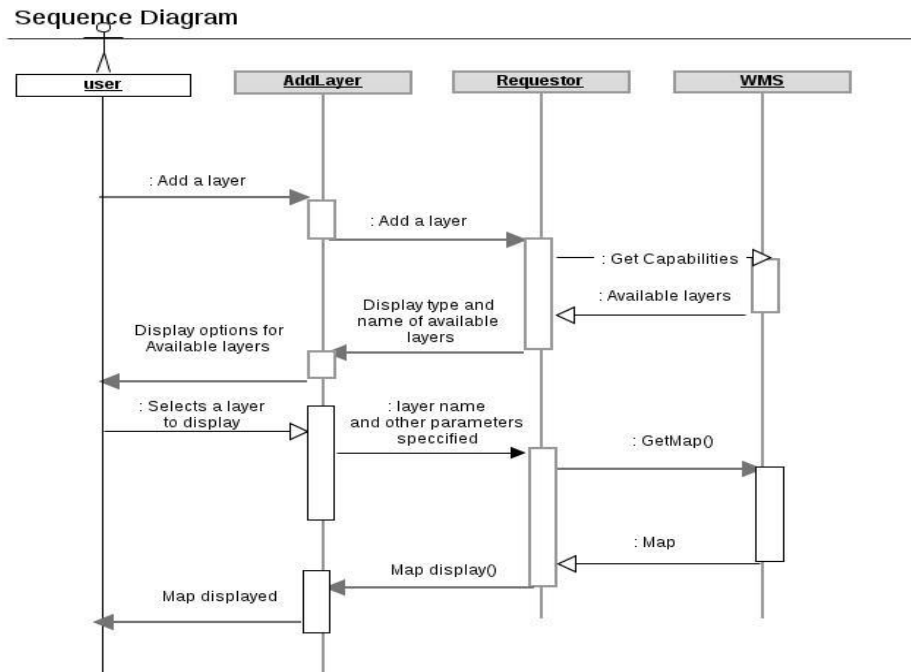


Fig:6.1.3.Sequence diagram

**OBJECT:**

An object has state, behavior, and identity. The structure and behavior of similar objects are defined in their common class. Each object in a diagram indicates some instance of a class. An object that is not named is referred to as a class instance.

The object icon is similar to a class icon except that the name is underlined:

An object's concurrency is defined by the concurrency of its class.

**MESSAGE:**

A message is the communication carried between two objects that trigger an event. A message carries information from the source focus of control to the destination focus of control. The synchronization of a message can be modified through the message specification. Synchronization means a message where the sending object pauses to wait for results.

**LINK:**

A link should exist between two objects, including class utilities, only if there is a relationship between their corresponding classes. The existence of a relationship between two classes symbolizes a path of communication between instances of the classes: one object may send messages to another. The link is depicted as a straight line between objects or objects and class instances in a collaboration diagram. If an object links to itself, use the loop version of the icon.

**6.1.4. CLASS DIAGRAM:**

Identification of analysis classes:

A class is a set of objects that share a common structure and common behavior (the same attributes, operations, relationships and semantics). A class is an abstraction of real-world items.

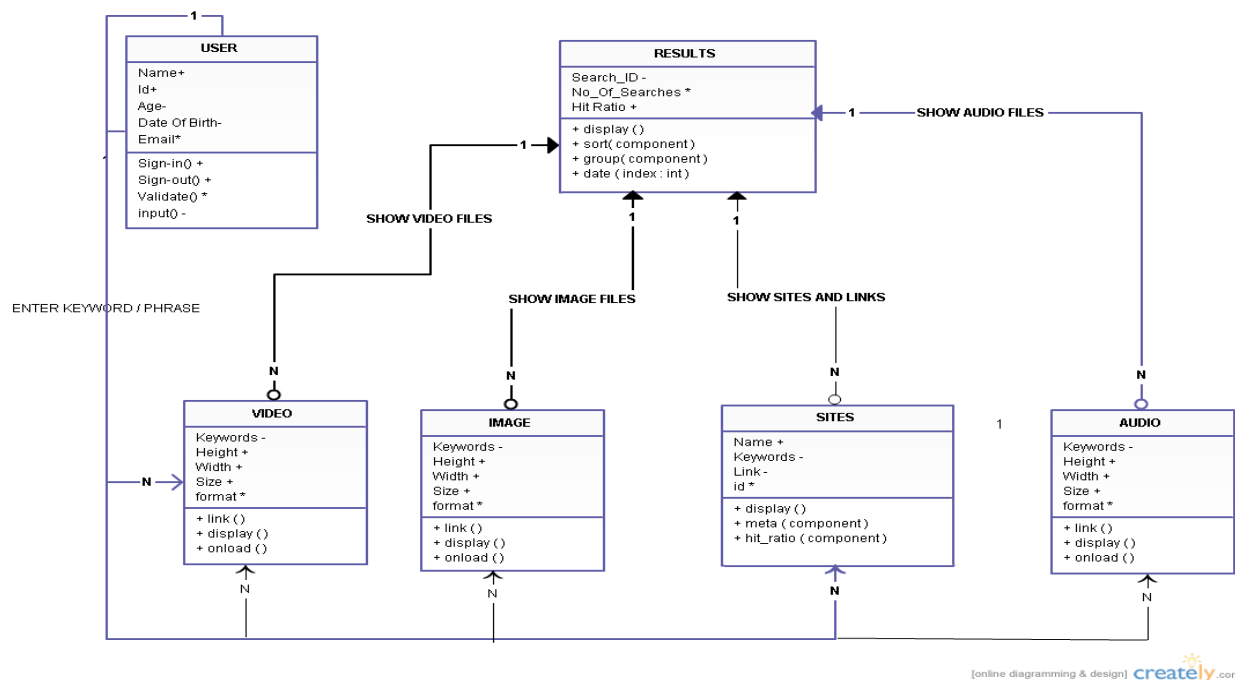
**CLASS DIAGRAM FOR SEARCH ENGINE**

Fig :6.1.4.Class diagram



### 1. TOP-DOWN:

Look for noun phrases composed of various adjectives in a class name. Avoid excessive refinement. Specialize only when the sub classes have significant behavior.

### 2. BOTTOM-UP:

Look for classes with similar attributes or methods. Group them by moving the common attributes and methods to an abstract class. You may have to alter the definitions a bit.

