

Web-stranica s dinamičnom web-kartom i bazom podataka logopedskih kabineta

Gunjević, Andrija

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Geodesy / Sveučilište u Zagrebu, Geodetski fakultet**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:256:942049>

Rights / Prava: [Attribution-NonCommercial 4.0 International/Imenovanje-Nekomercijalno 4.0 međunarodna](#)

Download date / Datum preuzimanja: **2025-01-28**



Repository / Repozitorij:

repozitorij.geof.unizg.hr/en





**SVEUČILIŠTE U ZAGREBU
GEODETSKI FAKULTET**

Andrija Gunjević

**WEB-STRANICA S DINAMIČNOM
WEB-KARTOM I BAZOM PODATAKA
LOGOPEDSKIH KABINETA**

Diplomski rad

Zagreb, 2024.

SVEUČILIŠTE U ZAGREBU
GEODETSKI FAKULTET



Na temelju članka 19. Etičkog kodeksa Sveučilišta u Zagrebu i Odluke br. 1_349_11 Fakultetskog vijeća Geodetskog fakulteta Sveučilišta u Zagrebu, od 26.10.2017. godine (klasa: 643-03/16-07/03), uređena je obaveza davanja „Izjave o izvornosti“ diplomskog rada koji se vrednuju na diplomskom studiju geodezije i geoinformatike, a u svrhu potvrđivanja da je rad izvorni rezultat rada studenata te da taj rad ne sadržava druge izvore osim onih koji su u njima navedeni.

IZJAVLJUJEM

Ja, **Andrija Gunjević**, izjavljujem da je moj diplomski rad izvorni rezultat mojeg rada te da se u izradi tog rada nisam koristio drugim izvorima osim onih koji su u njemu navedeni.

U Zagrebu, dana _____

Potpis studenta

I. AUTOR	
Ime i prezime:	Andrija Gunjević
Datum i mjesto rođenja:	
II. DIPLOMSKI RAD	
Naslov:	Web-stranica s dinamičnom web-kartom i bazom podataka logopedskih kabineta
Broj stranica:	56
Broj tablica:	1
Broj slika:	48
Broj bibliografskih podataka:	5+ 23 URL-a
Ustanova i mjesto gdje je rad izrađen:	Sveučilište u Zagrebu – Geodetski fakultet
Mentor:	izv. prof. dr. sc. Ivka Kljajić
Komentor:	
Voditelj:	
III. OCJENA I OBRANA	
Datum zadavanja teme:	29. siječnja 2024.
Datum obrane rada:	21. lipnja 2024.
Sastav povjerenstva pred kojim je branjen diplomski rad:	izv. prof. dr. sc. Ivka Kljajić
	doc. dr. sc. Ana Kuveždić Divjak
	izv. prof. dr. sc. Dubravko Gajski

Zahvala

Hvala mojoj zaručnici Antoniji što mi je uvijek bila oslonac i podrška kroz ovaj period života.

Bez njene potpore ništa od ovoga ne bi bilo moguće.

Web-stranica s dinamičnom web-kartom i bazom podataka logopedskih kabineta

Sažetak: *Web-stranice s geoinformacijama postale su ključni alat u suvremenom društvu, pružajući brojne prednosti i široku primjenu u različitim sektorima. Omogućuju jednostavan i brz pristup geoinformacijama koje su korisne za sve užurbaniji način života. Olakšavaju svakodnevne aktivnosti, od navigacije do pronalaženja specifičnih usluga u blizini. Posebno su korisne u područjima kao što su zdravstvo, obrazovanje i javne usluge, gdje mogu značajno poboljšati učinkovitost i pristupačnost usluga. Cilj ovog diplomskog rada je poboljšanje dostupnosti i kvalitete logopedskih usluga, podržavajući obitelji i stručnjake u pronalaženju i pružanju potrebne skrbi na učinkovit način. U sklopu diplomskog rada izrađen je web-strugač (web scraper) za prikupljanje javno dostupnih podataka, kreirana je baza podataka koristeći web-aplikaciju Supabase i izrađena je web-stranica primjenom razvojnog okvira Angular. Primjenom modernih tehnologija stvorena je dinamična, pouzdana i kontinuirano nadograđiva platforma. Korištenje alata otvorenog koda osigurava da je web-stranica javno dostupna i skalabilna, s mogućnošću evoluiranja kako bi zadovoljila buduće potrebe.*

