Aplikacija za dohvat i pohranu zemljopisnih koordinata

Dragić, Anna-Maria

Undergraduate thesis / Završni rad

2017

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:001588

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-26**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

APLIKACIJA ZA DOHVAT I POHRANU ZEMLJOPISNIH KOORDINATA

Završni rad

Anna-Maria Dragić

Osijek, 2017.

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. KORIŠTENE TEHOLOGIJE 2

2.1. HTML 2

2.2. CSS 3

2.3. JavaScript 4

2.4. PHP 5

2.5. Google Maps API 6

3. PRVA REALIZACIJA APLIKACIJE 8

3.1. Prikaz mape i rezultata korisniku 8

3.2. Dohvaćanje podataka sa Google mape i ispis tih podataka 9

3.3. Prikaz funkcioniranja prve realizacije aplikacije 10

4. DRUGA REALIZACIJA APLIKACIJE 12

4.1. Prikaz tražilice i ispis rezultata 12

4.2. Kreiranje baze podataka 16

4.3. Spremanje podataka u bazu 17

4.4. Prikaz funkcioniranja druge realizacije aplikacije 19

5. DIZAJN 21

6. ZAKLJUČAK 24

LITERATURA 25

SAŽETAK 26

ABSTRACT 27

ŽIVOTOPIS 28

PRILOZI 29

1. UVOD

U današnje vrijeme kada se jako puno putuje, vrlo je važno i korisno znati zemljopisne koordinate adrese na koju se zaputimo kako bismo što brže i lakše došli na željenu adresu. Uz pomoć nekih web tehnologija može se realizirati aplikacija za dohvat zemljopisnih koordinata pomoću koje bi korisnik mogao saznati podatke o željenoj lokaciji. Upravo takvu aplikaciju ćemo relizirati u ovom radu. Aplikacija je temeljena na web tehnologijama kao što su JavaScript, HTML, CSS, PHP i sl. koje su detaljnije pojašnjene u sljedećem poglavlju.

Zadatak završnog rada je dohvat podataka (zemljopisnih koordinata) te njihovo spremanje u bazu podataka. Izvršava se dohvaćanje podataka (zemljopisnih koordinata) od Google karte (Google maps) koristeći pri tome Geocoding Api. Aplikacija je izvedena na dva načina, prva realizacija koja je objašnjena u poglavlju 3 funkcionira na način da kada korisnik klikne bilo gdje na mapi, prikazat će mu se zemljopisne koordinate i nadmorska visina odabrane lokacije. Druga realizacija (opisana u poglavlju 4) je takva da je korisniku omogućeno upisivanje adrese za koju želi dobiti podatke, te potom ga aplikacija odvodi do te adrese na mapi i ispisuje mu zemljopisne koordinate te lokacije. Kod druge realizacije aplikacije ostvareno je i spremanje dohvaćenih podataka pri čemu je prvobitno kreirana baza podataka i tablica u koju spremamo rezultate. Korisnik ima opciju spremanja podataka ukoliko klikne na tipku „Pošalji“, a ukoliko ne klikne na tipku podatci neće biti spremljeni. Dakle, nakon klika na „Pošalji“ u bazu se spremaju adresa koju je korisnik unio te zemljopisna dužina i širina i tip te lokacije. Potom je u radu prikazano po koracima kako funkcioniraju obadvije realizacije.

1.1. Zadatak završnog rada

Zadatak završnog rada je realizirati web aplikaciju koja će vršiti dohvat zemljopisnih koordinata bilo koje lokacije na svijetu koju korisnik izabere, te će potom dohvaćene podatke spremati u unaprijed kreiranu bazu podataka.

2. KORIŠTENE TEHOLOGIJE

Prilikom razvoja aplikacije korištene su web tehnologije, kao što su JavaScript, HTML, CSS,PHP. Također korištena je i Google Maps API pomoću koje su se dohvaćali podatci tj. zemljopisne koordinate. Potrebno je prvo objasniti korištene tehnologije kako bi poslije u radu i kodu bili razumljivi korištene oznake i pricip rada.

2.1. HTML

HTML je standradni prezentacijski jezik za izradu web stranica. Kratica HTML znači Hyper Text Markup Language. Web dokument je tekstualni dokument i kao takav može se uređivat u bilo kojem tekst editoru. Sadržaj se piše unutar oznaka (engl. *tag*). Sadržaj koji se unosi određuje koja će se oznaka koristiti. HTML oznake označavaju dijelove sadržaja kao što su "naslov", "odlomak", "tablica" itd. Internet preglednik prilikom učitavanja web dokumenta interpretira tagove pa se sukladno tome uređuje njihov sadržaj.

HTML oznake (tags) su nazvi elemenata okruženi kutnim zagradama. HTML oznake su većinom u parovima,kao što su i . Prva oznaka u paru je početna oznaka, druga oznaka je krajnja oznaka. Početnu oznaku također se zovemo oznaka za otvaranje, a krajnju oznaku završna oznaka.

Deklaracija <!DOCTYPE html> predstavlja tip dokumenta i pokaže web pregledniku da ispravno prikaže web stranicu. Pojavljuje se samo jednom, na početku koda prije bilo koje HTML oznake. Svaki HTML dokument mora početi s tom deklaracijom.

Sam HTML dokument počinje s <html> i završava s </ html>. Vidljivi dio HTML dokumenta je između oznaka <body> i </body>. HTML naslovi su definirani oznakama <h1> do <h6>. <h1> definira najvažniji naslov, dok <h6> definira najmanje važan naslov. oznaka definira paragraf.

HTML tablica definirana je oznakom <table>. Svaki redak tablice definiran je oznakom <tr>. Zaglavlje tablice definirano je s oznakom <th>. Prema zadanim postavkama zaglavlja tablice su podebljana i usmjerena. Tablica podataka / ćelija definirana je oznakom <td>.

HTML veze su hiperveze. Možete kliknuti vezu i otići(skočiti) na drugi dokument. Svaki web dokument ima svoj URL (engl. *Uniform Resource Locator*), odnosno jedinstveni identifikator lokacije po kojem se razlikuje od klasičnih elektronskih dokumenata. U HTML-u veze su definirane pomoću <a> oznake: link text [1].

2.2. CSS

CSS je meta-jezik kojim se uređuje i definira izgled dokumenta kreiranih bilo kojim markup jezikom. Kratica CSS označava Cascading Style Sheets. Uređivanje i izgled dokumenata moguće je definirati u bilo kojem od tekstualnih editora. CSS se koristi za definiranje stilova za web stranice, uključujući dizajn, izgled i varijacije na zaslonu za različite uređaje i veličine zaslona. Izgled web dokumenata dobiva se tako da se elementima ili grupama elemenata dodjeljuju stilovi kao što su: boja, pozicija na stranici, font, visina,... Pri tome se prezentacija stranice odvaja od njezinog sadržaja, a HTML kod postaje neusporedivo čitkiji i manji. CSS je donio čitav niz načina na koje je uređivanje prikaza podataka koji dotada nisu postojali u HTML-u. S CSS-om je moguće jednostavnom promjenom nekoliko parametara promijeniti cjelokupni izgled stranice. CSS štedi puno posla. On može kontrolirati izgled više web stranica odjednom.

CSS se može dodati HTML-u elementima na 3 načina:

- Inline(u redu) - pomoću atributa stila u HTML elementima
- Interno - pomoću elementa <style> u odjeljku <head>
- Vanjski - pomoću vanjske CSS datoteke

Najčešći način dodavanja CSS-a jest zadržavanje stilova u zasebnim CSS datotekama. Inline CSS se koristi za primjenu jedinstvenog stila u jedan HTML element. Npr.: <h1 style="color:blue;">This is a Blue Heading</h1>

Interni CSS koristi se za definiranje stila za jednu HTML stranicu. Unutarnji CSS definiran je u <head> dijelu HTML stranice unutar elementa <style>.

Vanjski stilski obrazac koristi se za definiranje stila za mnoge HTML stranice. Vanjskom stilskom listom može se promijeniti izgled cjelokupne web sjedišta promjenom jedne datoteke. Svaki dokument u sjedištu mora sadržavati referencu na datoteku vanjskog stila unutar <link> elementa. Element <link> ide unutar <head> odjeljka:

<head>

<link rel="stylesheet" type="text/css" href="mystyle.css">

</head>

CSS svojstvo boja (color) određuje boju teksta koja će se koristiti. CSS *font-family* svojstvo definira font koji će se koristiti. CSS veličina fontova (font-size) određuje veličinu teksta koja će se koristiti.

