1. INTRODUCTION

1.1. Location Based Services (LBS)

Location-based services (LBSs) have been popular in recent years. The widely used modern mobile devices, such as smartphones and tablets, provide mobile users with numerous opportunities of communications and high awareness of their surroundings. For example, users can search for the nearest clinics or banks. LBSs serve users based on their submitted LBS queries (e.g., "Show me the clinic information within 1 mile"), which typically include a precise position location. However, the submitted information may be abused by untrusted LBS servers (and other parties that compromise the servers). They may track users or release their personal information to third parties, such as advertisers. Thus, we need to pay significant attention to protecting people's privacy. Numerous studies have been conducted to protect the location privacy of users. These studies have concentrated on dummy based anonymization. Users do not need to submit their precise location, they only submit a generated dummy location to servers. Thus, their true position cannot be obtained by an attacker.

However, existing methods, such as PAD (Privacy Area Dummy), require the server to return the request results of all k-generated dummy locations, thereby resulting in a waste of bandwidth and cost considerable processing time.

We accordingly propose a privacy-protected algorithm based on centroid without the true position for LBSs. In our proposed method, the request to LBS server does not contain the real user location; thus, we cannot judge the direct relation of two locations. So we adopt a centroid-based method, that is, to calculate the centroid of the dummy locations of one user and that of others , then calculate the Euclidean distance between the centroids. The method has several benefits. First, a centroid is equivalent to a cluster of a set of points, therefore, it can well represent the distribution of dummy locations and it can be used instead of the real location of users to calculate Euclidean distance. Second, the centroid requires minimal computation time because it performs only one calculation of Euclidean distance. Third, the location privacy of users can be protected given that the submitted locations do not contain their precise location. Finally, the proposed method requires the server to return only the request the user needs, thereby saving considerable bandwidth. The proposed method can well balance the user experience and protect the privacy of the user's location, in the meantime, it can save bandwidth and improve query success rate. Our system is based on the B/S architecture , so it is easy to deploy on the client side.

1.2. Project Overview

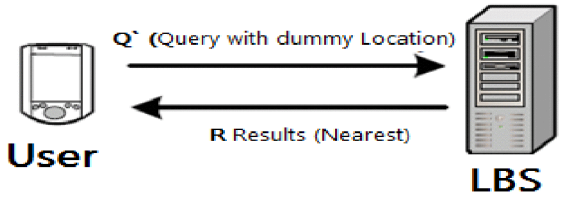
1.2.1. Existing System

- Numerous studies have been conducted to protect the location privacy of users. Most of these approaches achieve k-anonymity using location perturbation and obfuscation.
- But the location anonymizer stores the exact location information of all users, thus compromising the privacy.
- Some studies shows that the system may require a trusted third party, responsible for carrying out simple matching operations correctly that would modify the location of users.

- But it is difficult to find a third party that can be fully trusted with protection of user location details.

1.2.2. Proposed System

- The main idea is to replace a trusted third party, termed query server (QS), between the user and the service provider (LBS).
- A centroid is equivalent to a cluster of a set of points. Hence it can well represent the distribution of dummy locations and it can be used instead of the real location of users.
- Finally, the proposed method requires the server to return only the request the user needs, thereby saving considerable bandwidth.



1.2.3. Scope of Proposed System

The scope of our application Location Privacy Protection in Location Based Services using KNN Queries is to provide an efficient and enhanced software tool for the users which:

- Requires minimal computation time.
- Requires minimal communication cost, saving bandwidth.
- User exact location is secure.

2. SYSTEM REQUIREMENTS

2.1. Software Requirements

- Operating System : Windows family
- Technology : Java (1.7/1.8)
- Web Technologies : Html, Html-5, JavaScript, CSS.
- Web Server : Apache Tomcat 7/8
- Database : My SQL 5.5
- UML : Star UML

2.2. Hardware Requirements

- Processer : intel Core i3 or above.
- Ram : Min 1 GB
- Hard Disk : Min 100 GB

3. TECHNOLOGY

3.1. Javascript

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

- JavaScript is a lightweight, interpreted programming language.
- Designed for creating network-centric applications.
- Complementary to and integrated with Java.
- Complementary to and integrated with HTML.
- Open and cross-platform.

3.1.1. Client-Side JavaScript

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser. It means that a web page need not be a static HTML, but can include programs that interact with the user, control the browser, and dynamically create HTML content. JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

3.1.2 Advantages of JavaScript

- Less server interaction − You can validate user input before sending the page off to the server. This saves server traffic, which means less load on your server.

- Immediate feedback to the visitors − They don't have to wait for a page reload to see if they have forgotten to enter something.

- Increased interactivity − You can create interfaces that react when the user hovers over them with a mouse or activates them via the keyboard.

- Richer interfaces − You can use JavaScript to include such items as drag-and-drop components and sliders to give a Rich Interface to your site visitors.

3.2 HTML

HTML stands for Hypertext Markup Language, and it is the most widely used language to write Web Pages. Hypertext refers to the way in which Web pages (HTML documents) are linked together. Thus, the link available on a webpage is called Hypertext.

As its name suggests, HTML is a Markup Language which means you use HTML to simply "mark-up" a text document with tags that tell a Web browser how to structure it to display. Originally, HTML was developed with the intent of defining the structure of documents like headings, paragraphs, lists, and so forth to facilitate the sharing of scientific information between researchers.

Now, HTML is being widely used to format web pages with the help of different tags available in HTML language.

3.2.1 Advantages:

- HTML is used to build a websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript etc.

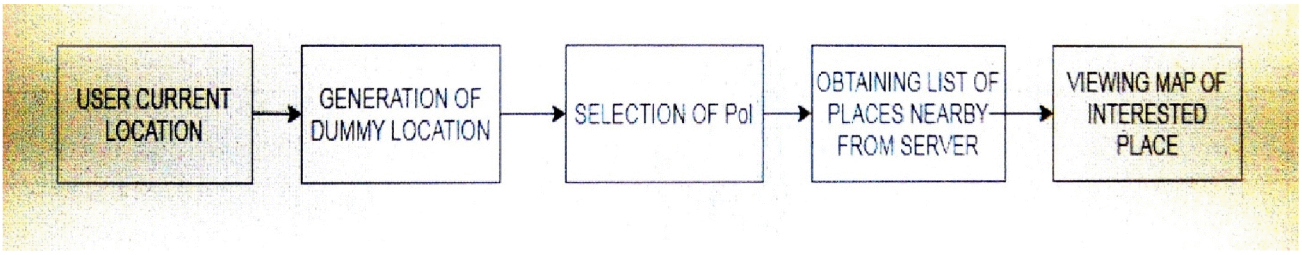
3.2.2 HTML Document Structure

A typical HTML document will have the following structure −

```
<html>
    <head> Document header related tags </head>
     <body>   Document body related tags  </body>
</html>
```

4. SYSTEM DESIGN

4.1. SYSTEM ARCHITECTURE



4.2. INTRODUCTION TO UML

UML stands for Unified Modelling Language. UML is a standardized general purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

- The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.
- The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.
- The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

The Primary goals in the design of the UML are as follows:

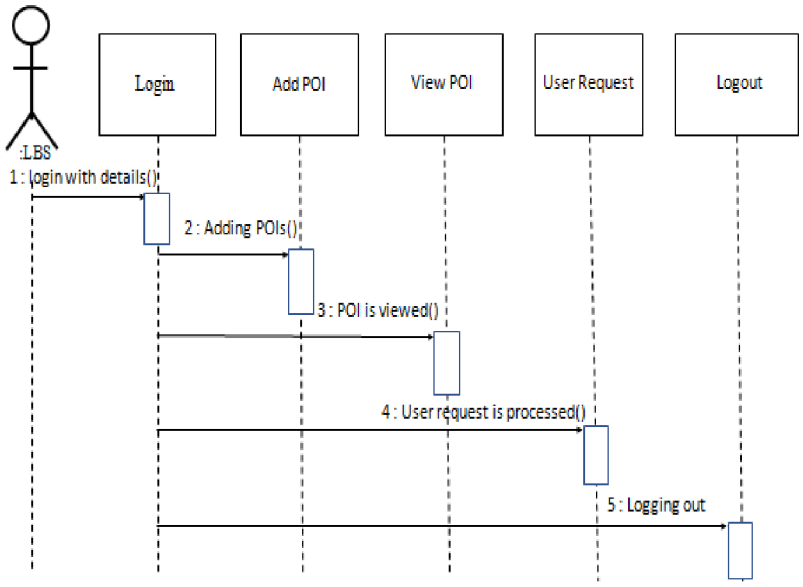
1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.

