U R Web

Autori:
Amarandei Sonia-Angela 2B3
Anghelache Oana-Iuliana 2B3
Cristei Ioan-Daniel 2B6
Manea Petru-Mircea 2B6

Cuprins:

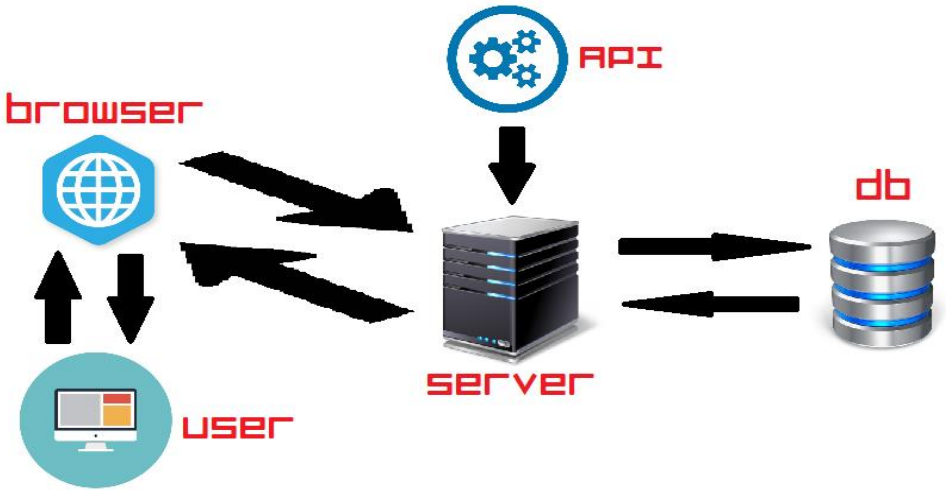
1. Descriere

Luand in consideratie faptul ca fiecare utilizator este identificat unic via un dispozitiv personal (e.g., telefon mobil) sau a unui marcaj (cod de tip QR, RFID,...), sa se dezvolte o aplicatie a prezenta si/sau sugera in mod "inteligent" resurse -- cunoscuti, obiective de interes (benzinarii, magazine, parcuri, morgi etc.), informatii de ghidare si altele -- disponibile in imediata vecinatate conform serviciilor publice de geo-localizare existente. Aplicatia Web va fi suficient de modulara astfel incat sa ofere noi functionalitati pe baza unor extensii (module ce pot fi incarcate dinamic). Interogarile vor putea fi realizate pe baza unui API REST, iar interactiunea cu utilizatorul va fi una cat mai naturala.