### 3. REUSABILITY:

Move the attributes and methods as high as possible in the hierarchy.

### 4. MULTIPLE INHERITANCES:

Avoid excessive use of multiple inheritances. One way of getting benefits of multiple inheritances is to inherit from the most appropriate class and add an object of another class as an attribute.

## 6.1.5. AGGREGATION OR A-PART-OF RELATIONSHIP:

It represents the situation where a class consists of several component classes. A class that is composed of other classes doesn't behave like its parts. It behaves very differently. The major properties of this relationship are transitivity and anti symmetry.

The **questions** whose answers will determine the distinction between the part and whole relationships are:

- Does the part class belong to the problem domain?
- Is the part class within the system's responsibilities?
- Does the part class capture more than a single value?( If not then simply include it as an attribute of the whole class)
- Does it provide a useful abstraction in dealing with the problem domain?

There are three types of aggregation relationships. They are:

**ASSEMBLY:**

It is constructed from its parts and an assembly-partsituation physically exists.

**CONTAINER:**

A physical whole encompasses but is not constructed from physical parts.

**COLLECTION MEMBER:**

A conceptual whole encompasses parts that may be physical or conceptual. The container and collection are represented by hollow diamonds but composition is represented by solid diamond.

**6.2. DATA FLOW DIAGRAMS:**

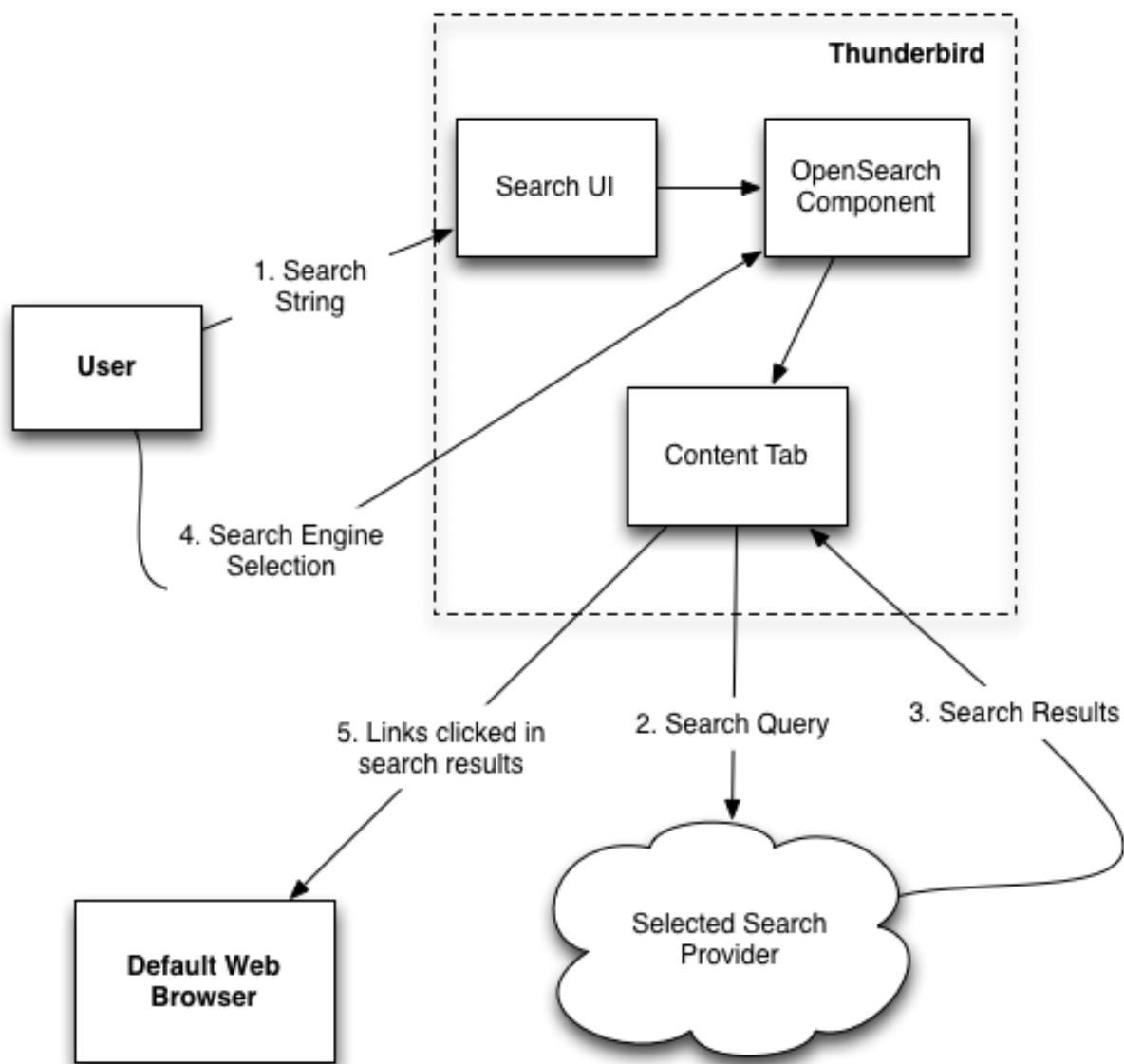
A data flow diagram is graphical tool used to Descriptionribe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be Descriptionribed logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full Descriptionription of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Game and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a Descriptionriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is

necessary and an adequate amount of detail is Descriptionribed for analyst to understand the process.

Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a “bubble Chart” has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.



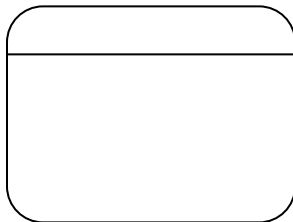
Fig;6.2.Data flow diagram.

### **6.2.1. DFD SYMBOLS:**

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data

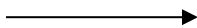
5.



Process that transforms data flow.



Source or Destination of data



Data flow



Data Store

### **6.2.2. CONSTRUCTING A DFD:**

Several rules of thumb are used in drawing DFD'S:

Process should be named and numbered for an easy reference. Each name should be representative of the process. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal. When a process is exploded into lower level details, they are numbered. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each word capitalized. A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow

in and out. Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

### **6.2.3. SILENT FEATURE OF DFD'S :**

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

### **DATA FLOW:**

- 1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a datatore to show a read before an update.
- 2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
- 3) A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
- 4) A Data flow to a data store means update (delete or change).