Ključne riječi: *Angular, geoinformatika, logopedija, web-karta, web-stranica.*

Website with a Dynamic Web Map and Database of Speech-Language Pathology Practices

Abstract: *Websites with geoinformation are a crucial tool in modern society, offering numerous advantages with a wide application across various sectors. They provide easy and quick access to geoinformation useful for an increasingly busy lifestyle. These websites facilitate everyday activities, from navigation to finding specific nearby services. They are particularly useful in areas such as healthcare, education, and public services, where they can significantly improve the efficiency and accessibility of those services. The aim of this thesis is to improve the availability and quality of speech-language therapy services, supporting families and professionals in efficiently finding and providing the necessary care. As part of the thesis, a web scraper was created to collect publicly available data, a database was built using the Supabase web application, and a website was developed using the Angular framework. By utilizing modern technologies, a dynamic, reliable, and continuously upgradable platform was created. The use of open-source tools ensures that the website is publicly accessible and scalable, capable of evolving to meet future needs.*

Keywords: *Angular, geoinformatics, speech-language pathology, web map, website.*

SADRŽAJ

1. UVOD.....	1
2. LOGOPEDIJA.....	2
2.1 Dostupnost i validnost podataka logopedskih kabineta.....	4
3. TEORIJSKA PODLOGA.....	5
3.1 Povijest web-stranica i web-karata.....	5
3.2 Web-kartografija.....	6
3.3 Web-karte.....	7
3.4 Korištene tehnologije i alati.....	8
3.4.1 TypeScript.....	8
3.4.2 HTML.....	9
3.4.3 CSS.....	9
3.4.4 Angular.....	10
3.4.5 Leaflet.....	11
3.4.6 Supabase.....	11
3.4.7 Python.....	12
4. IZRADA WEB-STRANICE I WEB-KARTE.....	13
4.1 Prikupljanje podataka.....	13
4.2 Kreiranje baze podataka.....	20
4.3 Izrada web-stranice i web-karte.....	23
4.3.1 Servis Supabase.....	25
4.3.2 Komponenta za tablicu.....	27
4.3.3 Komponenta za kartu.....	30
5. REZULTATI I RASPRAVA.....	35
5.1 Web-strugač.....	35
5.2 Baza podataka.....	35
5.3 Web-stranica.....	36
6. ZAKLJUČAK.....	51
LITERATURA.....	52
POPIS TABLICA.....	54
POPIS SLIKA.....	54

1. UVOD

U suvremenom svijetu, način na koji se pristupa informacijama i uslugama u potpunosti je drugačiji nego prije. Informacije do kojih je nekada bilo teško doći, danas su dostupne jednim klikom. Međutim, to ne znači da su sve informacije najpraktičnije organizirane i prikazane. Taj problem se javlja kod logopedskih kabineta u Republici Hrvatskoj.

Problematika logopedije u Hrvatskoj je kompleksna i uključuje više problema. Jedan od glavnih problema je nedostatak logopeda, što rezultira dugim listama čekanja i otežanim pristupom uslugama za sve ljude kojima su potrebne. Manjak stručnjaka posebno pogađa ruralna područja, gdje su usluge logopeda često nedostupne (URL 1). Drugi problem je taj što roditelji i skrbnici često nisu dovoljno informirani o važnosti rane intervencije. Nedostatak svijesti i edukacije doprinosi kašnjenju u traženju pomoći, što može negativno utjecati na razvoj djece. Zbog nedostatka lako dostupnih resursa roditelji su prisiljeni čekati i dulje nego što bi trebali.

Cilj ovog diplomskog rada je izraditi web-stranicu s dinamičnom web-kartom koristeći tehnologije otvorenog koda. Primjenom tehnologija otvorenog koda kreirano je rješenje koje je besplatno, dostupno svima i lako nadogradivo. Drugo poglavlje ukratko opisuje struku logopedije, čime se logopedi bave, gdje rade i zašto je odabrana kao predmet izrade web-stranice s web-kartom. Opisana je problematika prikupljanja podataka i nedostatka izvornika. U trećem poglavlju dan je kratki pregled povijesti web-stranica i web-karata. Definirana je web-kartografija te je dana podjela web-karata. Također, objašnjene su tehnologije i alati korišteni u diplomskom radu. U četvrtom poglavlju obrađen je cijeli proces izrade web-stranice, od prikupljanja podataka, preko kreiranja baze podataka pa do izrade samog sučelja. Rezultati i rasprava dani su u petom poglavlju, na temelju kojih je izveden zaključak.