2. Arhitectura

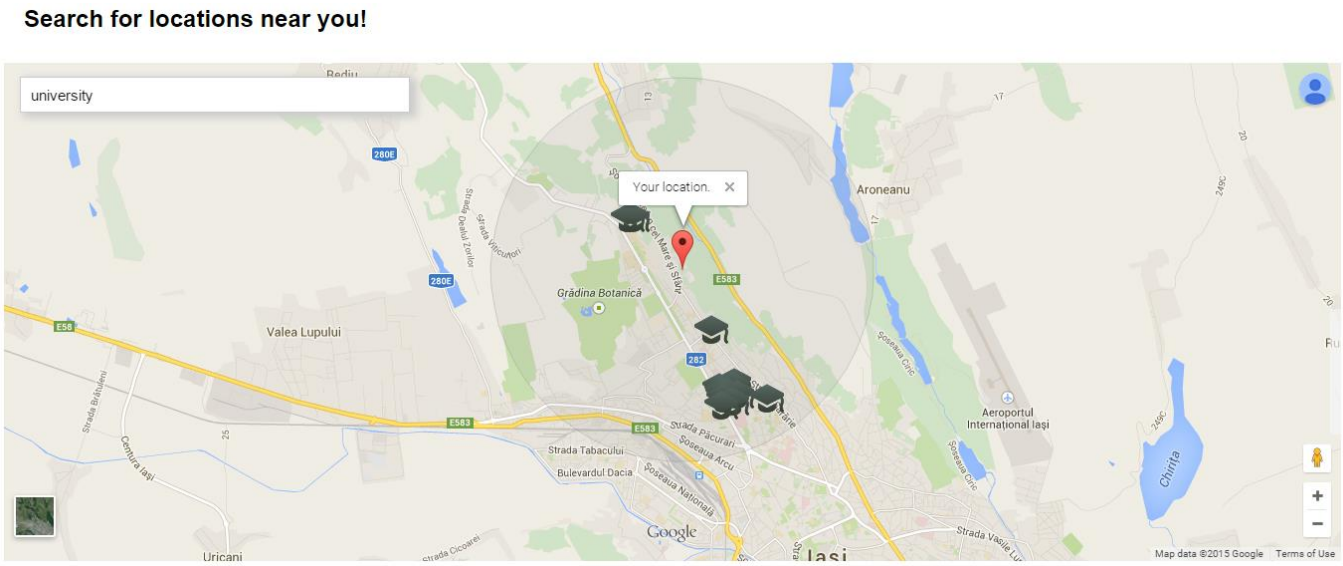
2.1. Design/Idei

Arhitectura aplicatiei va urma modelul MVC (Model-View-Controller).



Astfel View-ul va fi reprezentat de catre site-ul web aflat pe browser, acesta interactionand cu uitilizatorul si oferindu-i o interfata atractiva si usor de utilizat pentru a obtine, transmite si afisa informatiile necesare.

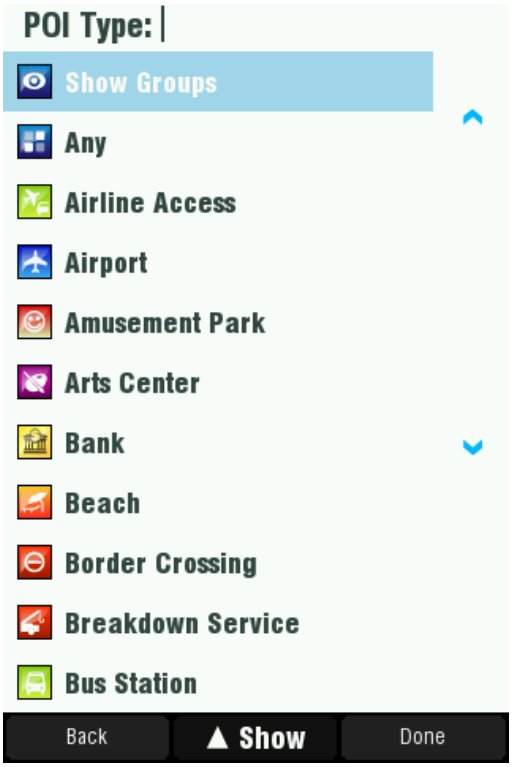
Acesta va fi creat utilizand HTML5, CSS3 si JavaScript. Se vor permite selectarea locatiilor predefinite (banci, parcuri etc.), acestea aparand in raza de actiune a utilizatorului sub forma unor iconite. Se va permite si cautarea unor locatii si vizualizarea pozitiei utilizatorilor logati cu aceasta aplicatie.