Stilski list *(Style sheet)* u CSS-u sastoji se od nekoliko pravila. Svako pravilo sastoji se od birača (*selector*) i deklaracijskog bloka.

Birač označava dio *markup-a* na koji se primjenjuje stil. Birač može biti:

- svi elementi istog tipa, npr. svi h2 elementi
- elementi određenog id ili class atributa:
  - id: jedinstven element
  - class: može obuhvaćati više od jednog elementa
- elementi u odnosu na druge elemente u DOM-u.

Deklaracijski blok su vitičaste zagrade unutar kojih se nalaze deklaracije. Svaka deklaracija sastoji se od svojstva, dvotočke (:) i vrijednosti. Između svake dvije uzastopne deklaracije mora se nalaziti točka zarez (;).

Vrijednosti mogu biti ključne riječi poput "center" (sredina) i "inherit" (naslijedi), brojčane vrijednosti poput 100 (debljina fonta), 200px (200 piksela), 50vw (50 % širine viewporta) ili 80% (80 % širine prozora). Vrijednosti boja mogu biti ključne riječi (npr. "red" za crveno), heksadecimalne vrijednosti (npr. #FF0000 ili #F00), RGB vrijednosti od 0 do 255 (npr. rgb(255, 0, 0)), RGBA vrijednosti koje uključuju i alpha prozirnost (npr. rgba(255, 0, 0, 0.8)) ili HSL/HSLA vrijednosti (npr. hsl(000, 100%, 50%), hsla(000, 100%, 50%, 80%)).

CSS također omogućava izradu responzivnih internetskih stranica. Taj način izrade stranice je u novije vrijeme posebno zastupljen budući da uređaji kojima je moguće povezivanje na internet dolaze u različitim veličinama [2].

2.3. JavaScript

JavaScript je skriptni jezik Zapanjujuća većina modernih web stranica koristi JavaScript, i svi moderni web preglednici - na računalima, igraćim konzolama, tabletima i pametnim telefonima - uključuju JavaScript interpretere, čineći JavaScript najviše sveprisutnim programskim skriptnim jezikom u povijesti. JavaScript je dio trojca tehnologija koje svi Web programeri moraju naučiti:

HTML kako bi se odredio sadržaj internetskih stranica, CSS kako bi se odredio izgled internetskih stranica, i JavaScript kako bi se odredilo ponašanje internetskih stranica.

JavaScript je jezik za skriptiranje koji omogućuje izradu interaktivnih web stranica kroz integraciju s HTML-om. Korišten je kao posrednik pri slanju podataka iz obrazaca administratorskog sučelja, te u svrhu interaktivne izmjene poveznica dodijeljenih nekoj liniji.

U HTML-u JavaScript kôd mora biti umetnut između oznaka <script> i </ script>. HTML dokument može sadržavati proizvoljan broj skripti. Skripte se mogu postaviti u <body> ili u <head> dijelu HTML stranice ili u oba.

JavaScript može promijeniti HTML sadržaj, može promjeniti HTML atribute, HTML stil (CSS), može sakriti HTML elemente, te prikazati HTML elemente [3].

2.4. PHP

PHP je skriptni jezik na strani poslužitelja namijenjen prvenstveno za razvoj web stranica, ali se također koristi kao programski jezik opće namjene. Izvorno ga je stvorio Rasmus Lerdorf 1994. godine, no PHP-u referentnu implementaciju sada proizvodi PHP-ov razvojni tim. Skraćenica za PHP je Hypertext Preprocessor. PHP je posebno pogodan za web razvoj.

PHP kod može biti ugrađen u HTML ili HTML5 ili se može koristiti u kombinaciji s raznim sustavima za upravljanje web sadržajem i web okvirom.

PHP skripte izvršavaju se na poslužitelju, te je zbog svoje jednostavnosti pogodan za početnike kao njihov prvi jezik koji se izvršava na strani poslužitelja. PHP kod se izvršava na poslužitelju, a rezultat se može vratiti u preglednik kao običan HTML. PHP datoteke mogu sadržavati tekst, HTML, CSS, JavaScript i PHP kod, te imaju proširenje .php [4].

Dva su načina na koje klijentski preglednik može slati informacije web poslužitelju:

- GET metoda
- POST metoda

GET metoda šalje korisničku informaciju priloženu zahtjevu stranice. Stranica i podatci odvajaju se ? oznakom. GET metoda je ograničena samo na slanje do 1024 znakova. GET se ne može koristiti za slanje binarnih podataka, kao što su slike ili tekstni dokumenti, na poslužitelj. Podatcima poslanim metodom GET može se pristupiti pomoću QUERY_STRING varijable. PHP daje $_GET asocijativni niz za pristup svim poslanim podacima pomoću GET metode.

Metoda POST prenosi informacije putem HTTP zaglavlja. Informacije su kodirane kako je opisano u slučaju GET metode i stavljene u zaglavlje pod nazivom QUERY_STRING.

POST metoda nema ograničenja na veličinu podataka koja se šalje. POST metoda može se koristiti za slanje ASCII kao i binarnih podataka. Podatci poslani POST metodom prolaze kroz HTTP zaglavlje tako da sigurnost ovisi o HTTP protokolu. Korištenjem Secure HTTP može se osigurati da vaši podaci budu sigurni. PHP pruža $ _POST asocijativno polje za pristup svim poslanim podacima pomoću POST metode.

PHP pruža razne mogućnosti koje su korisne pri razvoju jedne web stranice. PHP može generirati dinamički sadržaj stranice, može stvoriti, otvoriti, čitati, pisati, brisati i zatvoriti datoteke na poslužitelju. Također, može prikupljati podatke iz datoteka, može dodavati, brisati i mijenjati podatke u bazi podataka. Ovaj programski jezik se može koristiti za kontrolu pristupa korisnika, te može šifrirati podatke.

S PHP-om niste ograničeni na HTML izlaz. Izlaz mogu biti slike, PDF datoteke, pa čak i Flash filmovi.

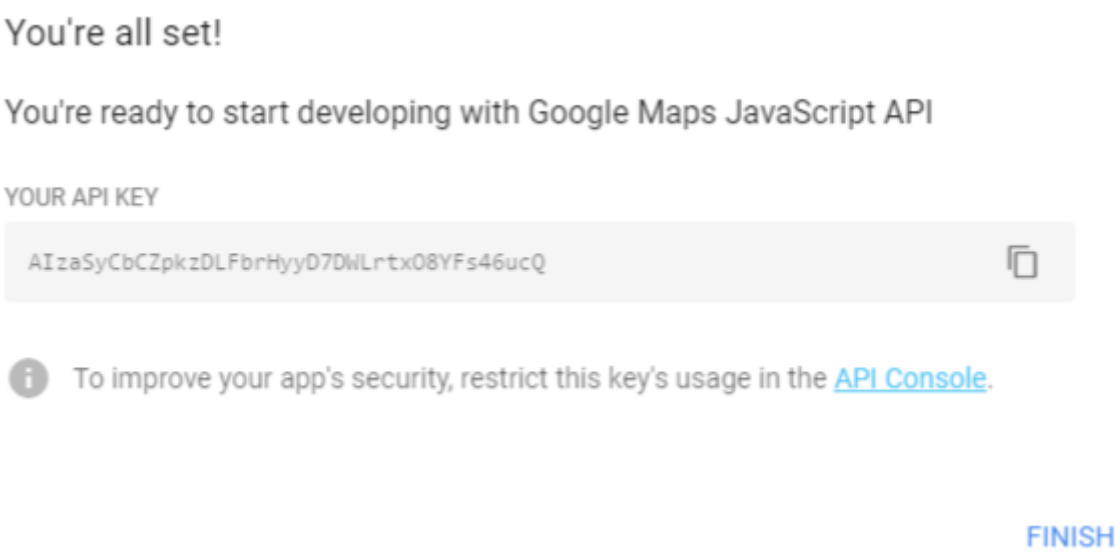
Između ostalog, PHP je dobro koristiti jer se izvodi na različitim platformama (Windows, Linux, Unix, Mac OS X itd.), on je kompatibilan s gotovo svim poslužiteljima koji se danas koriste (Apache, IIS, itd.), te podržava širok raspon baza podataka [5].