S obzirom na probleme navedene u drugom paragrafu, cilj izrađene web-stranice je olakšati pristup informacijama i uslugama logopedskih kabineta, te na taj način pružiti podršku obiteljima i stručnjacima u pronalaženju i pružanju adekvatne pomoći. Motivacija za izbor teme ovog diplomskog rada je korištenje novih tehnologija i znanja iz područja geoinformatike kako bi se stvorila lako dostupna baza podataka i web-stranica s web-kartom logopedskih kabineta. Izabrano je područje Grada Zagreba iz razloga što će ovakva web-stranica pomoći najvećem broju ljudi ako je locirana na području gdje najviše ljudi stanuje.

2. LOGOPEDIJA

Prema Europskom udruženju logopeda (*European Speech and Language Therapy Association – ESLA*), Logopedija je znanost koja se bavi prevencijom, otkrivanjem, dijagnosticiranjem i tretmanom poremećaja humane komunikacije pod kojom se podrazumijevaju svi oni procesi i funkcije koji su povezani s produkcijom govora, te s percepcijom i produkcijom oralnoga i pisanoga jezika, kao i oblicima neverbalne komunikacije (URL 2).

Prema Međunarodnom udruženju logopedije i fonijatrije (*International Association of Logopedics and Phoniatrics – IALP*), logoped je nezavisan stručnjak čije se središnje aktivnosti ostvaruju na području prevencije, procjene i intervencije u slučajevima poremećaja humane komunikacije, njihovog tretmana te znanstvenog istraživanja. Logoped je osposobljen za rad na prevenciji, dijagnostici i rehabilitaciji poremećaja verbalne i neverbalne komunikacije osoba s teškoćama u razvoju (intelektualne teškoće, cerebralna paraliza, motoričke i kronične bolesti, sljepoća, slabovidnost, autizam) kao i za odabir i razvoj podupirućih i alternativnih komunikacijskih sustava (URL 2).

Prema Američkom udruženju za govor, jezik i sluh (*American Speech-Language-Hearing Association – ASHA*), koristeći niz standardiziranih testova i neformalnih procjena, logoped, uz doprinos ključnih sudionika (npr. pacijent, roditelji, multidisciplinarni tim), utvrđuje prisutnost komunikacijskih poteškoća/poremećaja, vrstu poremećaja (npr. govor, jezik, tečnost itd.), težinu, prognozu i implikacije poremećaja na socijalizaciju i učenje. Na temelju rezultata procjene, logoped dijagnosticira komunikacijski, jezični ili govorni poremećaj te planira intervenciju u skladu s praksom utemeljenom na dokazima (American Speech-Language-Hearing Association, 2016).

Prema Odjelu za zdravstvene i rehabilitacijske znanosti Sveučilišta u Cape Townu (Department of Health and Rehabilitation Sciences, University of Cape Town) (URL 3), logopedi rade s pojedincima, grupama i zajednicama svih dobnih skupina, ne samo na procjeni i sanaciji komunikacijskih poremećaja, već i na promicanju zdravlja te prevenciji teškoća.

U rehabilitacijskom procesu logopedski rad obuhvaća domene komunikacijskih poremećaja (teškoće socijalne komunikacije, poremećaji iz autističnog spektra), jezičnih poremećaja, govornih poremećaja (poremećaji artikulacije, poremećaji tečnosti govora, i dr.), poremećaja

glasa, afazije i drugih neuroloških poremećaja jezika (dizartrija, apraksija), poremećaja pisanog jezika (disleksija, disgrafija), diskalkulije, specifičnih teškoća učenja, oštećenja sluha, poremećaja hranjenja i gutanja te odabira oblika potpomognute komunikacije. Koristeći specijalističke vještine, logopedi rade izravno s klijentima i njihovim njegovateljima kako bi im pružili prilagođenu podršku. Također blisko surađuju s učiteljima, odgajateljima, zdravstvenim radnicima kao što su liječnici, medicinske sestre, psiholozi te drugim srodnim zdravstvenim djelatnicima kako bi razvili individualne programe podrške (Royal College of Speech & Language Therapists, n.d.).