5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

4.3. UML DIAGRAMS

4.3.1. Sequence Diagram

A sequence diagram in UML is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



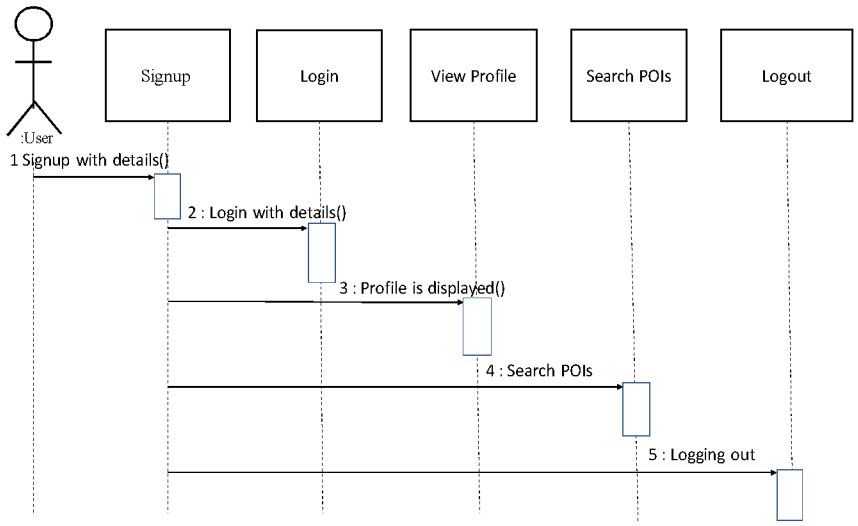


Fig 4.3.1: Sequence Diagrams

4.3.2 Activity Diagram

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

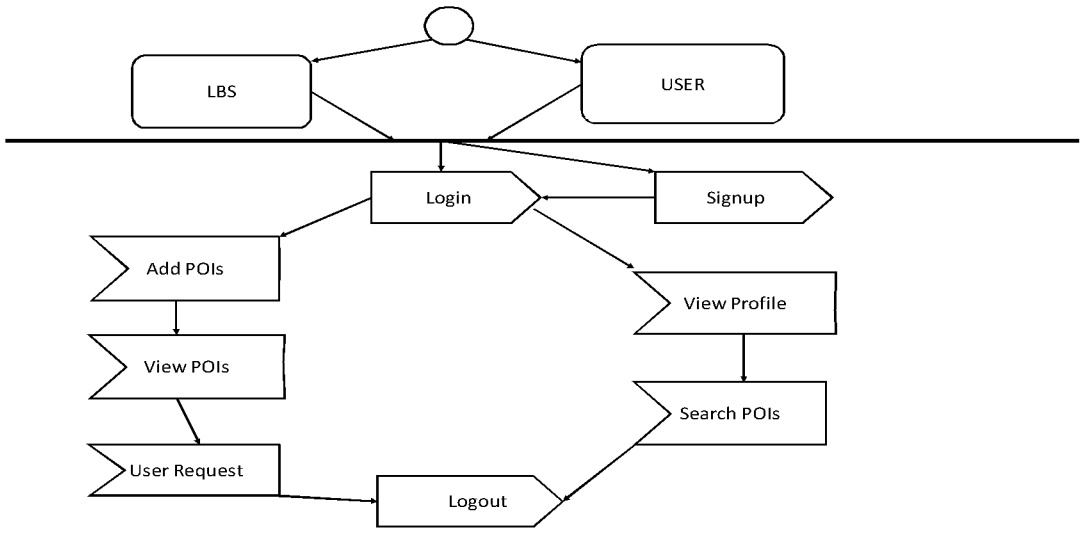


Fig 4.3.2: Activity Diagram

5. IMPLEMENTATION

5.1. MODULES

5.1.1. User Side

User is an LBS end user. Users can register and login into our application. Users share location their dummy locations instead of their exact location by using k-nearest neighbors algorithm. Users can specify the distance they want to travel to reach the location of PoI. Then a request with user's dummy location, PoI and distance range is generated and sent to the LBS server.

5.1.2. Server Side

LBS is a Location Based Server. This module maintains the spatial database and adds the PoI data.A spatial database is a database that is designed to store and get information that speaks to objects characterized in a geometric space such as maps etc. Here, the server accepts the users' requests and sends the locations of the nearest POIs users requested for.

5.2. Dataset

The dataset used for our application is a sample dataset of Zomato organization. The dataset consists of place ID, place name, place type (restaurant etc.), latitude, longitude, city. The input parameters used to retrieve data are the user generated dummy location, predicate for PoI and the distance range between dummy location and PoI location. The outputs are the names of the PoIs within the specified range and their location on the Google map.

pid	pname	type_	lat	lon	city
18339874	Farzi Cafe	restaurant	D12.9721612	D77.5960137	Bangalore
18340881	Barbeque Nation	restaurant	D25.11338	D55.215341	Dubai
18343124	Swaaddesh	restaurant	D25.61552768	D85.1154613	Patna
18343731	Mumbai Local	restaurant	D22.5336623	D88.3662166	Kolkata
18344478	Aravali Owls	restaurant	D0	D0	Faridabad
18345740	Aravalli Owls	restaurant	D0	D0	Faridabad
18346996	3Cherryz Sky Lounge & Cafe	restaurant	D25.31717627	D82.99116335	Varanasi
18349251	Prelibato	restaurant	D0	D0	Faridabad
18350020	Kargo	restaurant	D18.53656175	D73.89648478	Pune
18352452	Lucky Cat Coffee & Kitchen	restaurant	D-6.218932479	D106.8317481	Jakarta
18353030	Oh My!	restaurant	D0	D0	Faridabad
18353121	Flechazo	restaurant	D12.97537691	D77.696664	Bangalore
18354483	Farzi Cafe	restaurant	D18.5436256	D73.9051007	Pune
18355112	Burger Point	restaurant	D28.3949736	D77.3220773	Faridabad
18356019	The Hub	restaurant	D28.3765371	D77.3315619	Faridabad
18356469	Prost Brew Pub	cafe	D17.43147653	D78.40035025	Hyderabad

5.3. Code Snippets

5.3.1. Home Page

```
<%
   String message=request.getParameter("m");
   if(message!=null && message.equalsIgnoreCase("fail")){
   out.println("<script type=text/javascript>alert('Sorry, Login Fail')</script>");
}
 %>
<!DOCTYPE html>
<html lang="en">
<head>
<title>LBSNs</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<script type="application/x-javascript"> addEventListener("load", function() { setTimeout(hideURLbar, 0);
}, false); function hideURLbar(){ window.scrollTo(0,1); } </script>
<link href="css/bootstrap.css" type="text/css" rel="stylesheet" media="all">
<link href="css/style.css" type="text/css" rel="stylesheet" media="all">
<link href="css/font-awesome.css" rel="stylesheet"> <!-- font-awesome icons -->
<link rel="stylesheet" href="css/flexslider.css" type="text/css" media="screen" property="" />
</head>
<body>
        <div class="header">
        <h1><a href="#">Application  for Location Privacy Protection in LBS using KNN Queries
</a></h1>
        </div>
        <div class="header-nav">
```

```
<div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
<ul class="nav navbar-nav cl-effect-16">
<li><a href="index.jsp" class="active">Home</a>
    </li>
    <li><a href="lbs.jsp">LBS</a>
    </li>
    <li><a href="user.jsp">User</a>
    </li>
<li><a href="adduser.jsp">User Signup</a>
    </li></ul>
<div class="clearfix"> </div>
</div>
<div class="banner-text agileinfo">
<div class="container">
<div class="agile_banner_info">
<div class="agile_banner_info1">
<div id="typed-strings" class="agileits_w3layouts_strings">
                <p>>> Application  for Location  </p>
                <p><i> Privacy Protection in LBS </i></p>
                <p><i> using KNN Queries </p>
</div>
<span id="typed" style="white-space:pre;"></span>
</div> </div>
</div>
```