2.5. Google Maps API

Google Maps je tehnologija besplatnih digitalnih mrežnih karata, koje čine osnovu mnogih servisa i usluga, od pregledavanja satelitskih snimaka, planiranja rute putovanja (plana kretanja), lociranje traženih mjesta, itd. Dopušta jednostavnu implementaciju na različite web stranice, kombiniranje s drugim aplikacijama, razvoj dodataka i prilagođavanje specifičnim potrebama. Zasnovana na istoj tehnologiji postoji i kao zasebna aplikacija namijenjena instaliranju i korištenju na pojedinim osobnim računalima sa vezom na internet, Google Earth.

API (**A**pplication **P**rogramming **I**nterface) je skup određenih pravila i specifikacija koje programeri slijede tako da se mogu služiti uslugama ili resursima operacijskog sustava ili nekog drugog složenog programa kao standardne biblioteke rutina (funkcija, procedura, metoda), struktura podataka, objekata i protokola. Prilikom izrade ovog rada korišten je Google Maps JavaScript API [6].

Prije izrade aplikacije koja će koristiti API te kako bi bilo moguće koristiti Google API potrebno je zatražiti ključ s kojim će se dobiti dopuštenje korištenja svih mogućnosti koje API pruža. Ključ se dobiva nakon što se korisnik koji ga želi prijavi sa svojim Google računom, zatraži ključ (tu će biti potrebno navesti naziv projekta na kojem će se koristiti) i nakon toga kada se zatraženi ključ dobije korisnik ga unosi na za to predviđeno mjesto u izvornom kodu aplikacije. Dobiveni ključ vidljiv je na sljedećoj slici (Sl.2.5.1.):



Sl. 2.1. *Ključ za korištenje podataka od Google karti*

3. PRVA REALIZACIJA APLIKACIJE

Aplikacija je izvedena na dva načina. Prvi način, koji će biti objašnjen u ovom poglavlju je prikaz korisniku zemljopisnih koordinata i nadmorske visine prilikom klika bilo gdje na mapi svijeta.

3.1. Prikaz mape i rezultata korisniku

Korišten je Bootstrap, najpopularniji okvir za HTML, CSS i JavaScript za razvoj responzivnih web stranica i web aplikacija. Sadrži HTML i CSS temeljene dizajnirane predloške za tipografiju, oblike, gumbe, navigaciju i ostale komponente sučelja, kao i dodatna JavaScript proširenja.

Kreiran je dokument *pointer.php* u kojem je omogućen prikaz mape i rezultata korisniku. Prije svega bilo je potrebno definirati i neke detalje. Postavljen je *utf-8* standard za znakove, time je omogućeno da aplikacija kodira bio koji znak koji se unese, tj. da prepozna sve znakove uključujući znakove hrvatskog jezika kao što su č,ć,š,ž,đ. Također, omogućena je kompatibilnost sa internet explorerom (IE). Te, osigurano je da se aplikacija prilagodi uređaju tj. veličini uređaja koji joj pristupa (laptop, mobitelni uređaj, tablet). Ove tri navedene stavke tj. ove tri navedene oznake moraju doći prve u zaglavlju (head) , bilo koji drugi sadržaj u zaglavlju mora ići nakon njih. Dalje u kodu je definiran izgled aplikacije, između ostalog postavljen je naslov ove aplikacije, zbog jednostavnosti izabran je naslov „App Maps“, te je postavljena visina mape na 500px.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8"><!--Standard za znakove-->
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>App Maps</title>

    <style>
      #map {
        height: 500px;
      }
    </style>
  </head>
  <body>

    <div id="map"></div>
```

Sl. 3.1. *Omogućavanje prikaza mape i rezultata korisniku*

Potom se pomoću elemnta „*script*“ definirala klijentova stranu skripte. Bilo je potrebno povezanti ovu datoteku *pointer.php* sa JavaScript datotekom *kordinate.js* u kojoj se obavlja dohvaćanje

podataka sa *Google Maps-a* i ispis tih podataka. Također ispisan je red koda koji sadrži ključ za provjeru identifikacije kod *Google Maps-a* koji je dobiven prijavom na *Google Maps API*, taj ključ Google provjerava prilikom zahtjeva za pristup njihovom serveru i ukoliko je ključ ispravan zahtjev će biti odobren te ćemo dobiti podatke koje tražimo.

```
    <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
    <script src="js/jquery.min.js"></script>


    <!-- js od aplikacije tj tvoj js di dohvacas podatke i ispisujes korisniku-->
  <script type="text/javascript" src="js/kordinate.js"></script>

  <!-- js koji sadrži tvoj key koji ce google provjeriti prilikom zahtjeva na njihov server-->
          <script async defer src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCbCZpkzDLFbrHyyD7DWLrtxO8YFs46ucQ&callback=initMap"
  type="text/javascript"></script>


  </body>
</html>
```

Sl. 3.2. *Povezivanje datoteke sa Javascript datotekom i omogućavanje dohvata podataka od Google karti pomoću dobivenog ključa*

3.2. Dohvaćanje podataka sa Google mape i ispis tih podataka

Kako bi se omogućilo dohvaćanje podataka kreiran je JavaScript dokument pod nazivom *kordinate.js*. U ovom dokumentu prvenstveno je kreirana varijabla *map* preko čijeg id-a se dohvaćaju elementi. Također, konfiguriran je prikaz mape pri čemu je postavljeno da centar bude u Osijeku na zemljopisnim koordinatama Fakulteta elektrotehnike,računarstva i informacijskih tehnologija. Ovaj dio aplikacije funkcionira na način da se s klikom miša bilo gdje na mapi korisniku prikažu rezultati u informativnom okviru, što je kao i gore navedene stavke definirano u sljedećim linijama koda:

```
function initMap() {

  //kreiramo varijablu map i konfiguriramo prikaz mape
  //dohvacamo ga prema id map
  var map = new google.maps.Map(document.getElementById('map'), {
    zoom: 11,
    center: {lat: 45.556, lng: 18.695},  // FERIT kordinate: 45.55670228086372, 18.695896044373512
    mapTypeId: 'terrain'
  });

  //dohvacanje podataka sa google maps i kreiranje objekta,te spremanje u varijablu njenu vrijednost
  var elevator = new google.maps.ElevationService;
  var infowindow = new google.maps.InfoWindow({map: map});