Table:6.2.3 Dataflow

<b>DFD</b>
<b>Data Base Tables</b>
<b>Location Table</b>
<b>User Registration Table</b>

### **6.3. ENTITY-RELATIONSHIP MODEL:**

A relationship is how the data is shared between entities. There are three types of relationships between entities:

#### **1.ONE-TO-ONE**

One instance of an entity (A) is associated with one other instance of another entity (B). For example, in a database of employees, each employee name (A) is associated with only one social security number (B).

#### **2. ONE-TO-MANY**

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity A. For example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity A.

#### **3. MANY-TO-MANY**

One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A. For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.

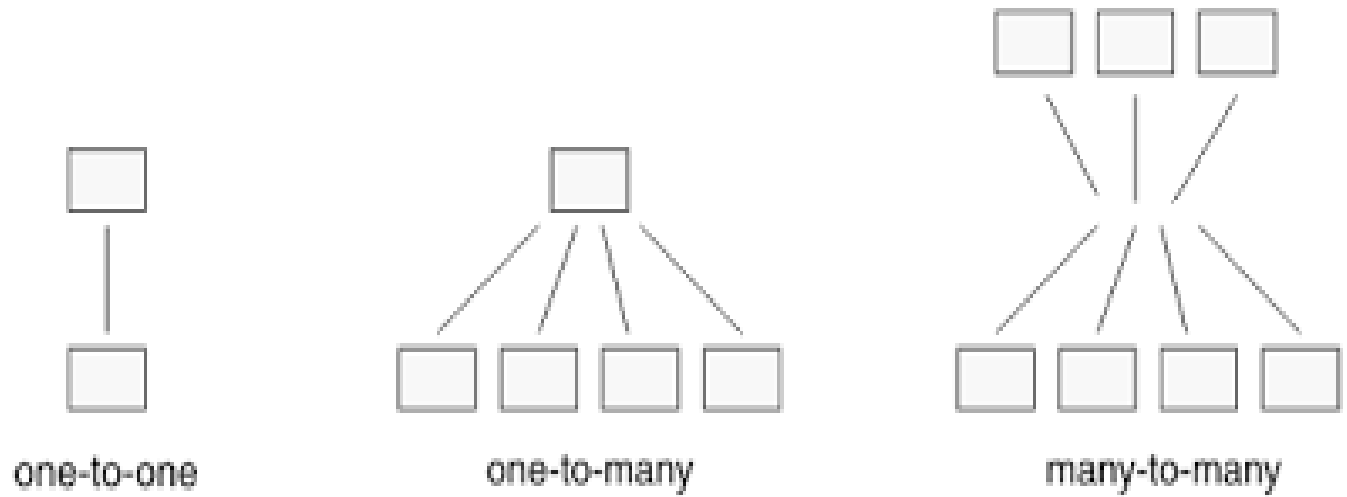
**6.3.1.ER-DIAGRAM:**

Fig:6.3.1( ER- Diagram)

## **7.TESTING**

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

### **7.1. TESTING OBJECTIVES:**

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

### **7.2.LEVELS OF TESTING:**

In order to uncover present in different phases we have the concept of levels of testing.



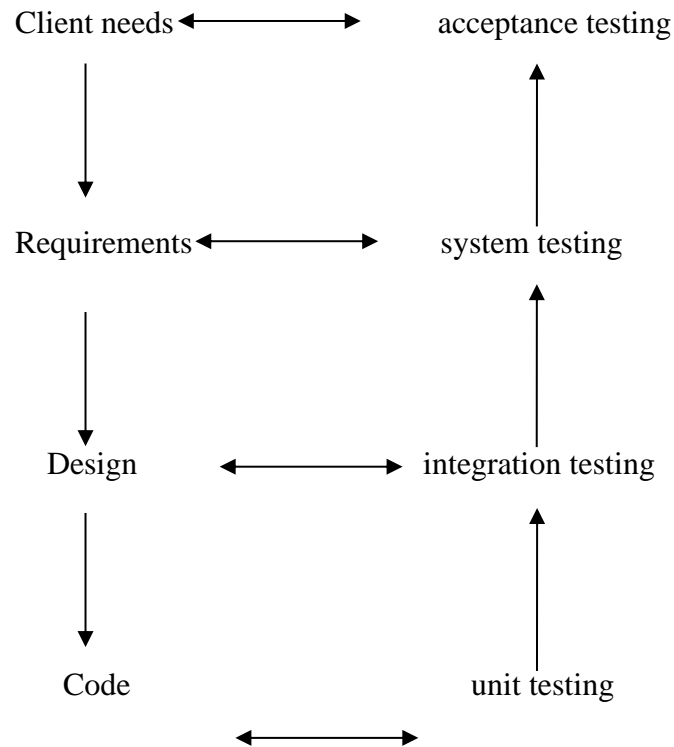
**THE BASIC LEVELS OF TESTING:**

Fig:7.2( Levels of Testing)

**7.3. CODE TESTING:**

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

**7.4. SPECIFICATION TESTING:**

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

## **7.5. UNIT TESTING:**

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

### **7.5.1. BLACK BOX TESTING**

#### **WHAT IS BLACK BOX TESTING?**

Black box testing is a software testing techniques in which **functionality of the software under test (SUT) is tested without looking at the internal code structure**, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

**In Black Box Testing we just focus on inputs and output of the software system** without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

### BLACK BOX TESTING - STEPS

Here are the generic steps followed to carry out any type of Black Box Testing.

- Initially requirements and specifications of the system are examined.
- Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
- Tester determines expected outputs for all those inputs.
- Software tester constructs test cases with the selected inputs.
- The test cases are executed.
- Software tester compares the actual outputs with the expected outputs.
- Defects if any are fixed and re-tested.

### TYPES OF BLACK BOX TESTING

There are many types of Black Box Testing but following are the prominent ones -

- **Functional testing** – This black box testing type is related to functional requirements of a system; it is done by software testers.
- **Non-functional testing** – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements such as performance, scalability, usability.

- **Regression testing** – Regression testing is done after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

### **7.5.2. WHITE BOX TESTING**

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability. White box testing is also known as **clear, open, structural, and glass box testing**.