5.3.2. User Current Location

```
<%
```

```
String title="Your Current Location";
%>
<%@ include file="uheader.jsp"%>
<%
int ccc=(Integer)session.getAttribute("ccc");
if(ccc>1){
response.sendRedirect("user.jsp?id=ccc");
}else{
        session.setAttribute("ccc",2);
        }
%>
<!DOCTYPE html>
<html>
  <head>
  <script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBN2bc7BmDlgN437Jn4tPOQAzCOs9k7
nZo&callback=initMap"
  async defer></script>
<script>
        if (navigator.geolocation)
        {               navigator.geolocation.getCurrentPosition(showCurrentLocation);
        }
        else
        {               alert("Geolocation API not supported.");
        }
```

```
        function showCurrentLocation(position)
        {
            var latitude = position.coords.latitude;
            var longitude = position.coords.longitude;
            var coords = new google.maps.LatLng(latitude, longitude);
        var mapOptions = {
            zoom: 15,
            center: coords,
            mapTypeControl: true,
            mapTypeId: google.maps.MapTypeId.ROADMAP
        };
        map = new google.maps.Map(
        document.getElementById("mapPlaceholder"), mapOptions
        );
        var marker = new google.maps.Marker({
        position: coords,
        map: map,
        title: "Current location!"
        });
        }
    </script>  </head>
<style>
#mapPlaceholder {
    height: 400px;
    width: 900px;
```

```
    </style>
    <body>
    <div><h2>Current Location</h2>
    <div id="mapPlaceholder"></div>
     </div></body>
<p id="demo"><h3><font size="" color="#99cc00">Click the button to get your coordinates:</p></font>
<div class="form_settings">
<input type="button" class="submit" onclick="getLocation()" value="Set Location">
</div>
<script>
var x = document.getElementById("demo");
function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showPosition);
    } else {
        x.innerHTML = "Geolocation is not supported by this browser.";
    }
}
function showPosition(position) {
                document.getElementById('latitude').value=position.coords.latitude ;
                document.getElementById('longitude').value=position.coords.longitude ;
}
</script>
<form method="post" action="upoi2.jsp">
```

Latitude: <input type="text" id="latitude" name="lat" required/>

Longitude : <input type="text" id="longitude" name="lon"/>

<input type="submit" class="submit" value="Centroid System"/>

</form></html>

<%@ include file="footer.jsp"%>

5.3.3. Getting dummy location

```
<%
String title="Distribution of dummy location..";
%>
<%@ include file="uheader.jsp"%>
<style type="text/css">
        #map {
    width: 500px;
    height: 200px;
    -webkit-border-radius: 10px;
    -moz-border-radius: 10px;
    border-radius: 10px;
    z-index: 1;
}
</style>
<%
session.setAttribute("ulat",request.getParameter("lat"));
session.setAttribute("ulon",request.getParameter("lon"));
```

```
%>
<center>
<script
src="http://maps.googleapis.com/maps/api/js">
</script>
<script>
var map;
function initialize()
{
var mapOpt = {
center:new
google.maps.LatLng(document.getElementById("lat").value,document.getElementById("lon").value
),
  zoom:16,
  mapTypeId:google.maps.MapTypeId.ROADMAP
  };
map=new google.maps.Map(document.getElementById("googleMap"),mapOpt);
}
google.maps.event.addDomListener(window, 'load',initialize);
</script></head>
<body>
<script type="text/javascript">
function getLocationDetails()
{
        alert(map.getBounds());
        document.getElementById("loc").value=map.getBounds();
```

```
}
</script><table border="2">
<tr><td>
<div id="googleMap"   border="2" style="width:400px;height:300px;"></div>
</td></tr>
</table></body>
</html><table><tr><td>
<button onclick="getLocationDetails();">Get Dummy Locations</button>
<form method="post" action="upoi3.jsp">
        <input type="hidden" name="lat" id="lat" value="<%=request.getParameter("lat")%>" >
        <input type="hidden" name="lon"id="lon" value="<%=request.getParameter("lon")%>">
        <input type="text"  name="loc" class="form-control" id="loc" required size="90" >
<br><br></table>
<input type="submit" value="Next" class="submit">
</form>
<%@ include file="footer.jsp"%>
```

5.3.4. User point of interest selection

```
<%
String title="Distribution of dummy location..";
%>
<%@ include file="uheader.jsp"%>
<%@ page  import="java.sql.*" import="com.mysql.*"  %>
<%
```

```
String loc=request.getParameter("loc");

String lat=request.getParameter("lat");

String lon=request.getParameter("lon");

loc = loc.replaceAll("[^a-zA-Z0-9. ]", "");

String xy[]=loc.split("\\s+");

session.setAttribute("arr",xy);

%>

<form method="post" action="upoi4.jsp">

User Predicate..

<input type="text" required  class="form-control" name="pred" placeholder="Looking for.. Ex: Hotels, Restaurants.. ">

<table align="right">

<tr><td>

How much distance you can travel..<br>

<input type="number" min="1" required   name="dist" placeholder="Distance in KM" size="27"></td>

</tr>

</table><table id="tab">

<tr><th>User Actual Position (Pos)<td><input type="text" value="<%=lat%>,<%=lat%>" required size="47" class="form-control" name="xb" readonly>

<tr><th>User dummy Position1 (Pos1)<td><input type="text" value="<%=xy[0]%>,<%=xy[1]%>" required size="47" class="form-control" name="xb" readonly>

<tr><th>User dummy Position2 (Pos2)<td><input type="text" value="<%=xy[2]%>,<%=xy[3]%>" required size="47" class="form-control" name="xb" readonly>

<tr><td><td><input type="submit"class="submit" value="Next">

</table> </form>

<%@ include file="footer.jsp"%>
```

5.3.5. Nearby PoI List Display

```
<%

String title="Response From LBS";

%>

<%@ include file="uheader.jsp"%>

<br><br><br><br>

<h1>Results</h1><br><br>

<table id="tab">

<tr><th>POI ID<th>POI Name<th>City<th>Distance<th>Map

<%@ page  import="java.sql.*" import="com.mysql.*"  import="com.ct.*"%>

<%

String data="";

String id=request.getParameter("id");

Connection con1 = DatabaseCon.getConnection();

Statement st1 = con1.createStatement();

Statement st2 = con1.createStatement();

ResultSet rs1=null;

ResultSet rs2=null;

try

{

        String sss1 = "select * from temp order by dist";

        ResultSet rs=st1.executeQuery(sss1);

        while(rs.next())

        {
```

```
rs1=st2.executeQuery("select * from poi where pid='"+rs.getString(1)+"' ");

while(rs1.next()){

%>

<tr><td><h4><%=rs1.getString(1)%><td><h4><%=rs1.getString(2)%><td><h4><%=rs1.getString
("city")%><td><%=rs.getString(2)%>

<td><a href="#"
onclick="window.open('view.jsp?lat2=<%=rs1.getString("lat")%>&&lon2=<%=rs1.getString("lon")
%>',

'newwindow', 'width=750, height=600');

setTimeout(function(){window.close()}, 10000); return false;"><h4><font size=""
color="#ee1567">View</font></a>

<%

} } }

catch(Exception e){System.out.println(e);

e.printStackTrace();}

%>

<%out.println("<script type=text/javascript>alert('Request Accepted'); </script> ");

%>

</table>

<%@ include file="footer.jsp"%>
```

5.3.6. Map display of selected place

```
<%@ page  import="java.sql.*" import="databaseconnection.*" %>

<%

String lat=request.getParameter("lat2");

String lon=request.getParameter("lon2");
```

```
lat=lat.replaceAll("D","");

lon=lon.replaceAll("D","");

%>

<form method="post" name="ff" action="#">

<input type="hidden" name="lat" value="<%=lat%>" readonly>

<input type="hidden" name="lon" value="<%=lon%>" readonly>

</form>

<!DOCTYPE html>

<html>

<head>

<script  src="http://maps.googleapis.com/maps/api/js?key=AIzaSyDAkjJOa-
Qj3Ncv13K5W4iH3sjH1sc_UPg&sensor=true">

</script>

<script>

function initialize()

{

var lati=document.ff.lat.value;

var loni=document.ff.lon.value;

var myCenter=new google.maps.LatLng(lati,loni);

var mapProp = {

  center: myCenter,

  zoom:15,

  mapTypeId: google.maps.MapTypeId.ROADMAP

  };

var mapDiv= document.getElementById("googleMap");
```

```
var map = new google.maps.Map(mapDiv,mapProp);
var marker = new google.maps.Marker({
  position: myCenter,
  map: map,
  title:'Click to zoom'
  });
marker.setMap(map);
google.maps.event.addListener(marker,'click',function() {
  map.setZoom(9);
  map.setCenter(marker.getPosition());
  });
}
google.maps.event.addDomListener(mapDiv, 'load', initialize);
</script>
<script type="text/javascript">
window.setTimeout(function alertfunc(){
        alert("Session timed out!");
        window.location = "user.html";
}, 20*60*1000);
</script>
</head>
<body> <br>
<div id="googleMap" style="width:720px;height:500px;"></div>
```