Logopedi rade (URL 2):

- u sustavu zdravstvene zaštite (centrima za rehabilitaciju slušanja i govora, pedijatrijskim, otorinolaringološkim, neurološkim, psihijatrijskim, audiološkim i fonijatrijskim klinikama, ustanovama za mentalno zdravlje i savjetovalištima),
- području odgoja i obrazovanja (kao stručni suradnici u predškolskim ustanovama, osnovnim školama, centrima za odgoj i obrazovanje),
- sustavu socijalne skrbi (dječjim domovima, gerijatrijskim ustanovama),
- posebnim ustanovama (centrima i ustanovama za rehabilitaciju),
- znanstvenoistraživačkim institucijama (istraživačkim centrima, fakultetima),
- nevladinim udrugama i
- privatnoj praksi.

U raznim medijima ističe se kako je u Republici Hrvatskoj manjak logopeda s obzirom na potrebe kao i s obzirom na europski prosjek. Sukladno tome, logopedi su preopterećeni, a liste čekanja za početak terapije i/ili savjetovanje sve duže. Potonje predstavlja velik problem jer je važno na vrijeme pristupiti rješavanju komunikacijskih i govorno-jezičnih poremećaja, osobito kod djece. Zbog navedenog, roditelji se često odlučuju na uključenje u privatni oblik podrške. Zbog lakše organizacije i usklađivanja logopedске terapije s drugim obavezama roditelja i djece (posao, škola, trening...), roditelji su često u potrazi za informacijama o, primarno, postojećim privatnim logopedskim kabinetima, a zatim i o njihovoj lokaciji. Izrada web-stranice i web-karte odabrana je za ovaj diplomski rad kako bi se roditeljima omogućio lakši pristup relevantnim informacijama na jednom mjestu. Također, web-stranica i web-karta bi pomogla u

bržem pronalasku i kontaktiranju logopeda, čime bi se smanjile liste čekanja i ubrzao početak potrebne terapije.

2.1 DOSTUPNOST I VALIDNOST PODATAKA LOGOPEDSKIH KABINETA

Jedan od primarnih izazova pri izradi web-stranice je prikupljanje podataka o postojećim logopedskim kabinetima. Budući da u Republici Hrvatskoj trenutno ne postoji komora logopeda, isto tako ne postoji standardizirani naziv za ustanove koje se bave logopedskim poslovima. Nepostojanje centraliziranog resursa ili stranice gdje su dostupne informacije otežava pronalaženje relevantnih informacija. U interesu prikupljanja što više logopedskih kabineta, planirano je korištenje izvornika *OpenStreetMap* i *Google Maps*.

Iako na *OpenStreetMapu* postoji oznaka za logopediju *healthcare=speech_therapist* (URL 4), ona se na razini cijele baze podataka *OpenStreetMapa* koristi tri puta u cijelom svijetu, što ograničava njezinu korisnost u pružanju pouzdanih podataka.

S druge strane, *Google Maps* je puno ažurniji, ali je puno teže brzo doći do informacija. Web-struganje (eng. *web scraping*) je metoda pomoću koje se mogu brzo i efikasno preuzeti željeni podaci s *Google Mapsa* bez ručnog unošenja istih.

Prema *techopedia.com*, web-struganje je postupak izvlačenja podataka s određene web-stranice. Obuhvaća posjećivanje web-stranice, preuzimanje sadržaja stranice i njegovu analizu kako bi se izdvojili željeni podaci (URL 5). S obzirom na to da se pomoću web-struganja mogu preuzeti sadržaji koji ne bi trebali biti dostupni javno, treba biti oprezan s primjenjivanjem istoga. Postupak web-struganja nije problematičan, ali to ne znači da je svaka uporaba web-struganja legalna (URL 6). U sklopu ovog diplomskog rada prikupljali su se samo javno dostupni podaci na stranici *Google Maps*.

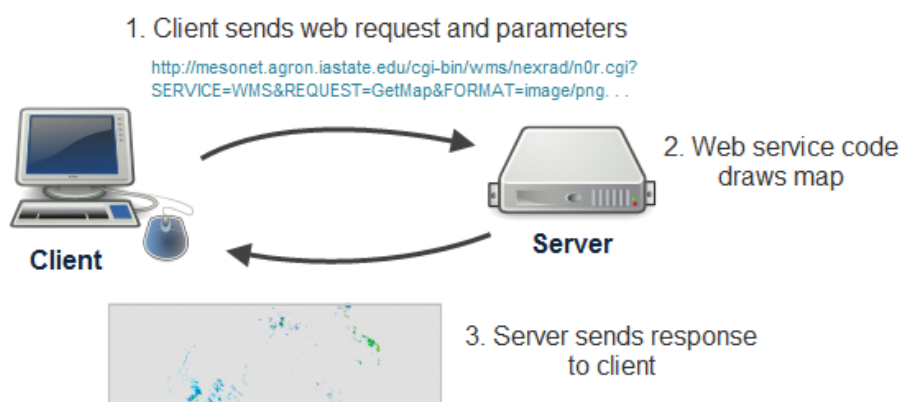
Osim što je teško doći do točnih podataka, teško je provjeriti validnost samih podataka. Budući da ne postoji automatski način provjere podataka, za svaki logopedski kabinet potrebno je pojedinačno provjeravati, nalazi li se na lokaciji gdje piše, je li broj telefona ispravan i je li adresa na kojoj se nalazi web-stranica ispravna.