It is one of two parts of the "**box testing**" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

### **WHAT DO YOU VERIFY IN WHITE BOX TESTING?**

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

## HOW DO YOU PERFORM WHITE BOX TESTING?

To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

### STEP 1) UNDERSTAND THE SOURCE CODE

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

### Step 2) CREATE TEST CASES AND EXECUTE

The second basic step to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of processes in the application. This method requires that the tester must have intimate knowledge of the code and is often done

by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

## **7.6. SYSTEM TESTING:**

Once the individual module testing is completed, modules are assembled and integrated to perform as a system. The top down testing, which began from upper level to lower level module, was carried out to check whether the entire system is performing satisfactorily.

There are three main kinds of System testing:

Alpha Testing

Beta Testing

Acceptance Testing

### **7.6.1. ALPHA TESTING:**

This refers to the system testing that is carried out by the test team with the Organization.

### **7.6.2. BETA TESTING:**

This refers to the system testing that is performed by a selected group of friendly customers

### **7.6.3. ACCEPTANCE TESTING:**

This refers to the system testing that is performed by the customer to determine whether or not to accept the delivery of the system.

<u>Test Condition ID</u>	<u>Descriptionription</u> <u>nof coverage</u>	<u>Expected results</u>	<u>Covered by script</u>
1.	Verification of a particular record	If a particular record already exists it displays a message	This type of test in {verify} procedure in every Jsp file where a record is inserted via an interface
2.	Updating of a particular record	All the details should not be updated.	This type of test is covered in all the Asp files where updations are made.
3.	Validity of login	Only the authorized persons must access system.	This is covered in the login procedure for the validity of a user

Table:7.6.3(Aacceptance Testing)

#### **7.6.4. INTEGRATION TESTING:**

Data can be lost across an interface, one module can have an adverse effort on the other sub functions, when combined, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. The testing was done with sample data. The developed system has run successfully for this sample data. The need for integrated test is to find the overall system performance.

#### **7.7. OUTPUT TESTING:**

After performance of the validation testing, the next step is output testing. The output displayed or generated by the system under consideration is tested by asking the user about the format required by system. The output format on the screen is found to be correct as format was

designed in the system phase according to the user needs. Hence the output testing does not result in any correction in the system.

### **7.8. TEST PLAN:**

The test-plan is basically a list of test-cases that need to be run on the system. Some of the test-cases can be run independently for some components (report generation from the database, for example, can be tested independently) and some of the test-cases require the whole system to be ready for their execution. It is better to test each component as and when it is ready before integrating the components. It is important to note that the test-cases cover all the aspects of the system (ie, all the requirements stated in the RS document).

Figure.7.8: Test plan

<b><u>Sno</u></b>	<b><u>Test case Title</u></b>	<b><u>Descriptionription</u></b>	<b><u>Expected Outcome</u></b>	<b><u>The requirement in RS that is being tested</u></b>	<b><u>Result</u></b>
1	Successful User Verification	The login to the system should be tried with the login assigned by the admin and the correct password	Login should be successful and the user should enter in to the system	RS1	Passed
2	Unsuccessful User Verification due to wrong password	Login to the system with a wrong password	Login should fail with an error 'Invalid Password'	RS1	Passed
3	Unsuccessful User Verification due to invalid login id	Login to the system with a invalid login id	Login should fail with an error 'Invalid user id'	RS1	Passed

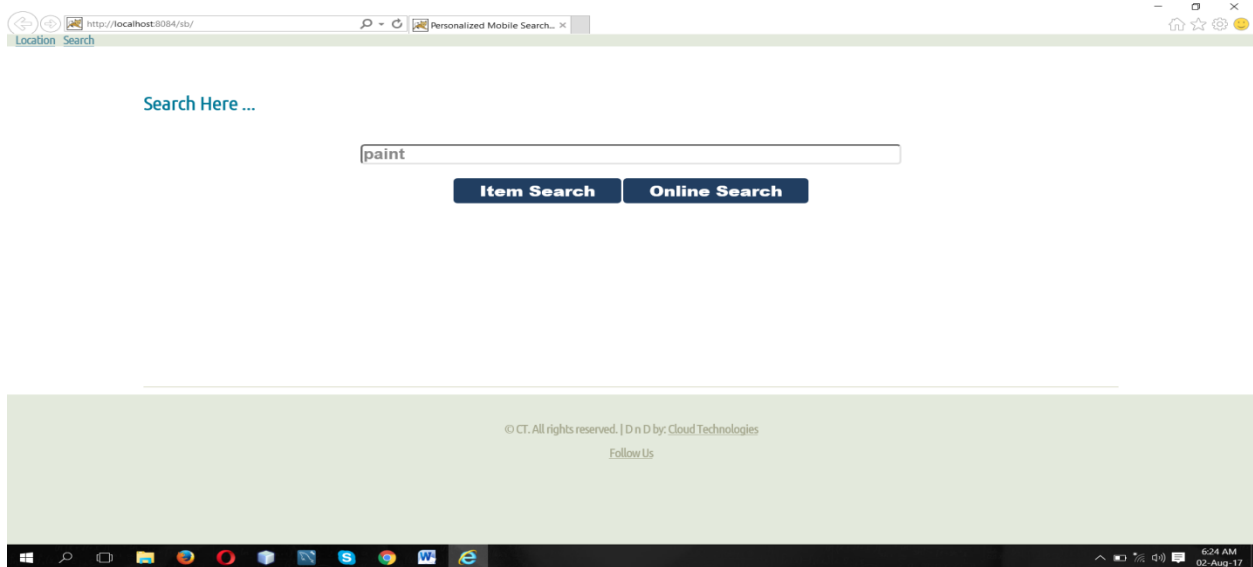
Fig:7.8( Test plan).