5.3.7. User Requests to LBS

```
<%
String title="Requests from User..";
%>
<%@ page  import="java.sql.*" import="com.mysql.*"  %>
<%@ include file="sheader.jsp"%>
<%
                    String message=request.getParameter("id");
                    if(message!=null && message.equalsIgnoreCase("succ"))
                    {
        out.println("<script type=text/javascript>alert('Data send to user'); </script>");
                    }
%>
<%
int c=0;
 try{
        Connection con=DatabaseCon.getConnection();
        Statement st1 = con.createStatement();
        String sss1 = "select * from urequest where status='non' ";
        ResultSet rs=st1.executeQuery(sss1);
        %>

        <table id="tab">
        <tr><th>Id<th>User<th>Predicate<th>Pos1<th>Pos2<th>Distance<th>Action</tr>
        <%
        while(rs.next())
```

```
{%>
<tr><td><%=rs.getString(1)%>
<td><%=rs.getString("name")%>
<td><%=rs.getString("pred")%>
<td><%=rs.getString("xb")%>
<br><%=rs.getString("yb")%>
<td><%=rs.getString("xt")%>
<br><%=rs.getString("yt")%>
<td><%=rs.getString("dist")%>

<td><a href="execute.jsp?id=<%=rs.getString(1)%>"><font size=""
color="#ee1567">Process</a>
<%
} }
catch(Exception e){ }
%>
</table></font>
<%@ include file="footer.jsp"%>
```

5.3.8. Database Connectivity

```
package com.mysql;
import java.sql.*;
public class DatabaseCon
{       static Connection con;
        public static Connection getConnection()
```

```
{
Try     {
                Class.forName("com.mysql.jdbc.Driver");
                con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/LBS","root","root");
        }
        catch(Exception e)
        {       System.out.println(" database error:"+e);
        }
        return con;
} }
```

5.3.10. K-Nearest Neighbours Algorithm (Using Euclidean Distance)

```
package com.ct;
public class  KNN
{
public static double distance(String s1, String s2, String s3, String s4, String unit) {
double lat1=0;double lon1=0;double lat2=0;double lon2=0;
s1=s1.replaceAll("D","");
 lat1=Double.parseDouble(s1);
s2=s2.replaceAll("D","");
 lon1=Double.parseDouble(s2);
s3=s3.replaceAll("D","");
 lat2=Double.parseDouble(s3);
s4=s4.replaceAll("D","");
```

```
 lon2=Double.parseDouble(s4);

double theta = lon1 - lon2;

double dist = Math.sin(deg2rad(lat1)) * Math.sin(deg2rad(lat2)) + Math.cos(deg2rad(lat1)) *
Math.cos(deg2rad(lat2)) * Math.cos(deg2rad(theta));

  dist = Math.acos(dist);

  dist = rad2deg(dist);

  dist = dist * 60 * 1.1515;

  if (unit == "K") {

    dist = dist * 1.609344;

          dist=dist;

          if(dist>1)

            {       dist=dist+2; }

}

else if (unit == "N") {

          dist = dist * 0.8684; }

  return (dist);

}

public static double deg2rad(double deg) {

  return (deg * Math.PI / 180.0);

}

public static double rad2deg(double rad) {

  return (rad * 180 / Math.PI);

}
```

6. TESTING

6.1. INRODUCTION

Testing is a process, to evaluate the functionality of an application with an intent to find whether the developed application met the specified requirements or not and to identify the defects to ensure that the product is defect free in order to produce the quality product.

According to ANSI/IEEE 1059 standard, Testing can be defined as - A process of analyzing a software item to detect the differences between existing and required conditions (that is defects/errors/bugs) and to evaluate the features of the software item.

6.2 TYPES OF TESTS

6.2.1. Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2. Integration Testing

Testing of all integrated modules to verify the combined functionality after integration is termed as Integration Testing. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

6.2.3.Functional testing

It is a type of software testing whereby the system is tested against the functional requirements/specifications. Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application.

6.2.3.1. White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose.
It is used to test areas that cannot be reached from a black box level.

6.2.3.2. Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. The test provides inputs and responds to outputs without considering how the software works.

6.2.4. System Testing

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.2.5. Back-end Testing

Whenever an input or data is entered on front-end application, it stores in the database and the testing of such database is known as Database Testing or Backend testing. In back-end testing GUI is not involved, testers are directly connected to the database with proper access and testers can easily verify data by running a few queries on the database. There can be issues identified like data loss, deadlock, data corruption etc during this back-end testing and these issues are critical to fixing before the system goes live into the production environment.

6.3. Performance Evaluation

6.3.1. Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3.2. Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3.3. Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3.4. Back-end Testing

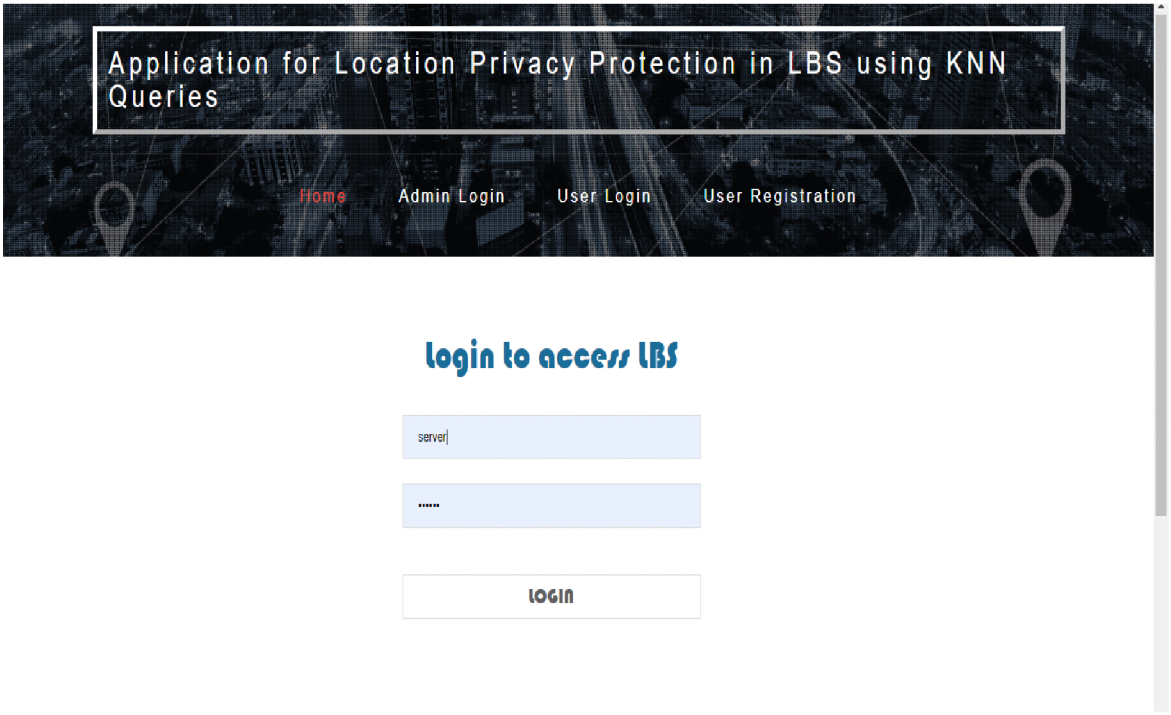
This is a vital phase as it checks for data integrity and consistency in the database. It tries to identify if data is written or not, if database is updated, if any data is corrupted etc.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