3. TEORIJSKA PODLOGA

3.1 POVIJEST WEB-STRANICA I WEB-KARATA

Tim Berners-Lee, britanski znanstvenik, izumio je World Wide Web (WWW ili web) 1989. godine dok je radio u CERN-u. Web je izvorno razvijen kako bi zadovoljio zahtjeve za brzom razmjenom informacija između znanstvenika na sveučilištima i institutima diljem svijeta. Ovaj sustav za razmjenu informacija temelji se na protokolu za razmjenu hiperteksta (eng. *HyperText Transfer protocol – HTTP*). Dizajn WWW-a omogućio je jednostavan pristup postojećim informacijama, a prve web-stranice pružale su korisne resurse za znanstvenike iz CERN-a, uključujući CERN-ov telefonski imenik i vodiče za korištenje CERN-ovih centralnih računala (URL 7). S masovnim prihvaćanjem interneta sredinom 1990-ih, ljudi su počeli razmišljati o tome kako dijeliti geoinformacije koristeći web.

U ranom razdoblju weba, geoinformacije su bile dostupne samo na osobnim računalima ili glavnim računalima unutar čelnih organizacija. GIS analitičari pristupali su podacima putem radnih računala koja su bila povezana s poslužiteljima koji su se nalazili unutar ureda. Prve karte na webu su bile u obliku statičnih slika karata. Ubrzo se shvatilo da postoji potencijal za interaktivne karte. Prvi softveri poput *Map Servera* i *Esri ArcIMS-a* omogućili su stvaranje tih interaktivnih karata. Te prve karte su za današnje standarde bile jako pikselizirane i gotovo nekoristive, ali tada su bile revolucionarne. Funkcionirale su na način da korisnik šalje zahtjev za kartom, poslužitelj taj zahtjev obrađuje te vraća korisniku kartu (slika 3.1). Taj način funkcioniranja se i danas koristi u većini web-karata (URL 8).



Slika 3.1. Vizualizacija dohvaćanja karte iz servera (URL 9)

3.2 WEB-KARTOGRAFIJA

Web-kartografija je moderan pristup kartografiji koji olakšava pristup i rukovanje geoinformacijama putem weba. Web-kartografija se odnosi na bilo koju vizualnu prezentaciju prostora dostupnog na webu ili namijenjen webu. Najveća podskupina karata koje nisu obuhvaćene ovom definicijom mogu se pronaći u digitalnim arhivama analognih karata, jer te karte nisu izvorno namijenjene za web. Pojam web-kartografije pokriva karte koje su prilagođene korisniku kao i one koje nisu prilagođene korisniku. Iako je koncept web-kartografije često praćen slikom animiranih i interaktivnih karata, to je samo podskupina svih karata dostupnih u ovom mediju (Župan i Frangeš, 2015).

Web-kartiranje je postupak dizajniranja, implementacije, generiranja i isporuke karata na webu. Dok se web-kartiranje prije svega bavi tehnološkim pitanjima, web-kartografija također proučava teorijske aspekte: korištenje web-karata, primjenjivost web-karata, društvene aspekte i dr. Geoinformacijski sustavi bitni su čimbenici unutar web-kartiranja. Pojmovi web-GIS i web-kartiranje vrlo se često koriste kao sinonimi, iako nemaju isto značenje. Pojava web-kartiranja može se smatrati glavnim trendom u kartografiji. Prije toga, kartografija je bila ograničena na nekoliko tvrtki, instituta i agencija za izradu karata koje su trebale skup hardver, softver, profesionalne kartografe i geoinformatičare. Kod web-kartiranja, tehnologije za izradu karata su besplatno dostupne, a geoinformacije omogućuju da više ljudi nego ikada izrađuju web-karte (Župan i Frangeš, 2015).