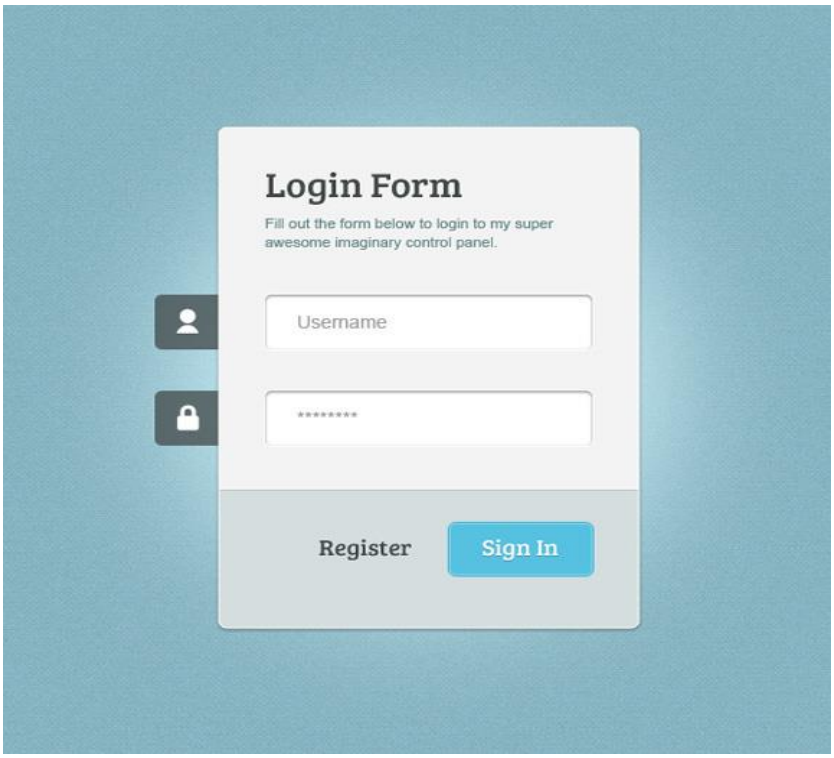
Modelul va fi reprezentat de catre server, acesta fiind creat in PHP si interactionand cu baza de date, fiind creata in Oracle PL\SQL. Acesta va primi informatii de la diferite API-uri precum Google (maps), Facebook, Twitter etc. Aceste informatii vor fi utilizate pentru geolocalizare si detectarea interactiunii cu ceilalti utilizatori.

Vor exista paginile de LogIn, CreateAccount, SearchLocations etc.

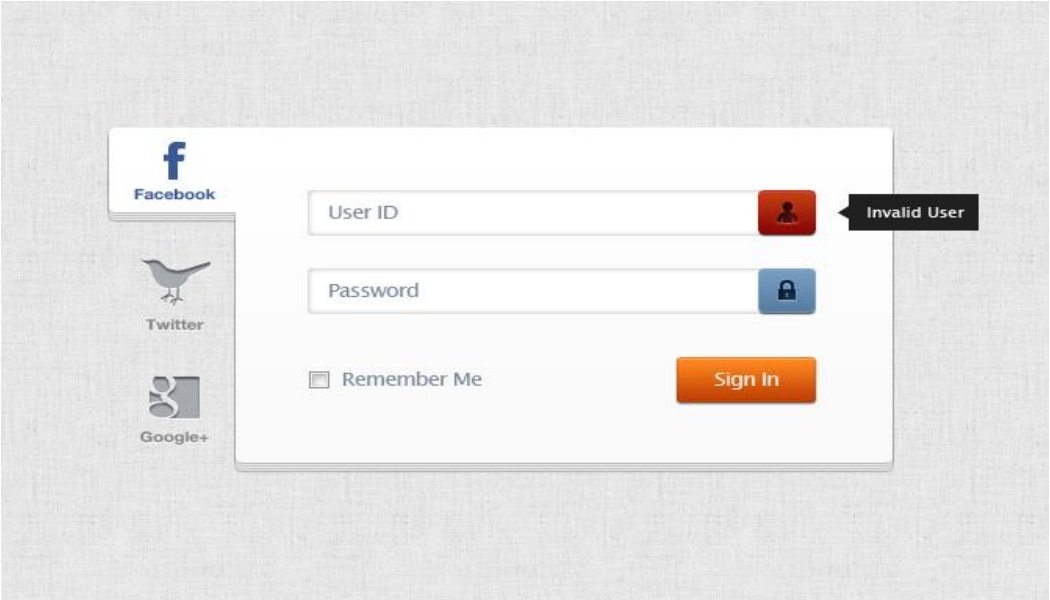
Se vor putea selecta punctele de interes. Acestea vor fi afisate sub forma unei liste avand langa fiecare element un switch button ce va permite vizualizarea acestor puncte. Va mai exista un buton care va genera o ruta catre acea locatie selectata.