## 8.SCREENSHOTS

### 8.1. USER SEARCH:

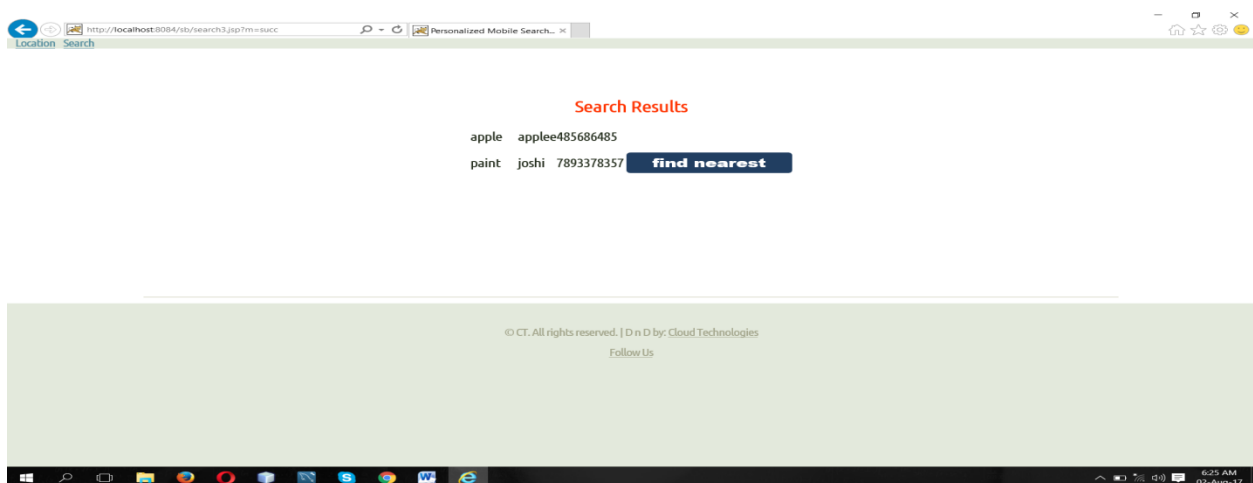
**Description:** Here the user can search the information needed for the user



**Fig:8.1**(User search)

### 8.1.1.USER SEARCH RESULTS:

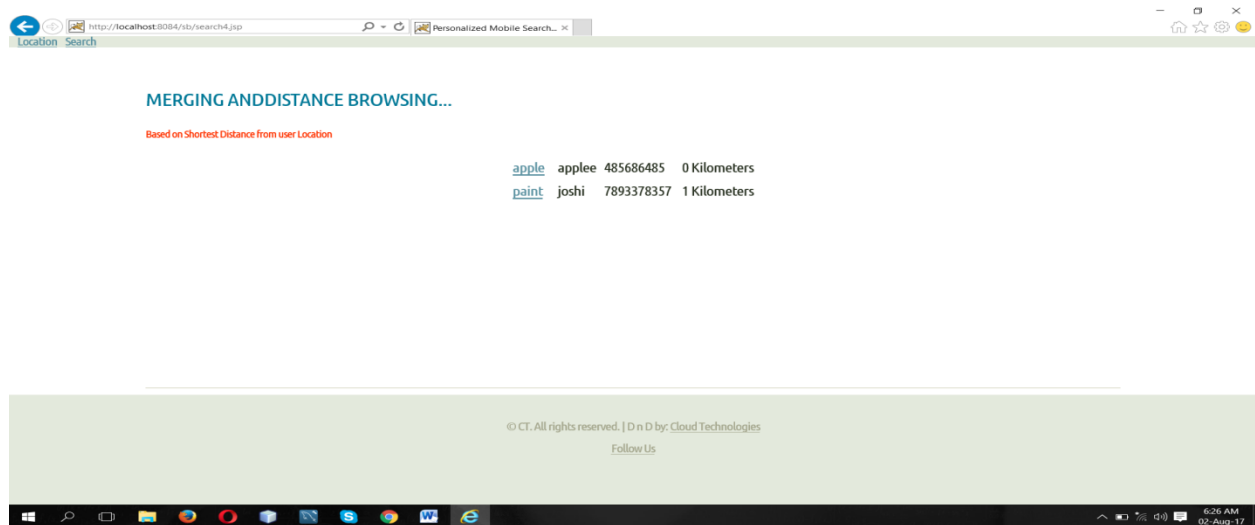
**Description:** The user searched data will be displayed here and the nearest place will be displayed



**Fig :8.1.1**(User search result)

### 8.1.2.SEARCH THE NEAREST CLIENT LOCATION:

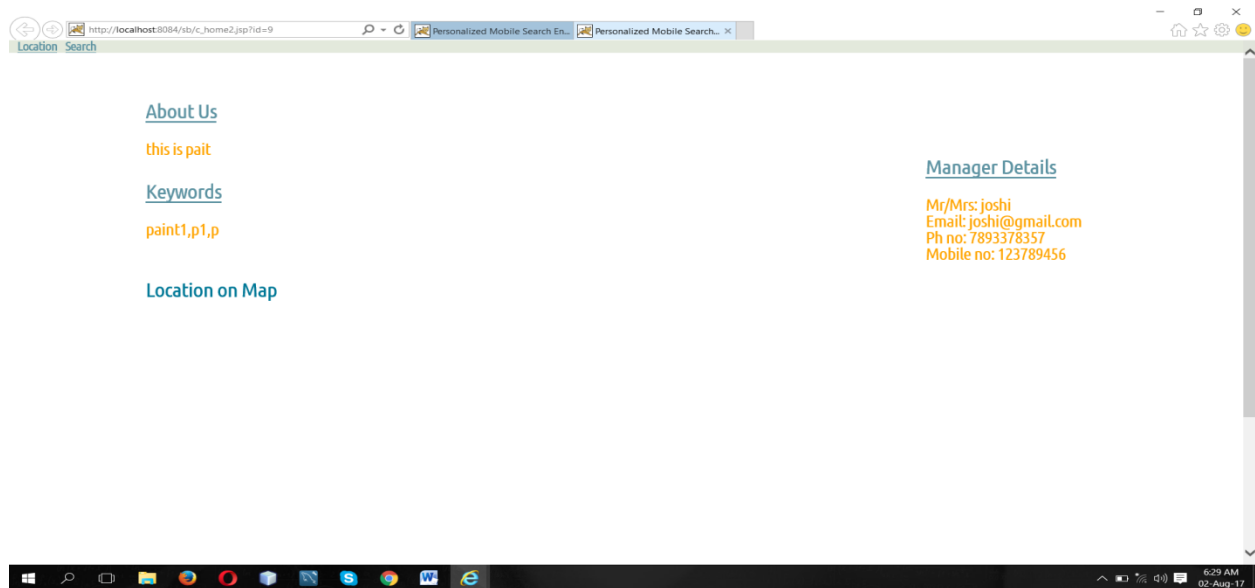
**Description:** The nearest distance to the user's location is calculated.