Svaka osoba s minimalnim predznanjem i infrastrukturom ima mogućnost postati pružateljem geoinformacija. Koliko su ove činjenice dobre, toliko su i loše. Jednostavan prijenos geoinformacija i demokratizacija kartografije s jedne strane dovodi do puno više izrađenih karata. S druge strane, više karata nije uvijek dobro, pogotovo ako se pri izradi takvih karata ne poštuju kartografska pravila koja se godinama poštuju iz dobrih razloga.

Sve u svemu, korištenje weba za distribuciju karata je veliki korak i napredak za kartografiju. Otvara se svijet s puno novih mogućnosti gdje se karte mogu izraditi brže, kvalitetnije i jeftinije nego ikad.

3.3 WEB-KARTE

Web-karte mogu biti statične i dinamične, a obje vrste mogu biti samo za gledanje ili interaktivne. Većina karata na webu još uvijek su statične i nisu interaktivne. To su npr. skanirane karte postavljene na web. Neke vrlo stare i teško dostupne karte postaju na taj način dostupne mnogima. Statične karte mogu biti i interaktivne. „Klikom“ na pojedino mjesto izazivaju se određene operacije, npr. pridruživanje dodatnih informacija, zumiranje i sl. Web ima nekoliko mogućnosti za prikaz dinamičkih procesa putem animacije. Tipičan je primjer globus koji se okreće ili kretanje oblaka na meteorološkim kartama.

Dostupnost i aktualnost, dvije glavne prednosti web-karata, nisu uvijek potpuno ispunjene. Neke web-stranice nisu redovito održavane pa korisnici gube povjerenje u te stranice. Mnogo je važnije da u praksi postoje i ograničenja u dostupnosti: pronalaženje web-karata, jezik, dostupnost svima, web-karte i geopodaci uz naplatu, dostupnost interneta, brzina prijenosa podataka. Ekonomski činioci glavni su razlozi ograničenja u pristupu na web. Podaci dostupni bez naplate nisu uvijek najkvalitetniji.

Kreiranje karata na web-stranicama postaje nova specijalizacija kartografa i kartografskih tvrtki, čiji proizvodi i usluge moraju postati dostupni i preko weba. Sadržaj karte ovisi u velikoj mjeri o mjerilu. U načelu karte na zaslonu (ekranu) monitora, pa prema tome i web-karte imaju promjenljivo mjerilo, jer se mogu povećavati i smanjivati (zumiranje). Kartografi mogu u kreiranju karata primijeniti tri vrste zumiranja.

- Pri statičkom linearnom zumiranju slika se linearno povećava, ali sadržaj ostaje isti. Karta je spremljena kao slika. Ako je grafika vektorska, slika je jednako oštra, a ako je rasterska postaju vidljivi pikseli.
- U statičkom stupnjevitom zumiranju dostupna je serija karata istog područja, svaka oblikovana za drugo mjerilo. Pri zumiranju softver automatski bira najprikladniju kartu za traženo mjerilo.
- U dinamičkom zumiranju postoji izravna veza između mjerila i sadržaja karte. Što je mjerilo krupnije, prikazuje se više detalja na karti. Potrebna je izravna veza slike i baze podataka. Kartografska generalizacija i simbolizacija najčešće se mijenja s mjerilom. Naselje u sitnome mjerilu prikazuje se kružićem, a u krupnijem konturom naselja (Frančula, 2004).

3.4 KORIŠTENE TEHNOLOGIJE I ALATI

U realizaciji web-stranice korišteno je nekoliko različitih tehnologija i alata. Sučelje web-stranice (eng. *front-end*) izrađeno je u razvojnom okviru (eng. *framework*) Angular, koji koristi programski jezik TypeScript. Uređivanje sadržaja stranice provedeno je primjenom HTML-a i CSS-a. Za izradu web-karte upotrijebljena je biblioteka *Leaflet* za interaktivne web-karte.

Pri prikupljanju podataka korišten je programski jezik Python. Za izradu baze podataka korištena je web-aplikacija Supabase, pomoću koje je izrađena PostgreSQL baza podataka. Prednost korištenja web-aplikacije Supabase je nekorištenje pozadinske platforme kao što je Spring Boot, čime se znatno ubrzava proces razvoja web-stranice.

3.4.1 TypeScript

TypeScript je programski jezik u kojem je napisan razvojni okvir Angular. Razvijen je kao nadogradnja programskog jezika JavaScript.