Pagina de log in va contine un formular de logare avand un text box pentru username, altul pentru parola si doua butoane “Register”, ce va redirectiona catre pagina de inregistrare, si butonul “LogIn” ce va trimite catre pagina principala de cautare.



Fiecare utilizator se va putea loga si folosind conturile de Facebook, Twitter si Google+. Acestea vor redirectiona catre paginile respective pentru a cere drepturile de acces la datele utilizatorului.

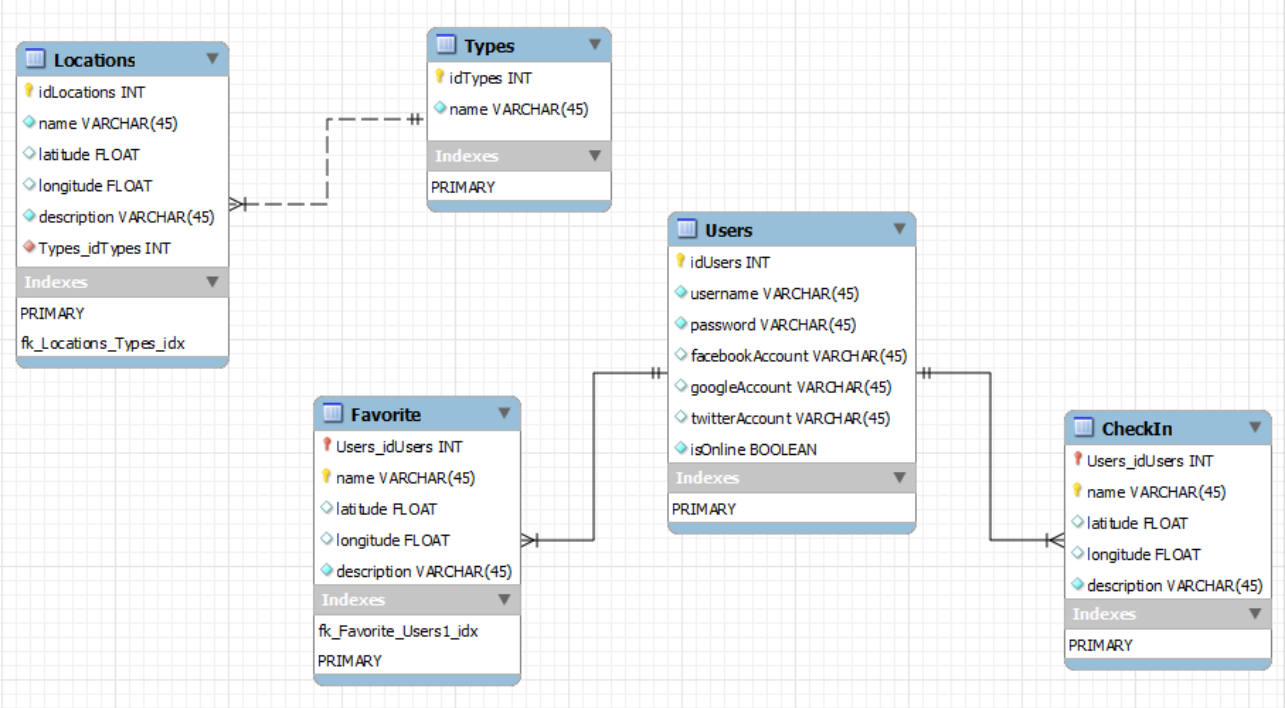


Check-in-urile vor fi gasite pe baza acestor conturi asociate.

Aceste date vor fi stocate intr-o baza de date ce va fi updatata constant. Baza de date va contine urmatoarele tabele.

- **Locations** ;
  - Locations_ID (primary key)
  - Name
  - *Latitude(optional)*
  - *Longitude(optional)*
  - Description
  - Types_ID(foreign key)
- **Users** ;
  - Users_ID (primary key)
  - Username
  - Password
  - FacebookAccount
  - GoogleAccount
  - TwitterAccount
  - IsOnline (yes/no)
- **Types** ;
  - Types_ID (primary key)
  - Name
- **CheckIn** ;
  - Users_ID (foreign key)
  - Name *(strada)*
  - Latitude*(a strazii)*
  - Longitude*(a strazii)*
  - Description
- **Favorite** ;
  - Users_ID(foreign key)
  - Name
  - Latitude
  - Longitude
  - Description

- Alte tabele vor putea fi adaugate ulterior in timpul dezvoltarii aplicatiei.