**Fig: 8.1.2(Search the nearest client location)**

### 8.1.3.CLIENT INFORMATION TO USER:

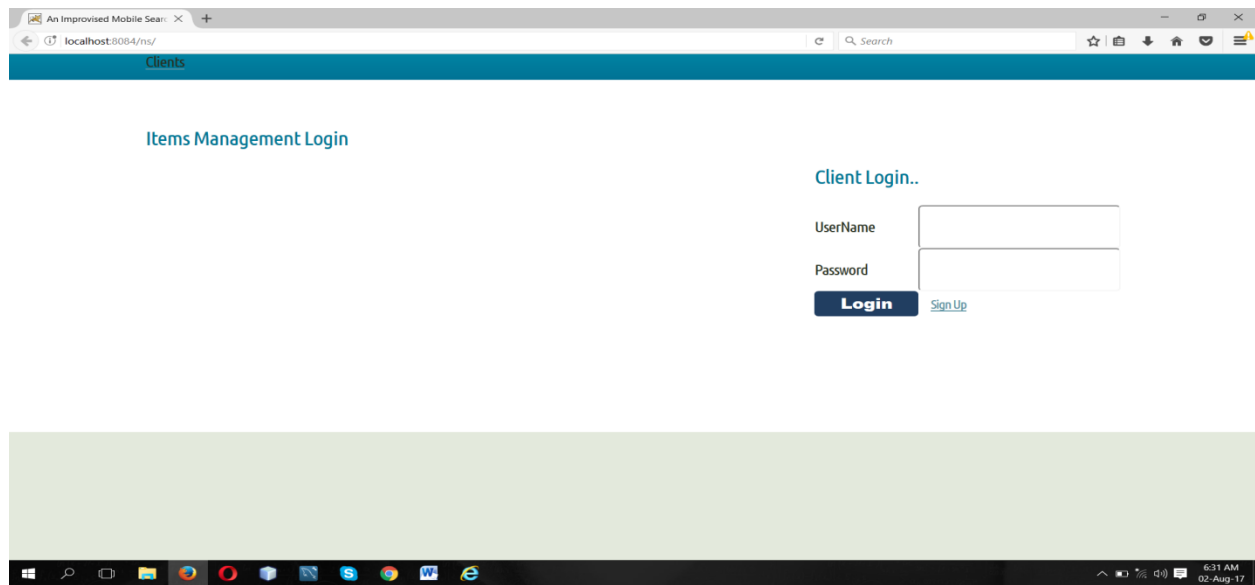
**Description:** Client information to the user.



**Fig :8.1.3(Client information to user)**

## 8.2. CLIENT LOGIN:

**Description:** The client can login through this.

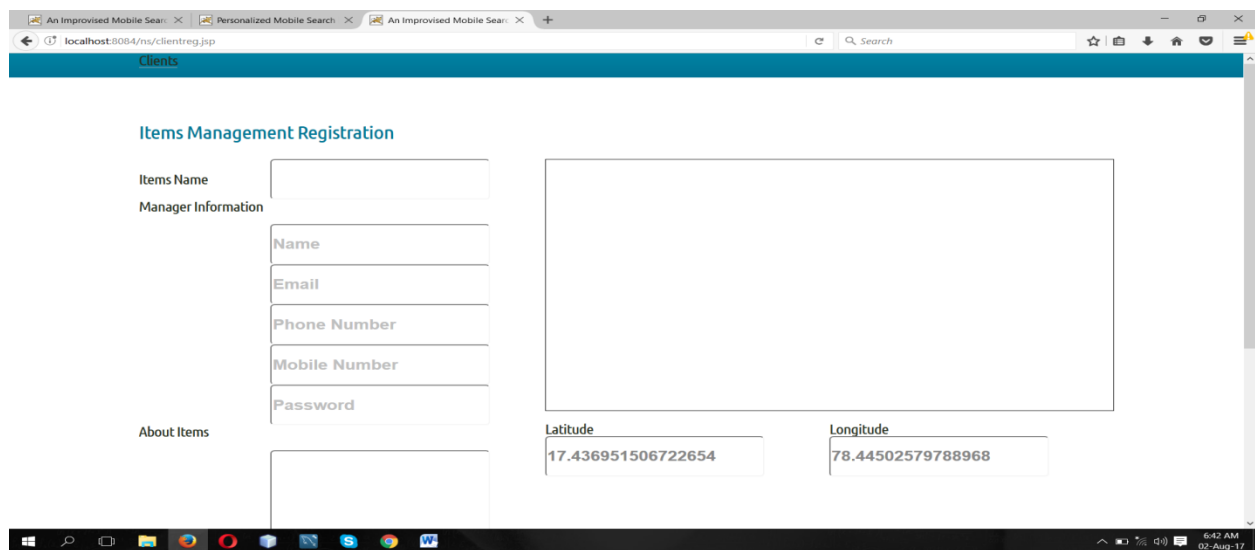


The screenshot shows a web browser window with the address bar displaying 'localhost:8084/ms/'. The page has a blue header with the text 'Clients'. Below the header, there is a section titled 'Items Management Login'. To the right of this section, there is a 'Client Login..' form. The form contains two input fields: 'UserName' and 'Password'. Below these fields are two buttons: 'Login' (in a dark blue box) and 'Sign Up' (in a light blue box). The browser's taskbar at the bottom shows various application icons and the system clock indicating 6:31 AM on 02-Aug-17.