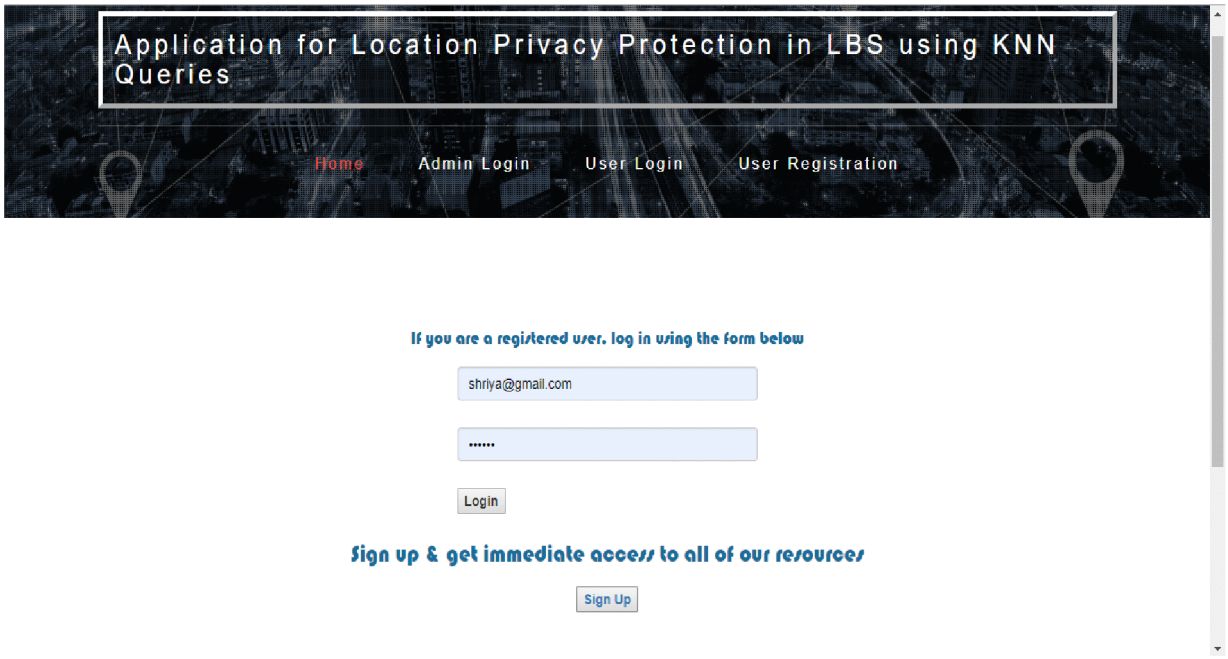
7. SCREENSHOTS

7.1. Application Home Page

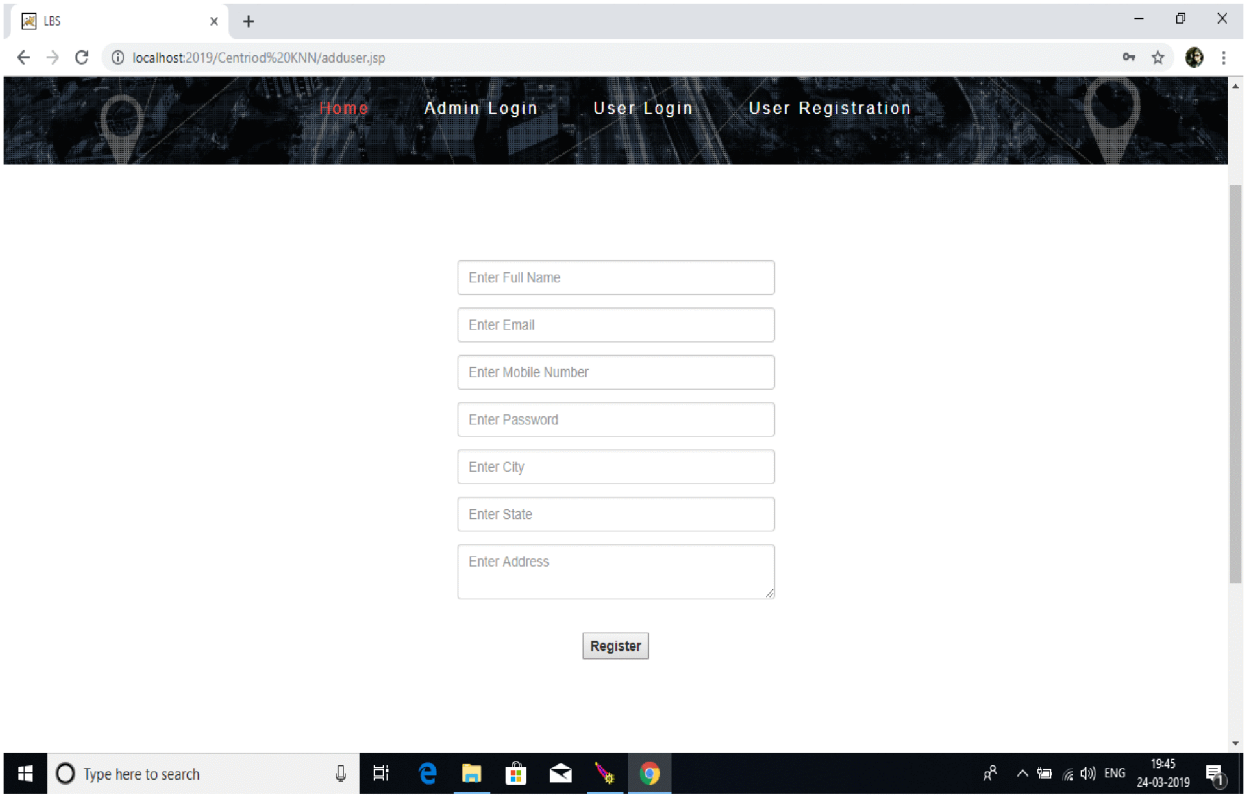
7.2. Server Login



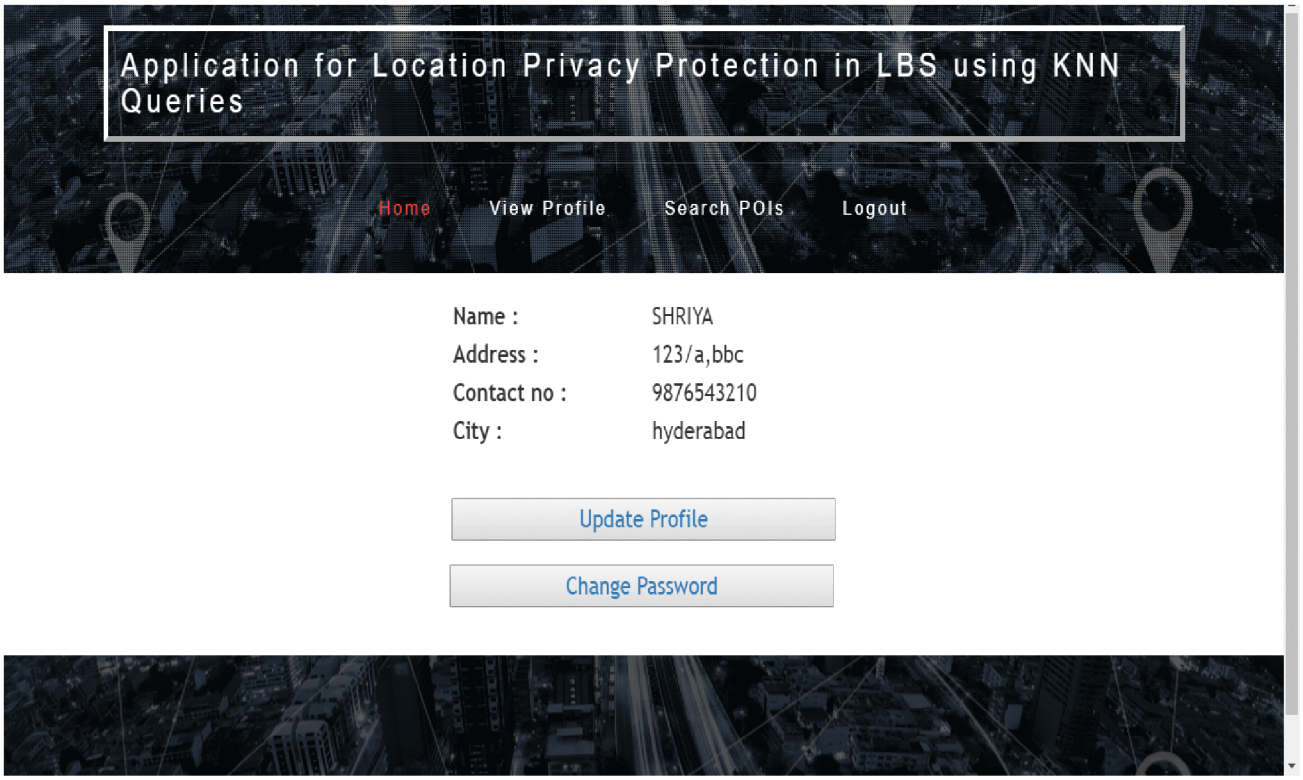
7.3. User Login



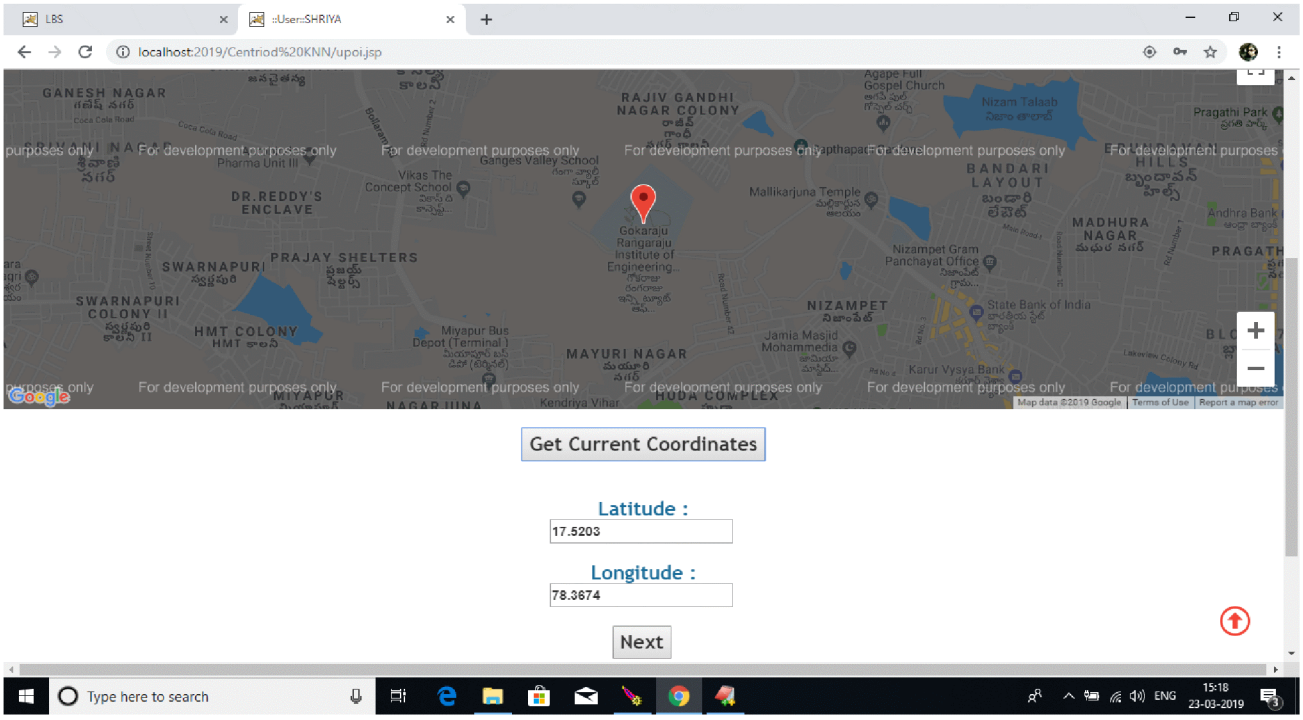
7.4. User Registration



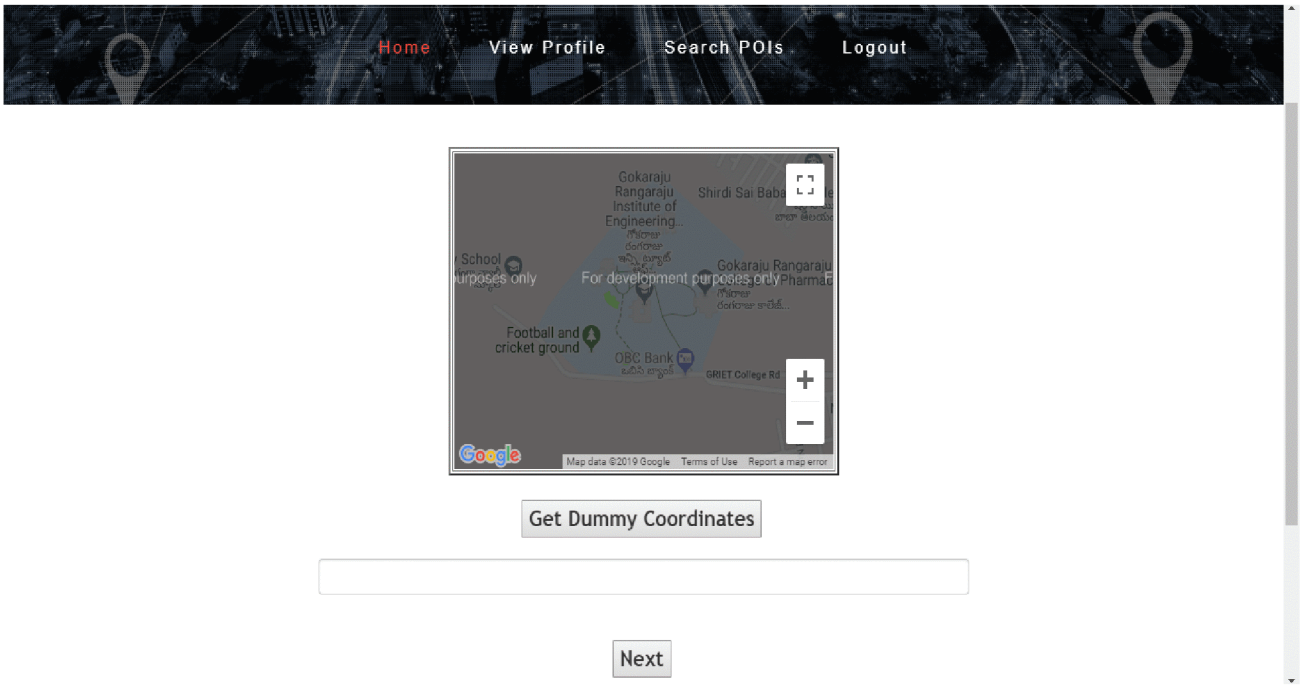
7.5 User Profile



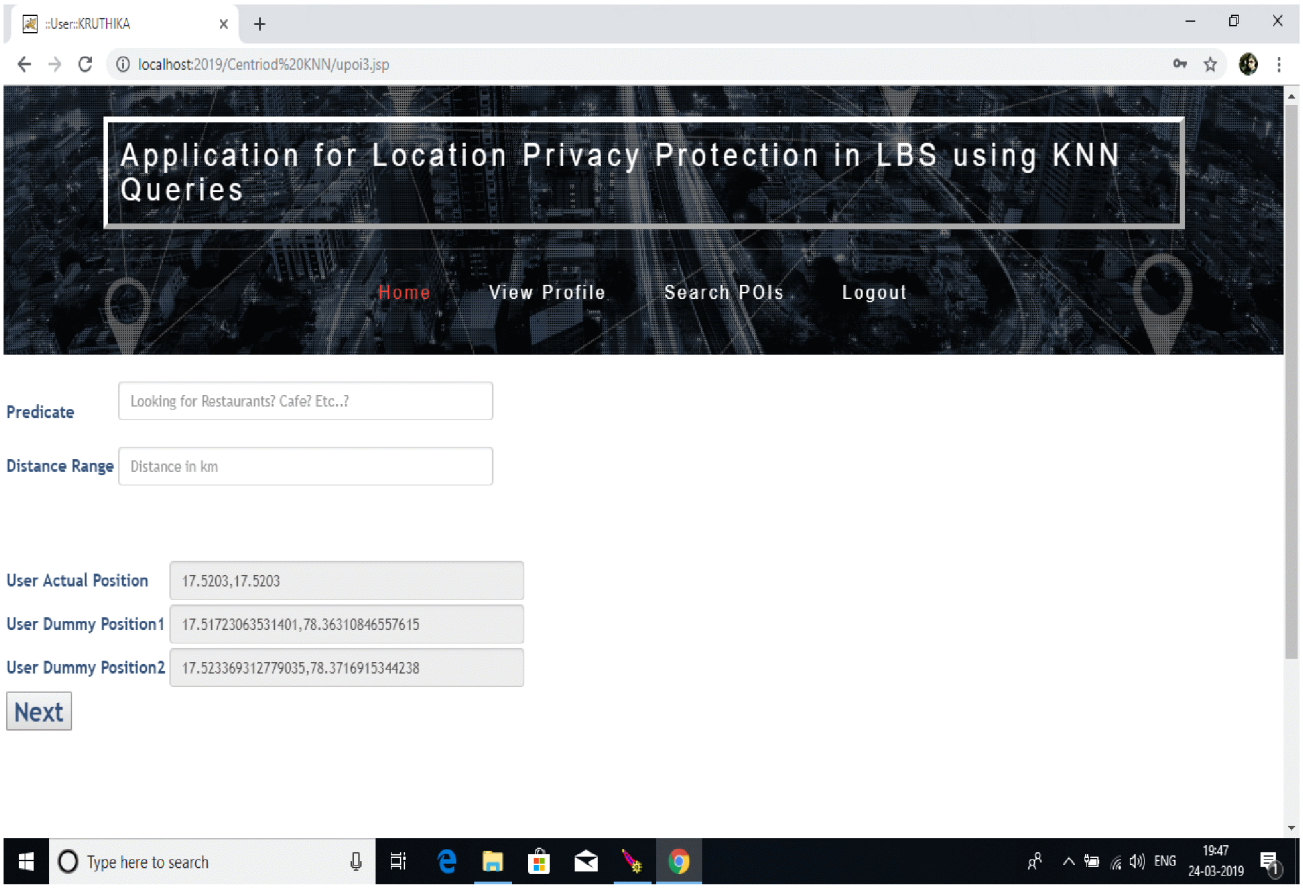
7.6 User Current Location



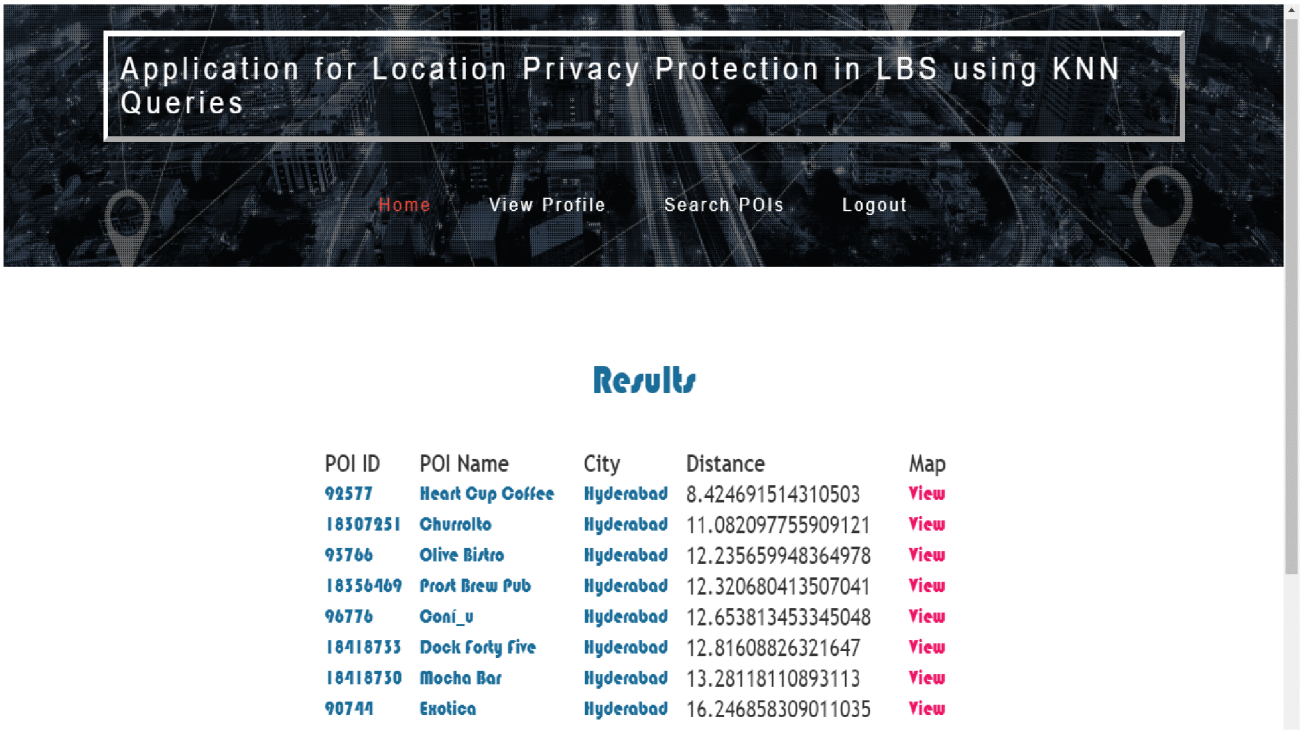
7.7. Getting Dummy Location



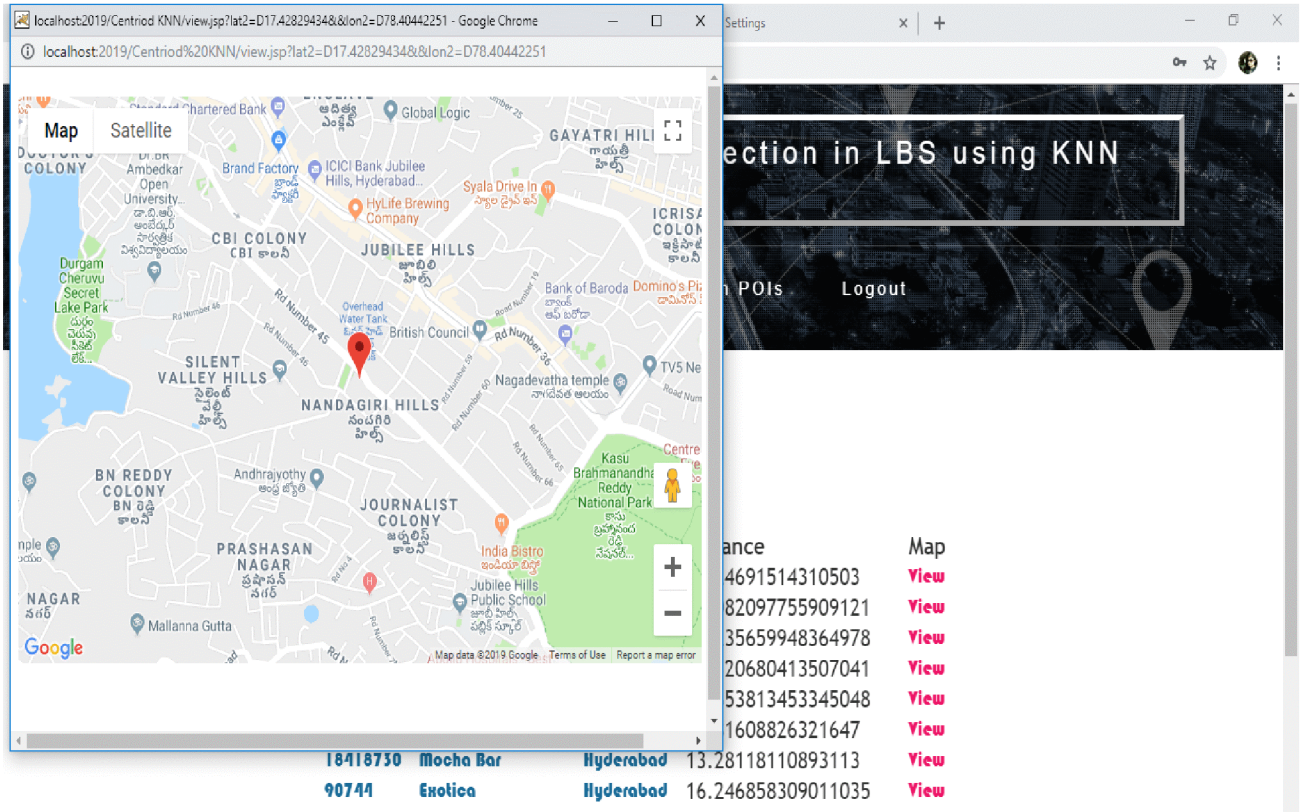
7.8 Search For PoI



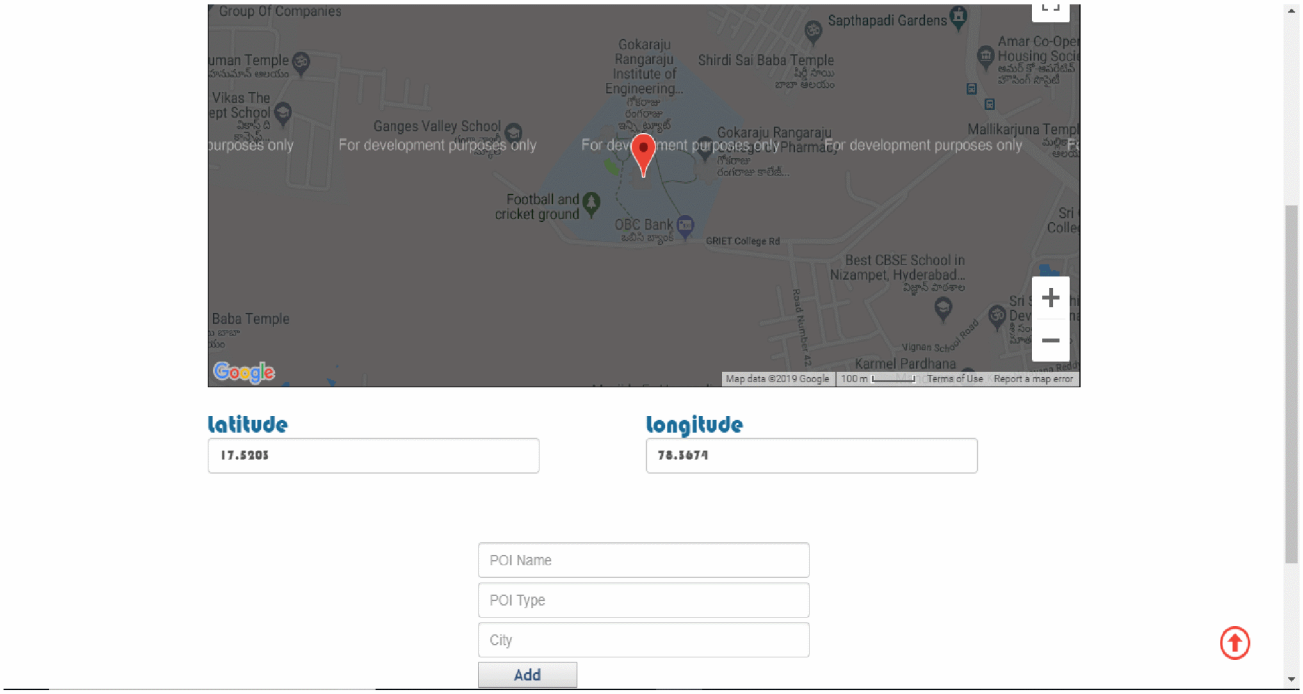
7.9. List Of Nearby PoI for User



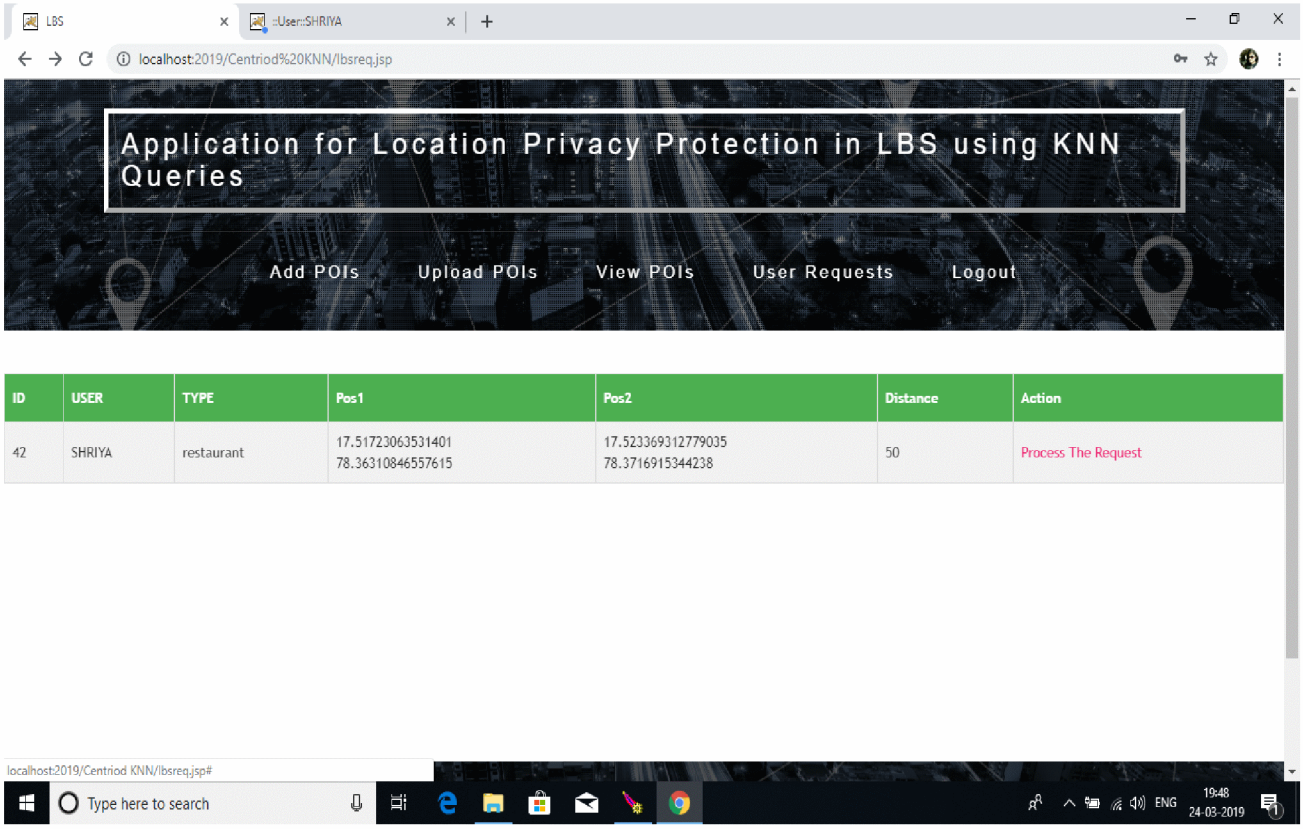
7.10. Map For Selected PoI



7.11. Adding Data Manually (Server)



7.13. User Requests Received By Server



8. CONCLUSION