Tabelul Locations

Va contine locatiile aflate in raza de actiune a utilizatorului. Acesta va fi un tabel de dimensiuni mari, continand multe intrari si va fi updatat constant: daca o locatie intra in raza de actiune a utilizatorului atunci aceasta va fi adaugata in tabel, iar daca o locatie iese din raza de actiune a utilizatorului atunci aceasta va fi stearsa din tabel.

Tabelul Users

Va contine toti utilizatorii care au creat un cont pe aceasta aplicatie si de aseamenea toate accounturile asociate (Google, Facebook, Twitter etc.). Acesta va fi updatat constant la crearea unui nou cont. Prin intermediul interogarilor se verifica daca un user este in raza de actiune a altui user. Utilizatorii vor fi tratati drept locatii obisnuite.

Tabelul Types

Va contine toate tipurile de categorii de locatii aflate pe API-ul maps. Fiecare categorie va contine cateva zeci de locatii din raza mea de actiune, iar utilizatorul va putea selecta ce categorii doreste sa vizualizeze prin utilizarea a switch-button-urilor. La selectarea unui switch se va face o interogare pe tabelul locatii. Acestea vor fi luate de pe https://developers.google.com/places/supported_types.

Tabelul CheckIn

Va contine toate check-in-urile oferite de diversele aplicatii asociate conturilor. Astfel check-in-urile vor fi tratate sub forma unor locatii obisnuite, dar vor aparea doar in raza de actiune a utilizatorului ce are conturile asociate. Tabelul va fi updatat la adaugarea unui check-in, dar fiecare check-in mai vechi de 1-2 zile va fi scos din baza de date.

Tabelul Favorite

Va contine locatiile favorite de cautare si vizualizare a utilizatorilor. Acesta va fi updatat constant in functie de fiecare utilizator in parte.

Indecsi

Pentru accesarea rapida a tipurilor de locatii sau a unei locatii anume vor fi creati indecsi in tabela Locatii pe coloanele **Type** si **Name**. Astfel vor putea fi accesate foarte rapid cautarea si tipurile de locatii dorite pentru a fi afisate de catre View.

Pentru accesarea rapida a locatiei utilizatorilor logati va fi creat un index pe coloana idUsers, ce va permite interactiunea rapida intre tabelele Users, CheckIn si Favorites.

Incarcarea dintr-un CSV sau XML

Tabelele Locatii si CheckIn vor fi incarcate local, pentru a putea fi diferite de la utilizator la utilizator, astfel permitandu-se o interactiune mai buna intre ele.

Preluarea datelor dintr-un formular al aplicatiei

Crearea unui nou cont sau logarea vor fi facute pe baza unui formular ce va transmite datele, acestea fiind verificate cu cele din tabelul Users.

Triggere

Fiecare tabel va dispune de triggere pentru autoincrementarea id-ului si de triggere pentru inserarea, updatarea si stergerea datelor din acestea.

Astfel tabelul Locations va dispune de triggere pentru adaugarea datelor ce intra in raza de actiune a utilizatorului si iesirea din raza de actiune a utilizatorului, precum si cel de stergere a datelor ce ies din raza de actiune a utilizatorului.

Tabelul Users va dispune de trigger ce va updata constant locatia utilizatorilor logati.

Tabelul CheckIn va dispune de triggere pentru adaugarea check-in-urilor aparute si stergerea acestora la depasirea a 24h de la adaugarea lor.

Deoarece fiecare trigger va updata automat tabelele sau la randul lui va apela alt trigger, datele nu vor ajunge sa fie corupte sau inconsistente, nici macar la efectuarea simulatana a tranzactiilor.