  //na klik miša na mapi se kreira pop-up element ili infowindow
  map.addListener('click', function(event) {
 displayLocation (event.latLng, elevator, infowindow);
  });
}
```

Sl. 3.3. *Konfiguriranje prikaza mape i omogućavanje da se prilikom klika miša prikaže informativni okvir*

Kada su definirane osnovne stvari za dohvat podataka, potrebno je omogućiti ispis tih istih podataka. Kreirana je funkcija koja ispisuje podatke u već spomenutom informativnom okviru (*infowindow*). Toj funkciji se predaju podatci, te se provjerava da li su ti podatci uredu tj. da li postoje traženi podatci za traženu lokaciju. Podatci koje predajemo toj funkciji su upravo ključni podatci koji se trebaju prikazati; lokacija i visina, te informativni okvir u kojem se prikazuju prethodna dva podatka. Ukoliko postoje podatci za traženu lokaciju, provjerava se da li je nadmorska visina veća od 0. Ako je nadmorska visina veća od nule ispisuje se kao „visina“, a ukoliko je manja od 0 ispisuje se kao „dubina“. U protivnom, ispisuje se da nema rezultata ili da se dogodila pogreška (error).

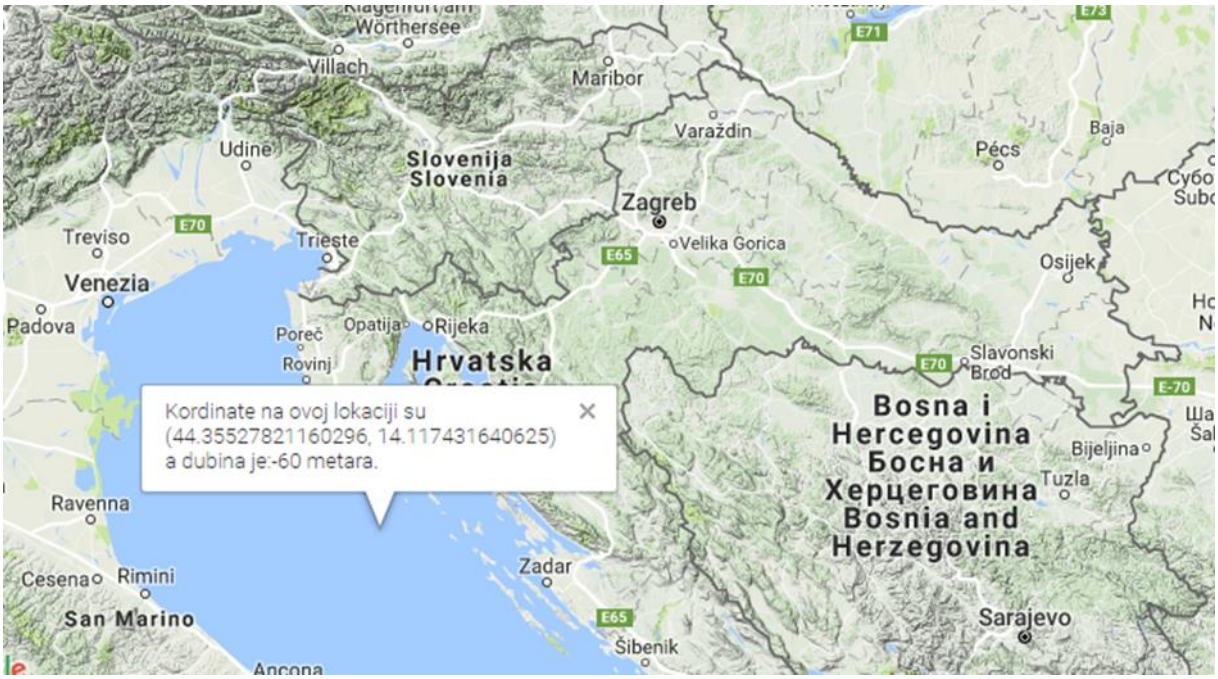
```
function displayLocationElevation(location, elevator, infowindow) {
  // Initiate the location request
  elevator.getElevationForLocations({
    'locations': [location]
  }, function(results, status) {
    infowindow.setPosition(location);
    if (status === google.maps.ElevationStatus.OK) {
      // Retrieve the first result
      if (results[0]) {
        if(results[0].elevation >= 0){
          // Open the infowindow indicating the elevation at the clicked position.
          infowindow.setContent('Kordinate na ovoj lokaciji su <br>' +
              location  + '<br>a visina je:' + results[0].elevation + ' metara.');
        }else{
          infowindow.setContent('Kordinate na ovoj lokaciji su <br>' +
              location  + '<br>a dubina je:' + results[0].elevation + ' metara.');
        }

      } else {
        infowindow.setContent('Nema rezultata');
      }
    } else {
      infowindow.setContent('Error: ' + status);
    }
  });
}
```

Sl. 3.4. *Ispis podataka*

3.3. Prikaz funkcioniranja prve realizacije aplikacije

Prva realizacija aplikacije funkcionira na vrlo jednostavan način; kada korisnik klikne mišem bilo gdje na mapi, ispisuju mu se zemljopisne koordinate i nadmorska visina te lokacije. Na sljedećim slikama može se vidjeti kada korisnik klikne na tlo ispiše se „koordinate i visina“, a ako klikne na morsku površinu ispiše se „koordinate i dubina“.

Sl.3.5. *Funkcioniranje aplikacije ukoliko kliknemo na zemlju*



Sl .3.6. *Funkcioniranje aplikacije ukoliko kliknemo na vodenu površinu*

4. DRUGA REALIZACIJA APLIKACIJE

Kao što je već pomenuto, drugi dio aplikacije je realiziran na način da korisnik ima mogućnost upisivanja u tražilicu (Search Box) adrese, te ga aplikacija odvede na željenu lokaciju i ispiše mu adresu koju je unio i njene koordinate, te može stisnuti tipku „Pošalji“ ukoliko te podatke želi poslati tj. spremiti u bazu podataka.

4.1. Prikaz tražilice i ispis rezultata

Kreiran je dokument *trazilica.php* za realizaciju ovog dijela aplikacije. Prvo je potrebno, kao i u prvoj realizaciji aplikacije, omogućiti prikaz mape korisniku. U kodu je *utf-8* standardom osigurano da aplikacija prepoznaje znakove hrvatskog jezika, također postavljena je visina i margine mape koja će se prikazati. Definirana su svojstva informativnog okvira u kojemu će se ispisati rezultati), to su između ostalih vrsta i veličina fonta. Omogućeno je dodavanje okvira za upis željene adrese ili tražilice (u kodu je nazvan *Search Box*). U taj okvir korisnik može unijeti bilo koju lokaciju na svijetu koju želi, te dobiti listu ponuđenih lokacija koje imaju isti ili sličan naziv između kojih korisnik odabire željenu adresu. Te osim svojstava informativnog okvira, definirana su i svojstva tražilice (*Search Box*).

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>Tražilica</title>
    <style>
          #map {
        height: 100%;
      }
      /* Optional: Makes the sample page fill the window. */
      html, body {
        height: 100%;
        margin: 0;
        padding: 0;
      }
      #description {
        font-family: Roboto;
        font-size: 15px;
        font-weight: 300;
      }

      #infowindow-content .title {
        font-weight: bold;
      }

      #infowindow-content {
        display: none;
      }

      #map #infowindow-content {
        display: inline;
      }
```

Sl. 4.1. *Definiranje svojstava mape, informativnog okvira i tražilice*

```
.pac-card {
  margin: 10px 10px 0 0;
  border-radius: 2px 0 0 2px;
  box-sizing: border-box;
  -moz-box-sizing: border-box;
  outline: none;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.3);
  background-color: #fff;
  font-family: Roboto;
}

#pac-container {
  padding-bottom: 12px;
  margin-right: 12px;
}

.pac-controls {
  display: inline-block;
  padding: 5px 11px;
}

.pac-controls label {
  font-family: Roboto;
  font-size: 13px;
  font-weight: 300;
}

#pac-input {
  background-color: #fff;
  font-family: Roboto;
  font-size: 15px;
  font-weight: 300;
  margin-left: 12px;
  padding: 0 11px 0 13px;
  text-overflow: ellipsis;
  width: 400px;
}
```

Sl. 4.2. *Definiranje svojstava mape, informativnog okvira i tražilice*

```
    #pac-input:focus {
      border-color: #4d90fe;
    }

    #title {
      color: #fff;
      background-color: #4d90fe;
      font-size: 25px;
      font-weight: 500;
      padding: 6px 12px;
    }
    #target {
      width: 345px;
    }
  </style>
</head>
<body>
  <input id="pac-input" class="controls" type="text" placeholder="Search Box">
  <div id="map"></div>
```

Sl. 4.3. *Definiranje svojstava mape, informativnog okvira i tražilice*

Nakom što se definiraju svojstva mape i njenih dijelova, kreirana je varijabla *map* preko čijeg id-a se dohvaćaju elementi. Također, kao i pri prvoj realizaciji aplikacije, konfiguriran je prikaz mape pri čemu je postavljeno da centar bude u Osijeku na zemljopisnim koordinatama Fakulteta elektrotehnike,računarstva i informacijskih tehnologija. Potom nakon varijable *map*, kreirana je varijabla *input* preko koje pomoću ugrađene funkcije dohvaćamo vrijednost koju je korisnik unio u tražilicu. Kreirana je varijabla *searchBox* i postavljeno je da se nalazi u gornjem lijevom kutu.

```
<script>

  function initAutocomplete() {
    var map = new google.maps.Map(document.getElementById('map'), {
      center: {lat: 45.556, lng: 18.695},
      zoom: 13,
      mapTypeId: 'roadmap'
    });

    // Create the search box and link it to the UI element.
    // kreiramo varijablu input i sa ugrađenom funkcijom dohvaćamo iz elementa
    //po id-u "pac-input" vrjednost koju je korisnik unio u trazilicu
    var input = document.getElementById('pac-input');
    var searchBox = new google.maps.places.SearchBox(input);
    map.controls[google.maps.ControlPosition.TOP_LEFT].push(input);

    // Bias the SearchBox results towards current map's viewport.
    map.addListener('bounds_changed', function() {
      searchBox.setBounds(map.getBounds());
    });
```