In this application, we design a novel all-dummy location generation method and improve the method of the server processing requests. Notably, our proposed method can guarantee location privacy and is proved to be efficient. The experimental results show that the service request success rate is over 80%, the cost of communication between the server and the client is greatly reduced.

Hence, our application is protecting user's location privacy using the concept of centroid by applying K-nearest neighbour (KNN) query for LBSs. By doing so, we remove the necessity of a third party between the user and the location based server along with protecting the privacy of user location details.

9. BIBLIOGRAPHY

[1] Zhao, Huan, and Meng Li. "Centroid-based KNN query in mobile service of LBS." In *NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium*, pp. 1-6. IEEE, 2018.

[2] Shokri, Reza, Carmela Troncoso, Claudia Diaz, Julien Freudiger, and Jean-Pierre Hubaux. "Unraveling an old cloak: k-anonymity for location privacy." In *Proceedings of the 9th annual ACM workshop on Privacy in the electronic society*, pp. 115-118. ACM, 2010.

[3] Zhao, Huan, Jiaolong Wan, and Zuo Chen. "A novel dummy-based KNN query anonymization method in mobile services." *International Journal of Smart Home* 10, no. 6 (2016): 137-154.

[4] Gruteser, Marco, and Dirk Grunwald. "Anonymous usage of location-based services through spatial and temporal cloaking." In *Proceedings of the 1st international conference on Mobile systems, applications and services*, pp. 31-42. ACM, 2003.

[5] Kido, Hidetoshi, Yutaka Yanagisawa, and Tetsuji Satoh. "An anonymous communication technique using dummies for location-based services." In *ICPS'05. Proceedings. International Conference on Pervasive Services, 2005.*, pp. 88-97. IEEE, 2005.

[6] Mokbel, Mohamed F., Chi-Yin Chow, and Walid G. Aref. "The new casper: Query processing for location services without compromising privacy." In *Proceedings of the 32nd international conference on Very large data bases*, pp. 763-774. VLDB Endowment, 2006.

[7] Gedik, Bugra, and Ling Liu. "Protecting location privacy with personalized k-anonymity: Architecture and algorithms." *IEEE Transactions on Mobile Computing* 7, no. 1 (2008): 1-18.

[8] Beizer, Boris. *Software testing techniques*. Dreamtech Press, 2003.

[9] Kotonya, Gerald, and Ian Sommerville. *Requirements engineering: processes and techniques*. Wiley Publishing, 1998.

[10] Duckett, J. "HTML & CSS, Design and Build Websites. Jon Wiley & Sons." *Inc., Indianapolis* (2011).

[11] Duckett, Jon. *JavaScript and JQuery: interactive front-end web development*. Wiley Publishing, 2014.

[12] https://developers.google.com/maps/documentation/

[13] https://www.w3schools.in/category/software-testing/

[14] https://www.w3schools.in/category/javascript-tutorial/