**Fig:8.2(Client login)**

### 8.2.1.REGISTRATION OF CLIENT:

**Description:** Client registration with the items and location.

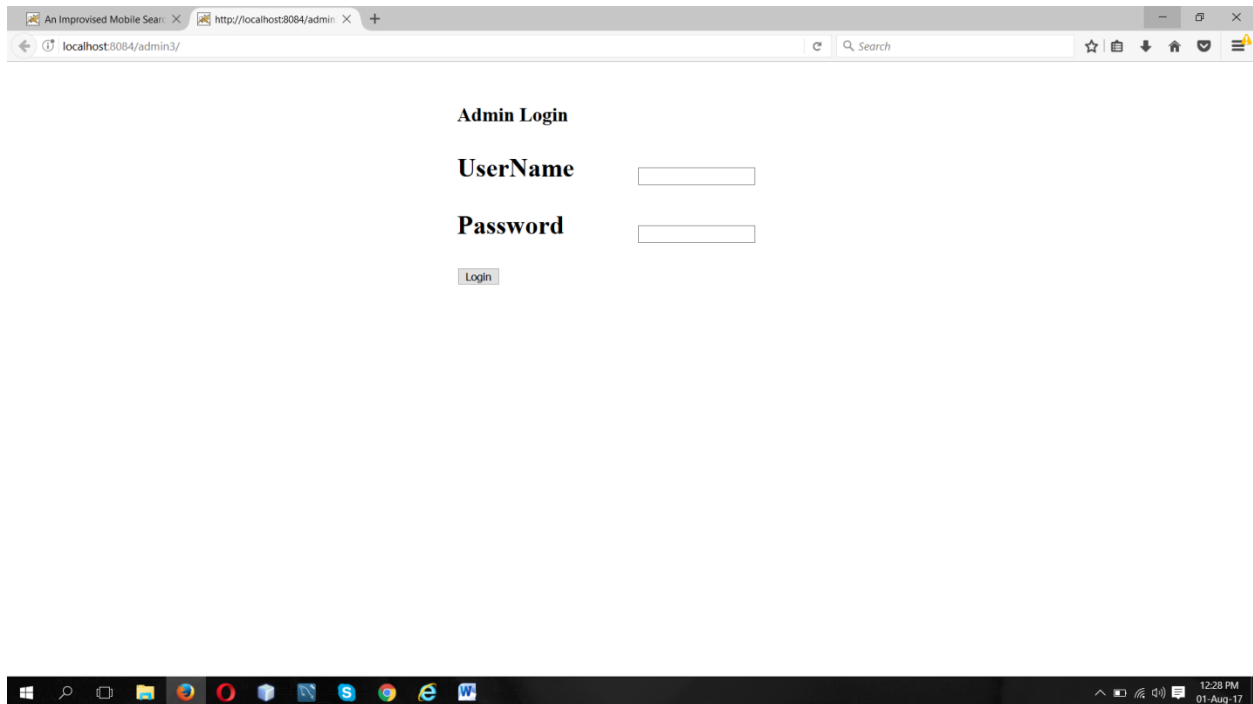


The screenshot shows a web browser window with the address bar displaying 'localhost:8084/ms/clientreg.jsp'. The page has a blue header with the text 'Clients'. Below the header, there is a section titled 'Items Management Registration'. The registration form is divided into several sections: 'Items Name' with a single input field; 'Manager Information' with a large rectangular input area; 'About Items' with a single input field; and a section for location coordinates with two input fields labeled 'Latitude' and 'Longitude'. The 'Latitude' field contains the value '17.436951506722654' and the 'Longitude' field contains the value '78.44502579788968'. The browser's taskbar at the bottom shows various application icons and the system clock indicating 6:42 AM on 02-Aug-17.

**Fig:8.2.1(Registration of client)**

### 8.3. ADMIN LOGIN PAGE:

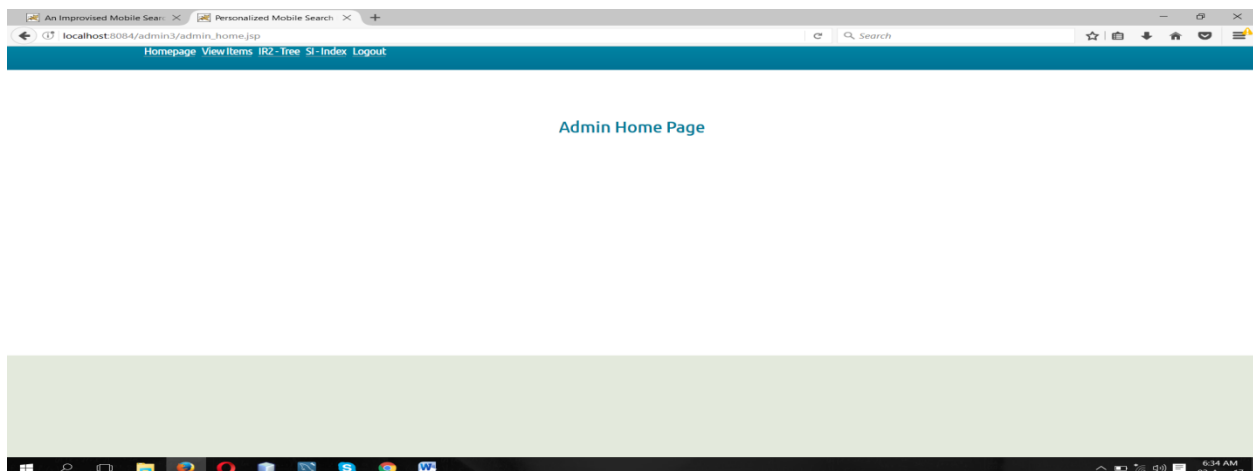
**Description:** Admin should enter the username and password for the clients information.



**Fig:8.3(Admin login page)**

#### 8.3.1.ADMIN HOME PAGE:

**Description:** After logged in with the admin the admin home page will be displayed.



**Fig:8.3.1(Admin home page)**

### 8.3.2.ITEMS CONNECTION WITH THE CLIENT:

**Description:** Displaying the items present in the admin page.

Hotel	Manager Name	Email	City
<a href="#">Surya Hotel</a>	ali	ali@gmail.com	Hyderabad
<a href="#">apple</a>	applee	apple@gmail.com	Hyderabad
<a href="#">paint</a>	joshi	joshi@gmail.com	Hyderabad
<a href="#">egg curry</a>	sai	kondasai@gmail.com	Hyderabad
<a href="#">Malar hotel</a>	malar	malar@gmail.com	Hyderabad
<a href="#">shawarma</a>	sai	sai@gmail.com	Hyderabad
<a href="#">Hyderabad Dhaba</a>	Sajid	sajid1@in.om	Hyderabad
<a href="#">SJ Rest</a>	sajid	sajid24x7@gmail.com	Hyderabad
<a href="#">CT Solutions</a>	Sajid	sajid2@gmail.com	Hyderabad
<a href="#">Blue Fox</a>	SAJID	sajid@in.om	Hyderabad
<a href="#">Ruchi Resturent</a>	siva	siva@gmail.com	Hyderabad
<a href="#">zomo</a>	sumo	sumo@gmail.com	Hyderabad
<a href="#">Kruthinga Resturent</a>	Swamy	swamy@gmail.com	Hyderabad