Sl. 4.4. *Konfiguracija prikaza mape i kreiranje tražilice (varijable searchBox)*

Varijabla *marker* je varijabla u koju se spremi korisnikov upit kada ga unese u tražilicu tj. za vrijeme kada se dohvaćaju podatci o željenoj adresi. Prije svakog novog upita brišu se svi prethodni podatci iz te varijable.

```
var markers = [];
// Listen for the event fired when the user selects a prediction and retrieve
// more details for that place.
searchBox.addListener('places_changed', function() {
  var places = searchBox.getPlaces();

  if (places.length == 0) {
    return;
  }

  // prije slanja upita brisemo sve podatke iz varijabli
  markers.forEach(function(marker) {
    marker.setMap(null);
  });
  markers = [];
```

Sl. 4.5. *Varijabla marker iz koje se brišu podatci prije svakog novog upita*

Za svaku adresu koju korisnik unese osigurano je da ima ikonu koja pokazuje gdje se nalazi ta adresa na mapi, dohvate se ime i zemljopisne koordinate (zemljopisna širina i dužina).

```
// For each place, get the icon, name and location.
var bounds = new google.maps.LatLngBounds();
places.forEach(function(place) {
  if (!place.geometry) {
    console.log("Returned place contains no geometry");
    return;
  }
  var icon = {
    url: place.icon,
    size: new google.maps.Size(71, 71),
    origin: new google.maps.Point(0, 0),
    anchor: new google.maps.Point(17, 34),
    scaledSize: new google.maps.Size(25, 25)
  };

  // Create a marker for each place.
  markers.push(new google.maps.Marker({
    map: map,
    icon: icon,
    title: place.name,
    position: place.geometry.location
  }));

  var infowindow = new google.maps.InfoWindow({map: map});
  infowindow.setPosition(place.geometry.location);
  var sirina = place.geometry.location.lat();
  var duzina = place.geometry.location.lng();
```

Sl. 4.6. *Dohvaćanje podataka za svaki upit*

Nakon što je obavljen dohvat navedenih podataka i osiguran prikaz informativnog okvira potrebno je ispisati podatke u njega. Dakle, ispisuju se koordinate, adresa lokacije koja je unešena te tip lokacije (cesta, fakultet, zdravstvena ustanova itd). Također, u informativnom okviru će biti prikazana tipka „Pošalji“ pomoću koje korisnik može poslati tj. spremiti dobivene rezultate u bazu podataka. To je osigurano pomoću forme u kojoj se nalaze četiri skrivena polja sa varijablama koja se šalju preko POST metode na dokument *spremi.php*.

```
infowindow.setContent(
  'Kordinate su: ' + sirina + ', ' + duzina +
  '<br>Lokacija: ' + place.formatted_address +
  '<br>types: ' + place.types +
  '<br> <form action="spremi.php" method="post">' +
  '<input type="hidden" name="sirina" value="'+sirina+'">' +
  '<input type="hidden" name="duzina" value="'+duzina+'">' +
  '<input type="hidden" name="lokacija" value="'+place.formatted_address+'">' +
  '<input type="hidden" name="tip_lokacije" value="'+place.types+'">' +
  '<input type="submit"> </form>'
);
```

Sl. 4.7. *Slanje podataka putem POST metode*

Kao i pri prvoj realizaciji aplikacije, i ovdje je bilo potrebno definirati kako će mapa biti prikazana korisniku u ovisnosti na kakvom je uređaju tj. u ovisnosti o veličini ekrana korisnikovog uređaja. I na kraju, ključno je bilo ispisati red koda koji koji sadrži ključ za provjeru identifikacije kod *Google Maps-a* koji je dobiven prijavom na *Google Maps API*, taj ključ Google provjerava prilikom zahtjeva da se pristupi njihovom serveru i ukoliko je ključ ispravan zahtjev će biti odobren te će se dobiti traženi podatci.

```
            if (place.geometry.viewport) {
              // Only geocodes have viewport.
              bounds.union(place.geometry.viewport);
            } else {
              bounds.extend(place.geometry.location);
            }
          });
          map.fitBounds(bounds);
        });
      }

    </script>
    <script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCbCZpkzDLFbrHyyD7DWLrtxO8YFs46ucQ&libraries=places&callback=initAutocomplete"
        async defer></script>

  </body>
</html>
```

Sl. 4.8. *Prilagodba apilikacije korisnikovom ekranu i omogućavanje dohvata podataka od Google mapsa pomoću dobivenog ključa*

4.2. Kreiranje baze podataka

Na stranici *phpMyAdmin* kreirana je baza podataka pod nazivom zavrsni_rad. Baza podataka je neophodna za spremanje rezultata koji se dobiju na izlazu aplikacije. Nakon toga unutar te baze kreirana je tablica u koju će se spremati željene rezultate, tablica je nazvana „adrese“ te sadrži id, širinu, visinu, lokaciju i tip lokacije. Kojeg su tipa navedeni podatci može se vidjeti na slici:

	#	Name	Type	Collation	Attributes	Null	Default
☐	1	id 🔑	int(11)			No	None
☐	2	sirina	double			No	None
☐	3	duzina	double			No	None
☐	4	lokacija	varchar(255)	utf8_croatian_mysql561_ci		No	None
☐	5	tip_lokacije	varchar(255)	utf8_croatian_mysql561_ci		No	None

Sl. 4.9. *Kreirana tablica*

Kada su baza podataka i tablica za spremanje podataka kreirane, mogu se slati rezultati u bazu i spremati ih.

4.3. Spremanje podataka u bazu

U dokumentu *trazilica.php* je već omogućeno slanje dohvaćenih podataka preko POST metode na dokument *spremi.php*. *Spremi.php* je dokument u kojem se primaju podatci s POST metodom i spremaju se u bazu.

Da bi smo spremili dohvaćene podatke na bazu, prvo je potrebno spojiti se na bazu, kreiraju se varijable za serverovo ime, korisničko ime, lozinku,te ime baze podataka i definiraju se njihovi nazivi. Potom se ostvaruje konekcija predajući funkciji mysqli gore navedene podatke. Neophodno je provjerili da li je konekcija uspješna, ukoliko nije poništavamo ju naredbom „die“.

```php
<?php

    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname = "zavrsni_rad";
    $ispis;

    // Create connection, pravis objekt, predajes funkciji podatke
    $conn = new mysqli($servername, $username, $password, $dbname);

    // Check connection
    if ($conn->connect_error) {
        die("Connection failed: " . $conn->connect_error);
    }
```

Sl. 4.10. *Spajanje na bazu*

Tablica u koju će se slati podatci nazvana je adrese a podatci koji će se slati u nju su zemljopisna širina, dužina, lokacija tj. unešena adresa te tip lokacije. Ovi podatci poprimaju vrijednosti podataka koji su primljeni putem POST metode iz doukumenta *trazilica.php*.

Nakon što su podatci poslani u bazu, provjerava se da li je upit (query) ispravan. Ukoliko jest, ispisuje se poruka da je željena radnja uspješno obaviljena. U protivnom ispisuje se da se dogodila pogreška. Tese na kraju u kodu zatvara konekicija prema bazi.

```php
// post metodom  dohvacamo po imenu
$sql = "INSERT INTO adrese (sirina, duzina, lokacija, tip_lokacije)
VALUES ('".$_POST["sirina"]."','".$_POST["duzina"]."','".$_POST["lokacija"]."','".$_POST["tip_lokacije"]."')";

// ako je upit(query) ispravan tj tocan ispisuje se poruka
if ($conn->query($sql) === TRUE) {
    $ispis = "Podaci su uspješno spremljeni!";
} else {
    $ispis = "Error: " . $sql . "<br>" . $conn->error;
}

//zatvori konekciju prema bazi
$conn->close();
```

Sl. 4.11. *Dohvaćanje podataka*

Kako bi se na kraju usavršila aplikacija te uljepšao ispis gore navedenog rađeno je na dizajnu, te je uljepšan ispis podataka omogućen sljedećim linijama koda koje su prikazane na slici 4.12. a izgled ispisa podataka koje se dobije na slici 4.13.