Prema službenoj stranici TypeScripta, glavna prednost programskog jezika TypeScript je sigurnost pri definiranju tipova podataka, što pruža dodatnu sigurnost prilikom razvoja web-stranica (URL 10). Takvo definiranje varijabli dovodi do ranije detekcije grešaka te održivosti koda. Također, postoje bolji alati koji pomažu u razvoju web-stranica u popularnim aplikacijama poput Visual Studio Code. TypeScript je u potpunosti kompatibilan s prijašnjim i novim verzijama JavaScripta.

Jedan od nedostataka TypeScripta je što je potrebno znati osnove JavaScripta. Budući da web-preglednici nemaju mogućnost interpretiranja TypeScript koda, isti se mora prevoditi (eng. *compile*) u JavaScript kod (URL 10). To dovodi do nešto sporijih performansi TypeScript aplikacija.

Iako je relativno nov jezik, TypeScript predstavlja veliki napredak u svijetu programiranja. U usporedbi s JavaScriptom, TypeScript omogućuje programerima veliki broj praktičnih dodataka koji ubrzavaju i olakšavaju razvoj web-stranica.

3.4.2 HTML

HTML (kratica od eng. *HyperText Markup Language*) je prezentacijski jezik koji definira značenje i strukturu sadržaja na webu. Zbog svoje jednostavnosti rasprostranjen je diljem svijeta kao najpopularniji jezik u domeni razvoja web-stranica. Svaki HTML dokument temelji se na HTML elementima. Svaki HTML element ima različitu uporabu. Svaka HTML oznaka započinje sa znakom manje od (<), a završava sa znakom veće od (>) (URL 11).

Najčešće se koristi zajedno s JavaScriptom i CSS-om. JavaScript se koristi kako bi se odredilo ponašanje HTML elemenata, dok se CSS koristi kako bi se isti uredili.

3.4.3 CSS

CSS (kratica od eng. *Cascading Style Sheets*) je stilski jezik koji služi za uređivanje web-stranice. Korištenjem CSS-a definira se na koji način će se pojedini HTML elementi prikazivati. CSS je jedan od temeljnih jezika otvorenog weba i standardiziran je za sve web-preglednike.

CSS dokument sastoji se od više pravila. Svako pravilo sastoji se od selektora i deklaracijskog bloka. Selektor (eng. *selector*) označava HTML element na koji se primjenjuje stil. Selektor se može odnositi na:

1. Sve HTML elemente istog tipa
2. Elemente određenog *id* ili *class* atributa:
 - a. *id*: jedinstveni identifikator nekog HTML elementa
 - b. *class*: može obuhvaćati više od jednog HTML elementa
3. HTML element u odnosu na druge elemente.

Pseudoklase su klase koje omogućuju opisivanje informacija koje se ne mogu selektirati pomoću HTML elementa. Na primjer, pseudoklasa *:hover* identificira sadržaj samo ako korisnik drži pokazivač nad sadržajem.

Deklaracijski blok su vitičaste zagrade unutar kojih se nalaze pravila po kojima se stilizira HTML element. Svaka deklaracija sastoji se od svojstva, dvotočke i vrijednosti. Između svake dvije uzastopne deklaracije treba se nalaziti točka sa zarezom (URL 12).

3.4.4 Angular

Angular je razvojni okvir izgrađen na programskom jeziku TypeScript. Prva verzija razvojnog okvira Angular, tada AngularJS, razvijena je 2009., a izdana je od strane Googlea 2010. Dok je AngularJS baziran na JavaScriptu, Angular se temelji na TypeScriptu.

Prema službenoj stranici Angulara, neke od prednosti korištenja okvira Angular su:

- a) Temeljen je na komponentama za izgradnju skalabilnih web-aplikacija.
- b) Kolekcija dobro integriranih biblioteka koje pokrivaju širok raspon funkcionalnosti.
- c) Paket alata za programere koji pomažu u razvoju web-aplikacija.

Neke od stavki važne za Angular su:

1. Komponente (eng. *Components*) – temeljni građevni blok za stvaranje aplikacija u Angularu. Osiguravaju dobro organiziranu strukturu web-aplikacije kako bi stranica bila lagano održiva.
2. Direktive (eng. *Directive*) – posebni atributi koji omogućuju rukovanje HTML elementima.
3. Servisi (eng. *Service*) – datoteke koje omogućuju dijeljenje iste logike u cijeloj aplikaciji.