**Fig:8.3.2(Items connection with the client)**

### 8.3.3PROCESSING OF THE ITEMS WITH IR2-TREE :

**Description:**The items with the details should be processed with i2-tree.

IR2-Tree

They nicely integrate two well-known concepts: R-tree, a popular spatialindex, and signature file, an effective method for keyword-based document retrieval. By doing so they develop a structure called the IR 2 -tree

S.no Unique Keywords

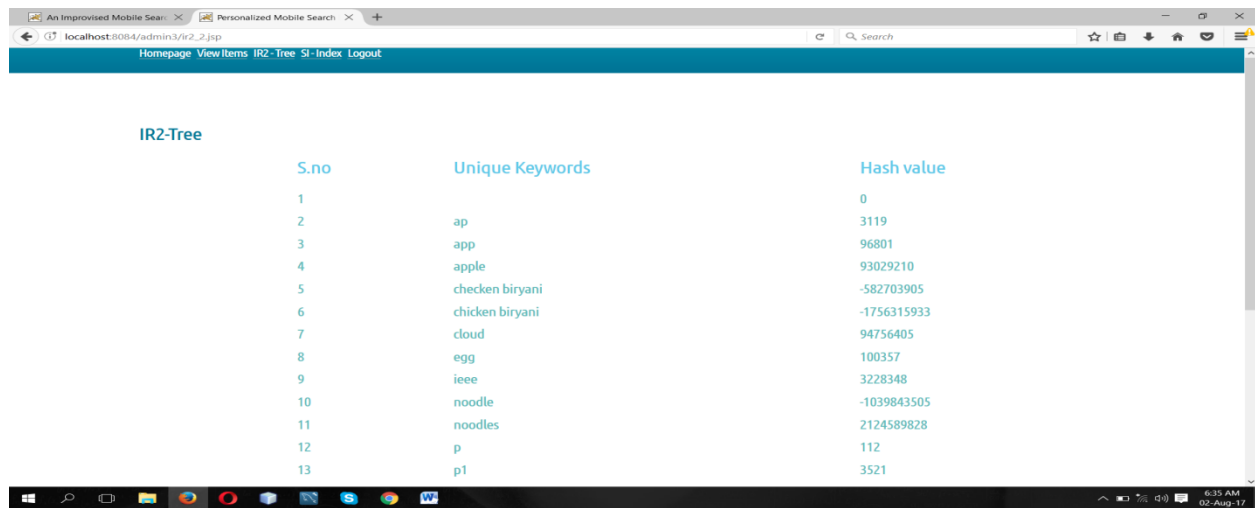
- ap
- app
- apple
- chicken biryani
- cloud
- egg
- leee
- noodle
- noodles
- p
- p1
- p2
- paint1

Process

**Fig:8.3.3( Processing of the items with ir2-trees)**

### 8.3.4 IR2-TREE WITH KEYWORDS:

**Description:** the item details with the unique key words and hash values.



S.no	Unique Keywords	Hash value
1		0
2	ap	3119
3	app	96801
4	apple	93029210
5	checken biryani	-582703905
6	chicken biryani	-1756315933
7	cloud	94756405
8	egg	100357
9	ieee	3228348
10	noodle	-1039843505
11	noodles	2124589828
12	p	112
13	p1	3521

**Fig:8.3.4(Ir2-tree with keywords)**

## 9.CONCLUSION

### 9.1.CONCLUSION

To adapt to the user mobility, we incorporated the user's Google Map locations in the personalization process. We observed that Google Map locations help to improve retrieval effectiveness, especially for location queries. We also proposed two privacy parameters, minimum Distance and exp Ratio, to address privacy issues in PMSE by allowing users to control the amount of personal information exposed to the PMSE server. The privacy parameters facilitate smooth control of privacy exposure while maintaining good ranking quality. For future work, we will investigate methods to exploit regular travel patterns and query patterns from the Google Map and click through data to further enhance the personalization effectiveness of PSME.

### 9.2.FUTURE SCOPE

Smaller search engines have also materialized over the past few weeks, each offering to improve the user experience. [Grokker](#) offers an interface that groups search results graphically, improving the way search results are segmented and displayed. [Eurekster](#), combines the social networking elements that are used by sites such as [Friendster](#), and provides results that can be filtered based upon what members of your group are searching. While all of these are interesting and provide a glimpse of the future of search, it will not be the small companies that change the way we search. With Google about to get an influx of cash from its upcoming IPO, Yahoo re-vamping Inktomi and Overture, and Microsoft finally jumping into the search arena, it will be these search engine powerhouses that enhance our search experience and take search engine technology to the next level.

## 10.REFERENCES

- [1] K.W. Church, W. Gale, P. Hanks, and D. Hindle, "Using Statistics in Lexical Analysis," Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon, Psychology Press, 1991.
- [2] Q. Gan, J. Attenberg, A. Markowetz, and T. Suel, "Analysis of Geographic Queries in a Search Engine Log," Proc. First Int'l Workshop Location and the Web (LocWeb), 2008.
- [3] Y.-Y. Chen, T. Suel, and A. Markowetz, "Efficient Query Processing in Geographic Web Search Engines," Proc. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR), 2006.
- [4] E. Agichtein, E. Brill, and S. Dumais, "Improving Web Search Ranking by Incorporating User Behavior Information," Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR), 2006.
- [5] T. Joachims, "Optimizing Search Engines Using Clickthrough Data," Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining, 2002.
- [6] K.W.-T. Leung, W. Ng, and D.L. Lee, "Personalized Concept Based Clustering of Search Engine Queries," IEEE Trans. Knowledge and Data Eng., vol. 20, no. 11, pp. 1505-1518, Nov. 2008.
- [7] B. Liu, W.S. Lee, P.S. Yu, and X. Li, "Partially Supervised Classification of Text Documents," Proc. Int'l Conf. Machine Learning (ICML), 2002.
- [8] J. Teevan, M.R. Morris, and S. Bush, "Discovering and Using Groups to Improve Personalized Search," Proc. ACM Int'l Conf. Web Search and Data Mining (WSDM), 2009.
- [9] S. Yokoji, "Kokono Search: A Location Based Search Engine," Proc. Int'l Conf. World Wide Web (WWW), 2001.
- [10] Fang Liu, Clement Yu, and Weiyi Meng "Personalized Web Search for Improving Retrieval Effectiveness" IEEE Trans. Knowledge and Data Eng., Vol. 16, No. 1, January 2004