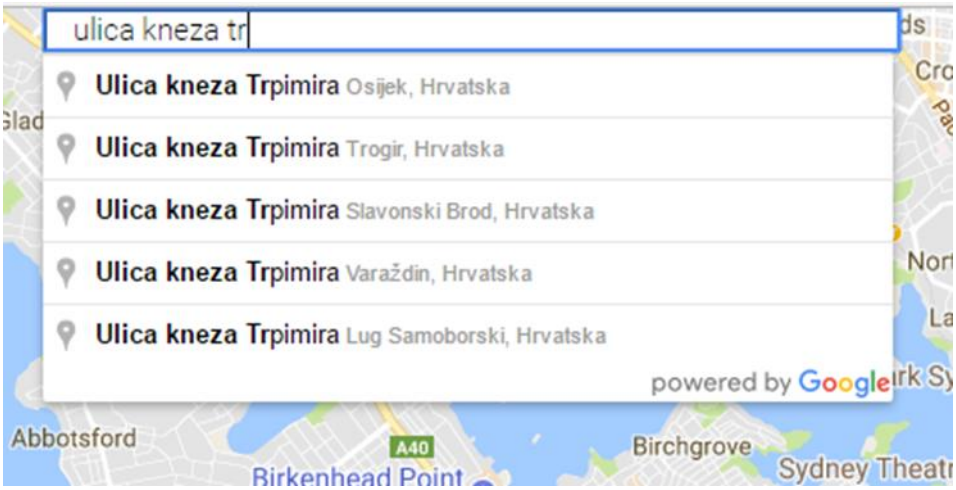
```php
<div class="col-lg-2 col-md-2 col-sm-2 text-center wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".2s">

    <h4>Sirina</h4>
    <?php echo $_POST["sirina"]; ?>

</div>
<div class="col-lg-2 col-md-2 col-sm-2 text-center wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".4s">

    <h4>Duzina </h4>
    <?php echo $_POST["duzina"]; ?>
</div>
<div class="col-lg-2 col-md-2 col-sm-2 text-center  wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".6s">

    <h4>Lokacija </h4>
    <?php echo $_POST["lokacija"]; ?>
</div>

<div class="col-lg-2 col-md-2 col-sm-2 text-center wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".8s">

    <h4>Tip</h4>
    <?php echo $_POST["tip_lokacije"]; ?>
</div>
<div class="col-lg-4 col-md-4 col-sm-4  wow bounceIn animated" data-wow-duration="1s" data-wow-delay="1s">
    <h1><?php echo $ispis; ?></h1>
    <a href="trazilica.php" class="btn btn-style-two btn-lg">NAZAD</a>
</div>
```

Sl. 4.12. *Omogućavanje ispisa dohvaćenih podataka*

Sl. 4.13. *Ispis dohvaćenih podataka*

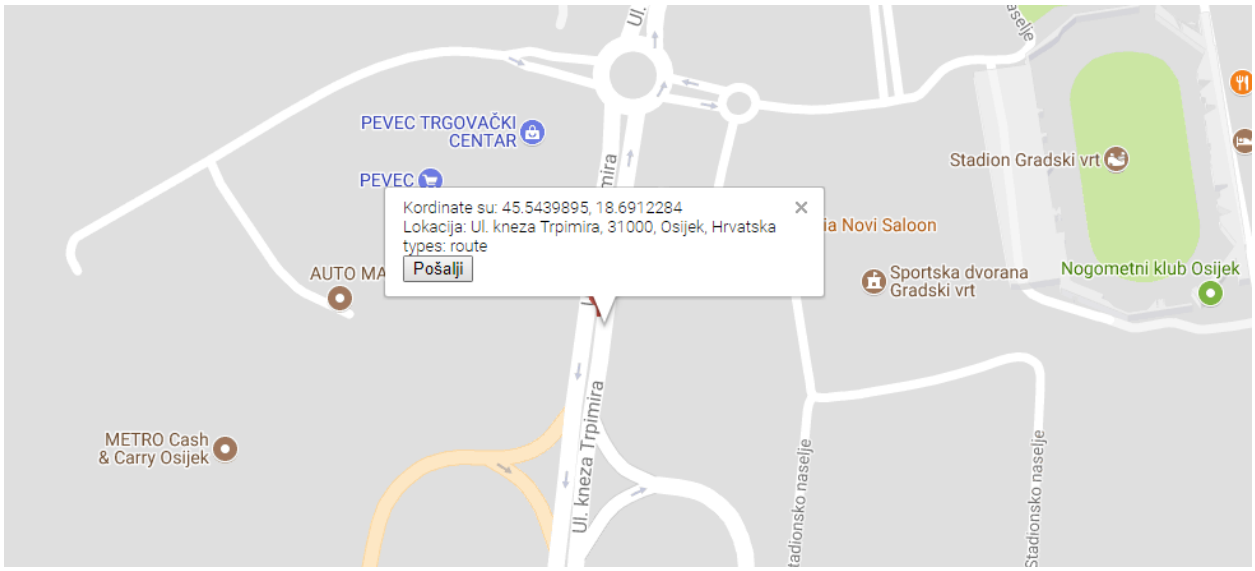
4.4. Prikaz funkcioniranja druge realizacije aplikacije

U tražilicu se unosi adresa, te kada se dobije padajući izbornik s više rezultata odabire se željena lokacija:



Sl. 4.14. *Unos željene lokacije u tražilicu*

Nakon što se izabre lokacija, aplikacija odvodi do te lokacije i ispisuje njene koordinate,naziv i tip lokacije,te ukoliko korisnik želi spremiti tu lokaciju treba pritisnuti tipku „Pošalji“.



Sl. 4.15. *Prikaz rezultata na mapi*

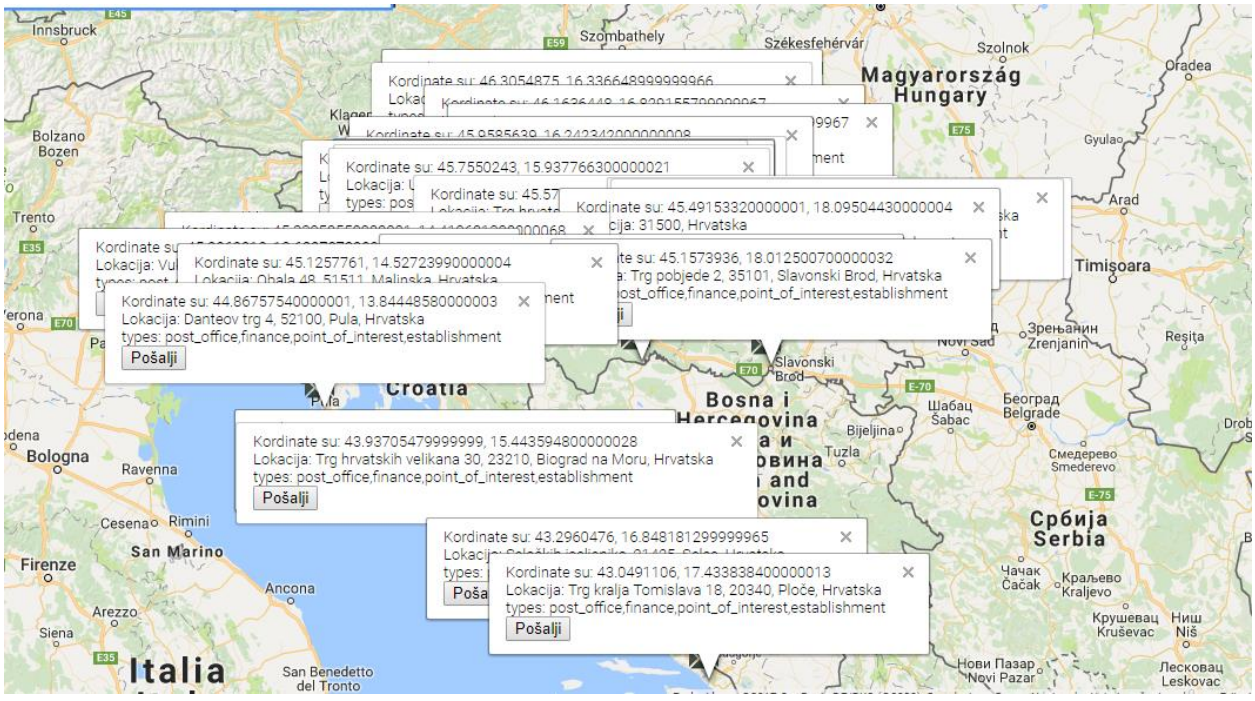
Nakon što korisnik pritisne tipku „Pošalji“ ispiše se ukoliko je uspješno spremljeno.