View-uri

Vor fi create view-uri pentru a permite selectarea diferitelor locatii din raza de actiune a utilizatorilor. Fiecare locatie corespunde unui anumit tip, iar baza de date va avea view-uri pentru a selecta aceste date din tabela Locations.

Exceptii

Fiecare adaugare de cont nou sau de logare incorecta, continerea de date corupte in fisierele pentru tabelele Locations si CheckIn, vor genera exceptii specifice ce vor fi prinse de aplicatia principala si afisate corespunzator.

Pachete

Functiile, procedurile, triggerele, view-urile si exceptiile asociate fiecarui tabel se vor afla intr-un pachet specific acelui tabel. Astfel, se va permite lucrul facil cu datele si organizarea acestora.

2.2. Cod relevant

HTML5

```
<!DOCTYPE html>

<html lang="en">

<head>

   <title>U R Web</title>

   <meta name="viewport" content="initial-scale=1.0, user-scalable=no">

   <meta charset="utf-8">

   <script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>

    <script
src="https://maps.googleapis.com/maps/api/js?v=3.exp&signed_in=true&libraries=places"></script>

    <script src="geolocation.js"></script>

    <link rel="stylesheet" href="style.css">

    <link href='http://fonts.googleapis.com/css?family=Special+Elite' rel='stylesheet' type='text/css'>

</head>


  <body>


   <div class="mainWrap">

      <h2 class="pageTitle">Search for locations near you!</h2>
```

```
  </div>

  <div class="mapContainer mainWrap">
    <div id="map-canvas"></div>
    <input id="locationInput" class="controls" type="text" placeholder="Search Box">
    <div id="popupNoSearchResults">
      <p>There are no such locations near you!</p>
      <input type="button" class="popupCloseButton" value="Close" onclick="closePopupWindow()">
    </div>
  </div>
 </body>
</html>
```

CSS3

```
        html, body {
          height: 100%;
          width: 100%;
          margin: 0;
          padding: 0;
          position: relative;
        }

        .mainWrap {
          width: 70%;
          margin: 0 auto;
          font-family: 'Lato', Helvetica, sans-serif;
```

```
}

.pageTitle {
  margin: 0;
  padding: 20px 0 0 20px;
}

.mapContainer {
  position: relative;
  top: 30px;
  height: 500px;
}

#map-canvas {
  position: relative;
  height: 100%;
  width: 100%;
}

#locationInput {
  position: absolute;
  width: 380px;
  height: 32px;
  font-size: 15px;
  border: 1px solid #CCC;
```

```
  top: 14px;
  left: 16px;
  padding-left: 10px;
  -moz-box-shadow:    5px 3px 13px 2px #ccc;
  -webkit-box-shadow: 5px 3px 13px 2px #ccc;
  box-shadow:         5px 3px 13px 2px #ccc;
}

#locationInput:focus {
  outline: none;
  border: 1px solid #4d90fe;
}

#popupNoSearchResults {
  opacity: 0;
  display: none;
  position: absolute;
  top: 14px;
  left: 418px;
  width: 200px;
  height: 100px;
  border-radius: 5px;
  border: 1px solid #CCC;
  background-color: white;
  padding: 10px;
```

```
  -moz-box-shadow:    1px 3px 21px -3px #000;
  -webkit-box-shadow: 1px 3px 21px -3px #000;
  box-shadow:         1px 3px 21px -3px #000;

  transition: opacity .5s ease-in-out;
  -moz-transition: opacity .5s ease-in-out;
  -webkit-transition: opacity .5s ease-in-out;
}

.popupCloseButton {
  width: 63px;
  height: 30px;
  border: 0px;
  padding: 0 10px;
  margin: 5px 0px;
  border-radius: 1px;

  cursor: pointer;
  color: #fff;
  background: #7aa76d;
  font-size: 14px;
  outline: none;
}
```

```
.popupCloseButton:hover {
  background: #7aa79f;
}

#popupNoSearchResults p {
  font-size: 16px;
}
```