Sl. 4.16. *Ispis rezultata upita*

Na kraju se može provjeriti u bazi podataka da li se željeni podatci nalaze u tablici „adrese“.

Sl. 4.17. *Dohvaćeni podatci spremljeni u bazu podataka*

Ukoliko se u tražilicu upiše pojam koji se nalazi na više lokacija u svijetu ili državi, na mapi će se dobiti prikaz svih tih lokacija. Npr. ukoliko korisnik upiše pojam Hrvatska pošta koja se nalazi na mnogo mjesta diljem Republike Hrvatske za rezultat će se dobiti lokacije svih pošta u državi:



Sl. 4.18. *Ukoliko se u tražilivu upiše Hrvatska pošta*

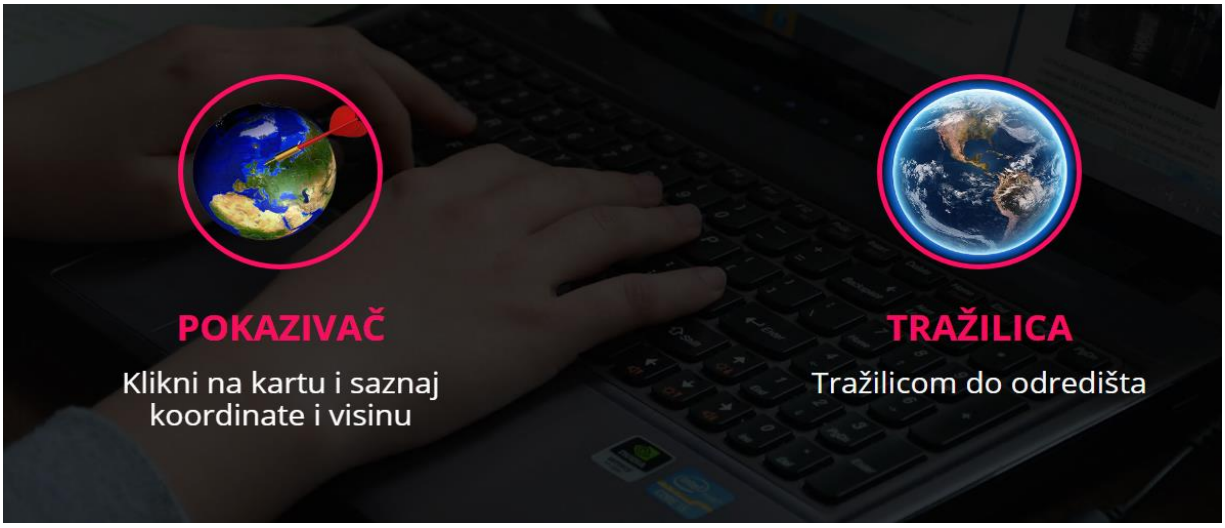
5. DIZAJN

Odabran je odgovarajući gotov predložak za dizajn web stranice tre prilagođen potrebama ove aplikacije. Ulaskom na web stranicu aplikacije omogućenoje korisniku odabir koju realizaciju želi koristiti, da li želi prvu realizaciju koja je nazvana „Pokazivač“ ili drugu koja je nazvana „Tražilica“. Postavljene su dvije ikone sa različitim animiranim slikama i odgovarajućim tekstom tekstom ispod ikone. Sve to je definirano u dokumentu *index.php*.

```
<div id="home-sec">
    <div class="overlay">
        <div class="container">
            <div class="row pad-top-bottom  move-me">
                <a href="pointer.php">
                    <div class="col-lg-5 col-md-5 col-sm-5 text-center">
                        <img src="assets/img/earth.png" class="main-img img-circle wow bounceIn animated" data-wow-duration="1s"
                        data-wow-delay=".2s" alt=""/>
                        <h1 class="wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".4s">POKAZIVAČ</h1>
                        <h2 class="wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".6s">Klikni na kartu i saznaj koordinate i
                        visinu</h2>
                    </div>
                </a>

                <div class="col-lg-2 col-md-2 col-sm-2 text-center  "></div>

                <a href="trazilica.php">
                    <div class="col-lg-5 col-md-5 col-sm-5 text-center">
                        <img src="assets/img/earth.png" class="main-img-2 img-circle wow bounceIn animated" data-wow-duration="1s"
                        data-wow-delay=".2s" alt="" />
                        <h1 class="wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".4s">TRAŽILICA</h1>
                        <h2 class="wow bounceIn animated" data-wow-duration="1s" data-wow-delay=".6s">Tražilicom do odredišta</h2>
                    </div>
                </a>
            </div>
        </div>
    </div>
</div>
```

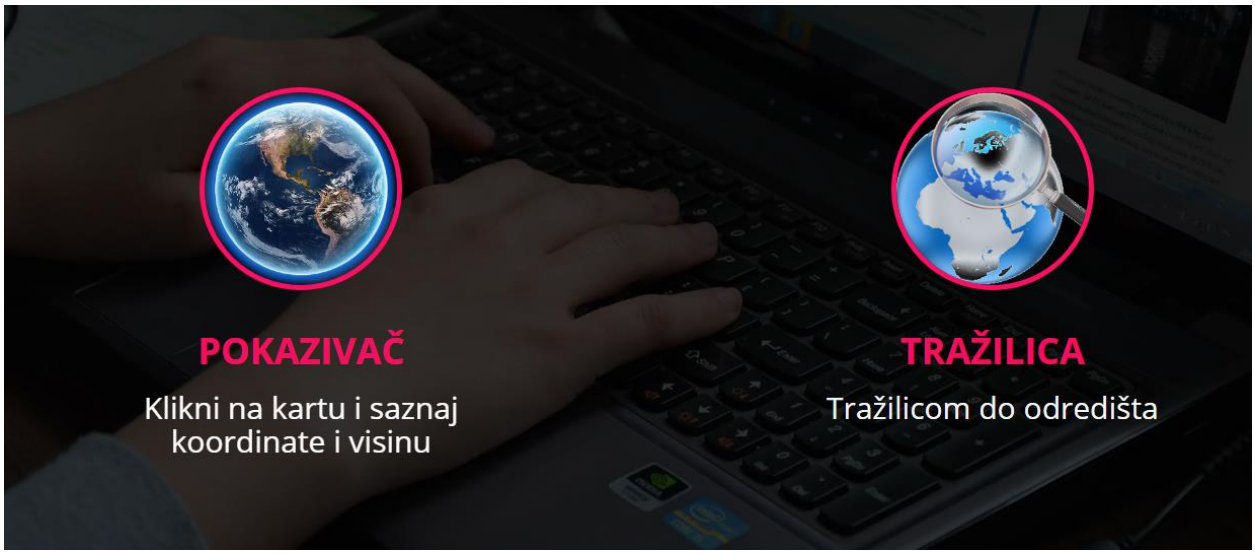
Sl. 5.1. *Omogućenje korisniku izbora realizacije koju želi koristiti*

Na slici 5.2. je prikazana izmjena animirane slike ukoliko korisnik odabere prvu realizaciju tj. Pokazivač.



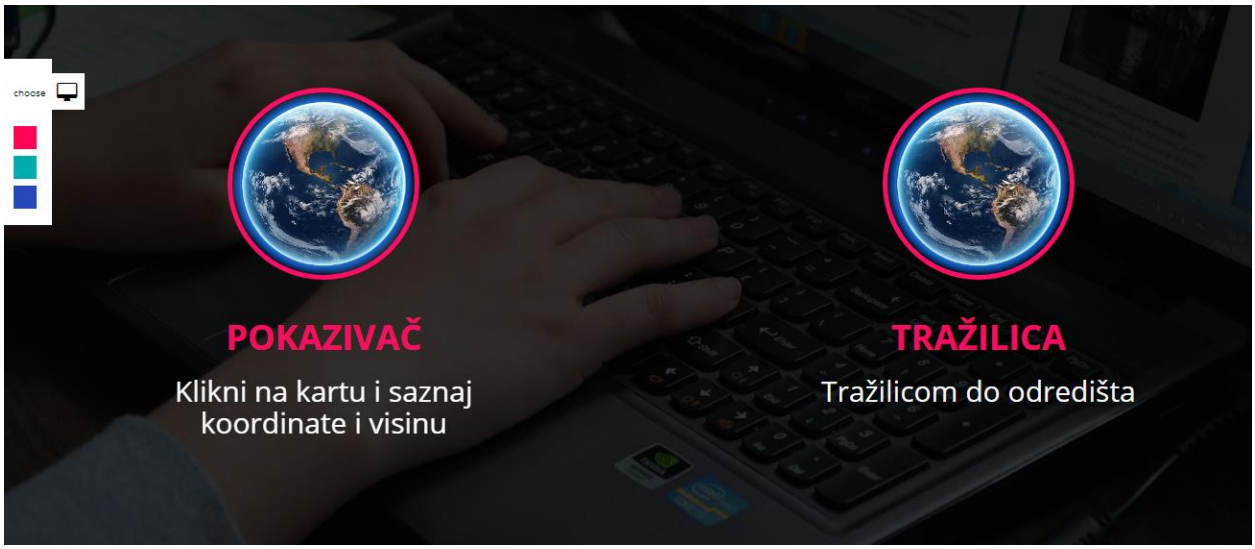
Sl. 5.2. *Izbor prve realizacija-POKAZIVAČ*

Na sljedećoj slici je prikazano što se dogodi ukoliko korisnik izabere drugu realizaciju aplikacije.



Sl. 5.3. *Izbor druge realizacije- TRAŽILICA*

Za boju slova i okvira ikone odabrana je roza, medjutim korisnik može promjeniti boju ukoliko bude želio. Padajući izbornik sa ponuđenim bojama prikazan je u lijevom kutu slike 5.4, a kod kojim je ovo omogućeno na slici 5.5.



Sl. 5.4. *Padajući izbornik s ponuđenim bojama*

```
<body>

    <div class="switcher" style="left:-50px;">
        <a id="switch-panel" class="hide-panel">
            <i class="fa fa-desktop"></i>

        </a>
        <p style="font-size:10px">choose</p>
        <ul class="colors-list">
            <li><a title="Pink" id="pink" class="pink" ></a></li>
            <li><a title="Green" id="green" class="green" ></a></li>
            <li><a title="Blue" id="blue" class="blue" ></a></li>
        </ul>
    </div>
```

Sl. 5.5. *Dio koda koji omogućuje listu boja*

6. ZAKLJUČAK

Uz pomoć neophodnih korištenih tehnologija HTML, CSS, JavaScript i PHP ostvareno je uspješno rješenje zadatka završnog rada. Svaka tehnologija sa svojim pogodnostima i mogućnostima pomogla da se uspješno ostvari dohvat podataka sa *Google Maps-a*, te ispis podataka i na kraju spremanje tih podataka u bazu podataka. Također, korišten je Bootstrap-okvir za HTML, CSS i JavaScript za razvoj responzivnih web stranica i web aplikacija koji je bio koristan za definiranje prikaza mape korisniku. Kako je aplikacija izvedena na dva načina, svaki od njih ima svoje pogodnosti. Prva realizacija aplikacije je izvedena u vidu igrice gdje korisnik može vidjeti koordinate i nadmorsku visinu bilo koje lokacije na svijetu koju odabere klikom miša na mapi. Međutim, nedostatak je što mora željenu lokaciju tražiti po mapi što dovodi do većeg utroška vremena. Dakle, prva realizacija aplikacije je pogodna za ljude koji žele istražiti mapu i zanimaju ga neke pojedinosti država i lokacija koje vidi na mapi. U drugoj realizaciji aplikacije je olakšano korisniku s tim što ima tražilicu u koju unosi željenu adresu i dobiva podatke točno za tu adresu. Ova realizacija je pogodna za osobe koje zanimaja točno određena lokacija ili više lokacija koje pronađe pomoću tražilice i spremi u bazu, te spremljene podatke može vidjeti i koristiti u budućnosti.

Može se zaključiti da je na jednostavan način uz poznate tehnologije kreirana vrlo jednostavna i praktična aplikacija.

LITERATURA

[1] w3schools , https://www.w3schools.com/html/default.asp , (20.5. 2017)

[2] w3schools , https://www.w3schools.com/css/ , (21.5, 2017)

[3] w3schools, https://www.w3schools.com/js/default.asp , (23.5.2017)

[4] Smith, N.: **PHP Essentials**, The History of PHP; An Overwiev of PHP, Playload Media, 2010

[5] tutorialspoint, https://www.tutorialspoint.com/php/php_get_post.htm ,

(27.5. 2017)

[6] Google Developers, https://developers.google.com/maps/faq#whatis ,

(29.5. 2017)

SAŽETAK

Zadatak završnog rada je bio realizirati aplikaciju za dohvat i pohranu zemljopisnih koordinata. Realizacija rada je ostavarena uz pomoć *Google Maps API-a* pri čemu smo dobili ključ pomoću kojeg smo dohvaćali podatke od *Google Maps-a*. Uz pomoć web tehnologija kao što su HTML, CSS, JavaScript i PHP aplikacija je realizirana na dva načina. Prva realizacija aplikacije omogućava korisniku da pri kliku mišem bilo gdje na mapi svijeta dobije ispisane zemljopisne koordinate i nadmorsku visinu te lokacije. Druga realizacija rada je nešto složenija, te ona omogućava korisniku da unese željenu adresu. Nakon unosa adrese u tražilicu aplikacija korisniku prikaže na mapi gdje se nalazi željena lokacija i ispiše mu zemljopisne koordinate. Pri drugoj realizaciji korisnik ima mogućnost dohvaćene podatke spremiti u unaprijed kreiranu bazu podataka. Nakon uspješnog ostvarenja obadvije realizacije aplikacije radili smo na dizajnu kako bi korištenje aplikacije bilo što jednostavnije i zanimljivije.

Ključne riječi: *Google Maps API*, web tehnologije, zemljopisne koordinate

ABSTRACT

APPLICATION FOR RETRIEVAL AND STORAGE OF GEOGRAPHIC COORDINATES

The task of final work was to realize the application for retrieval and storage of geographic coordinates. Work execution is omitted using the Google Maps API, where we have the key to retrieve data from Google Maps. With the help of web technologies such as HTML, CSS, JavaScript and PHP applications is realized in two ways. The first realization of the application allows the user to click with a mouse anywhere on the map of the world to get printed geographic coordinates and altitude of that location. The second implementation of the final work is somewhat more complicated and it allows the user to enter the desired address. After entering the address in the application search engine, the application shows the location where the desired adress is located and displays the geographic coordinates. In the second realization, the user has the ability to save the retrieved data into the database. After successful implementation of both realizations of application, we worked on the design to make application easier and more interesting for use.

Keywords: Google Maps API, web techology, geographic coordinates

ŽIVOTOPIS

Anna-Maria Dragić rođena je 11.06.1995. u Bad-Honnefu u Njemačkoj. U Odžaku u Bosni i Hercegovini je pohađala osnovnu i srednju školu. Srednja škola koju je pohađala je Tehničar za računarstvo, te je tijekom školovanja u 2. i 3. razredu srednje škole obavljala praksu u informatičkoj tvrtci Eurocompany u trajanju od 80 sati. Srednju školu završava 2014., te upisuje Elektrotehnički fakultet u Osijeku, preddiplomski studij smjer Komunikacije i informatika. Trenutno je završna godina preddiplomskog studija Komunikacije i informatika na Fakultetu elektrotehnike, računarstva i informacijskih tehologija u Osijeku. Osim hrvatskog jezika, studentica se koristi i engleskim jezikom.

_____

Anna-Maria Dragić

PRILOZI

CD

- rad u .doc i .pdf formatu
- datoteka s kodom aplikacije