JavaScript

```
var map;
var infowindow;
var pos;
var inputSearch;
var markersFound = [];
var popupClose;
var searchBox;
var CurentLat;
var CurentLong;

function initialize() {
  var mapOptions = {
              zoom: 14
  };

  map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

  // Try HTML5 geolocation
```

```
 if(navigator.geolocation) {


                    navigator.geolocation.getCurrentPosition(function(position) {

                     pos = new google.maps.LatLng(position.coords.latitude,
position.coords.longitude);

    CurentLat = position.coords.latitude;

    CurentLong = position.coords.longitude;


    var myRadius = new google.maps.Circle({

    center: pos,

    radius: 2500,

    strokeColor: "gray",

    strokeOpacity:0.3,

    strokeWeight:1,

    fillColor:"gray",

    fillOpacity:0.1

    });

    myRadius.setMap(map);


                    var marker = new google.maps.Marker({

                            map: map,

                            position: pos,

                            content: 'My location',

     animation:google.maps.Animation.DROP

                    });
```

```
                        google.maps.event.addListener(marker, 'click', function() {
                                infowindow.setContent("Your location.");
                                infowindow.open(map, this);
                        });

                    map.setCenter(pos);

                        inputSearch = document.getElementById('locationInput');

                        searchBox = new google.maps.places.SearchBox(inputSearch);
                        google.maps.event.addListener(searchBox, 'places_changed',
searchLocations);

                    }, function() {
                      handleNoGeolocation(true);
                    });

  } else {
            // Browser doesn't support Geolocation
            handleNoGeolocation(false);
  }

}
```

```
function searchLocations() {
  var request = {
  location: pos,
  radius: 2000,     // 2000 m = 2 km !
  types: [inputSearch.value]
  };

  if(markersFound.length > 0){
    removeLocationMarkers();
  }

  infowindow = new google.maps.InfoWindow();
  var service = new google.maps.places.PlacesService(map);
  service.nearbySearch(request, callback);
}



function codeAddress() {
  var geocoder = new google.maps.Geocoder();
  var address = inputSearch.value;

  geocoder.geocode( { 'address': address}, function(results, status) {
    if (status == google.maps.GeocoderStatus.OK) {
      for(var i =0; i< results.length; i++){
```

```
      if(distance(results[i].geometry.location.lat(), results[i].geometry.location.lng(),CurentLat,
CurentLong)<=2){
        map.setCenter(results[i].geometry.location);
        var marker = new google.maps.Marker({
        map: map,
        position: results[i].geometry.location
        });
        markersFound.push(marker);
        var markerAddress = results[0].formatted_address;

        google.maps.event.addListener(marker, 'click', function() {
          infowindow.setContent("Location found:<br />"+markerAddress);
          infowindow.open(map,this);
        });
      }
    }
  }

  if(!status == google.maps.GeocoderStatus.OK || markersFound.length ==0) {
    clearTimeout(popupClose);
    $("#popupNoSearchResults").css("opacity", 1);
    $("#popupNoSearchResults").css("display", "block");
    popupClose = setTimeout(function(){
      $("#popupNoSearchResults").css("opacity", 0);
      $("#popupNoSearchResults").css("display", "none");
    }, 2000);
```

```
    }


  });
}




function distance(lat1,lon1,lat2,lon2)
{
  var R = 6371; // Radius of the earth in km
  var dLat = deg2rad(lat2-lat1);  // deg2rad below
  var dLon = deg2rad(lon2-lon1);
  var a = Math.sin(dLat/2) * Math.sin(dLat/2) + Math.cos(deg2rad(lat1)) *
Math.cos(deg2rad(lat2)) * Math.sin(dLon/2) * Math.sin(dLon/2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c;
  return d;   // Distance in km !!
}

function deg2rad(deg) {
  return deg * (Math.PI/180);
}




function removeLocationMarkers() {
```

```
    for(var i = 0; i < markersFound.length; i++) {
        markersFound[i].setMap(null);
    }
    markersFound = [];
}



function callback(results, status) {
    if (status == google.maps.places.PlacesServiceStatus.OK) {
      for (var i = 0; i < results.length; i++) {
        createMarker(results[i]);
      }
    } else {
      codeAddress();
    }
}



function createMarker(place) {
    var pinIcon = new google.maps.MarkerImage(
      place.icon,
      null, /* size is determined at runtime */
```

```
    null, /* origin is 0,0 */
    null, /* anchor is bottom center of the scaled image */
    new google.maps.Size(34, 34)
   );

   var marker = new google.maps.Marker({
    map: map,
    position: place.geometry.location,
    icon: pinIcon
   });

   markersFound.push(marker);

   google.maps.event.addListener(marker, 'click', function() {
    infowindow.setContent(place.name + "<br />" + "Address: " + place.vicinity + "<br />" +
"Rating: " +place.rating);
    infowindow.open(map, this);
   });
}

function closePopupWindow() {
   clearTimeout(popupClose);
```

```
    $("#popupNoSearchResults").css("opacity", 0);
    $("#popupNoSearchResults").css("display", "none");
}




function handleNoGeolocation(errorFlag) {
  if (errorFlag) {
    var content = 'Error: The Geolocation service failed.';
  } else {
    var content = 'Error: Your browser doesn\'t support geolocation.';
  }

  var options = {
    map: map,
    position: new google.maps.LatLng(60, 105),
    content: content
  };

  var infowindow = new google.maps.InfoWindow(options);
  map.setCenter(options.position);
}
google.maps.event.addDomListener(window, 'load', initialize);
```

PL\SQL

```
DROP TABLE Locations ;


CREATE TABLE Locations (
 idLocatii NUMBER(10) NOT NULL,
 name VARCHAR2(45) NOT NULL,
 latitude NUMBER(10,10) NULL,
 longitude NUMBER(10,10) NULL,
 idTypes NUMBER(10) NOT NULL,
 description VARCHAR2(45) NOT NULL
 );


CREATE INDEX Locations_Type ON Locations (idTypes);


CREATE INDEX Locations_Name ON Locations (name);



DROP TABLE Users ;


CREATE TABLE Users (
 idUsers NUMBER(10) NOT NULL,
 username VARCHAR2(45) NOT NULL,
 password VARCHAR2(45) NOT NULL,
 description VARCHAR2(45) NOT NULL,
 facebookAccount VARCHAR2(45) NULL,
 googleAccount VARCHAR2(45) NULL,
```

```
 twitterAccount VARCHAR2(45) NULL,
 isOnline BOOLEAN NOT NULL
 );


DROP TABLE Types ;

CREATE TABLE Types (
 idTypes NUMBER(10) NOT NULL,
 name VARCHAR2(45) NOT NULL
 );

DROP TABLE CheckIn ;

CREATE TABLE CheckIn (
 idUsers NUMBER(10) NOT NULL,
 name VARCHAR2(45) NOT NULL,
 latitude NUMBER(10,10) NULL,
 longitude NUMBER(10,10) NULL,
 description VARCHAR2(45)  NOT NULL
 );

CREATE INDEX fk_CheckIn_idUsers ON CheckIn (idUsers);

DROP TABLE Favorite ;
```

```
CREATE TABLE Favorite (
 idUsers NUMBER(10) NOT NULL,
 name VARCHAR2(45) NOT NULL,
 latitude NUMBER(10,10) NULL,
 longitude NUMBER(10,10) NULL,
 description VARCHAR2(45)  NOT NULL
 );

CREATE INDEX fk_Favorite_idUsers ON Favorite (idUsers);

CREATE OR REPLACE PROCEDURE populeazaLocatii
 (v_name IN OUT VARCHAR2, v_latitude IN OUT NUMBER, v_longitude IN OUT NUMBER, v_type IN OUT VARCHAR2, v_description IN OUT VARCHAR2, v_rating IN OUT NUMBER)
 IS
BEGIN
    INSERT INTO Locatii(name,latitude,longitude,type,description,rating) VALUES (v_name,v_latitude,v_longitude,v_type,v_description,v_rating);
END populeazaLocatii;

CREATE OR REPLACE PROCEDURE isOn(categorie IN Locations.type%type )
 IS
BEGIN
 Select name from Locations where type = categorie;
END isOn;
```