Angular se temelji na modularnosti. Svaka komponenta se sastoji od HTML dokumenta koji definira kako će komponenta izgledati, CSS dokumenta koji stilizira HTML, te TypeScript dokumenta koji definira ponašanje komponente. Korištenje samostalnih (eng. *standalone*) komponenti omogućuje da je svaka komponenta smisljena cjelina s određenom zadaćom. Moguće je deklarirati sve servise, druge prateće komponente i ostalo što je potrebno kako bi komponenta funkcionirala (URL 13).

Za pozive prema bazi podataka korištena je biblioteka RxJS, koja se najčešće koristi u Angular projektima. RxJS (kratica od eng. *Reactive Extensions Library for JavaScript*) je biblioteka za reaktivno programiranje. Koristi jednu temeljnu vrstu podatka, *Observable*, i izvedenice iz njega. Najviše će se koristiti *Subject* koji omogućuje višestruko slanje podataka u različite komponente. *Subject* funkcionira na način da se unutar komponente pretplaćuje na njega te svaki put kada se vrijednost pošalje preko *Subjecta*, izvršava se definirana radnja (URL 14).

3.4.5 Leaflet

Leaflet je vodeća JavaScript biblioteka otvorenog koda za responzivne interaktivne karte. Uz *OpenLayers* i *GoogleMaps API* jedna je od najpopularnijih JavaScript biblioteka za izradu web-karata. Neke od mogućnosti su korištenje podataka iz GeoJSON datoteka, dodavanje stilova, dodavanje markera na kartu i dodavanje skočnih prozora. *Leaflet* podržava korištenje WMS slojeva, GeoJSON slojeva, vektorskih slojeva te Tile slojeva. Koristi se zbog jednostavnosti te velike zajednice korisnika (URL 15).

3.4.6 Supabase

Supabase je web-aplikacija za izgradnju sigurnih i učinkovitih PostgreSQL baza podataka s minimalnom konfiguracijom (URL 16). Aplikacija pruža široku lepezu funkcionalnosti sličnih drugim aplikacijama poput Firebasea, kao što su provjera autentičnosti, funkcije bez poslužitelja i pohrana podataka. Budući da se Supabase temelji na tehnologijama otvorenog koda, fleksibilnija je i nudi kontrolu nad podacima i aplikacijama. Za izradu web-stranice u okviru diplomskog rada korištena je samo PostgreSQL baza podataka s PostGIS dodatkom za geopodatke.

PostgreSQL je objektno-relacijska baza podataka otvorenog koda koja koristi i proširuje SQL u kombinaciji s mnogim značajkama koje sigurno pohranjuju i skaliraju najsloženija radna opterećenja (URL 17). PostgreSQL pruža veću fleksibilnost od drugih baza podataka otvorenog koda te ima izrazito veliku zajednicu korisnika i dobru podršku.

Osim toga, PostgreSQL ima širok izbor dodataka koji nadodaju funkcionalnosti na postojeću bazu podataka. Za potrebe ovog diplomskog rada korišten je dodatak PostGIS za PostgreSQL bazu podataka. PostGIS proširuje mogućnosti PostgreSQL dodavanjem podrške za pohranjivanje i postavljanje upita o geopodacima (URL 18).

3.4.7 Python

Python je jedan od najpopularnijih programskih jezika. Osnovna karakteristika Pythona je jednostavnost jezika koju opisuje mali broj naredbi potrebnih kako bi se napisao dobar kod. Druga karakteristika je dinamičnost jezika. Za razliku od programskog jezika TypeScript ili Java, nije potrebno deklarirati tip varijable prije izvršavanja napisane skripte, nego je tip varijable dinamičan, odnosno može se lako promijeniti. U tom pogledu sličniji je programskom jeziku JavaScript nego programskom jeziku TypeScript. Također, Python ima mogućnost proširenja s velikim brojem biblioteka i modula. Zbog toga se Python koristi za izradu web-aplikacija, obradu podataka, interakciju s bazama podataka, razvoj korisničkih sučelja, vizualizacije i sl. Proširivost i jednostavnost omogućile su Pythonu široki raspon iskoristivosti, pa ga tako koristi velik broj poznatih tvrtki poput Googlea, Youtubea i sličnih. Također, može se koristiti za skriptiranje u raznim GIS programima, poput ArcGIS-a i QGIS-a (Miler, 2021).

Neke od prednosti korištenja programskog jezika Python su (URL 19):

- a) Sintaksa je jednostavnija za početnike, lakše ju je čitati, pisati i otkloniti neispravnosti.
- b) Fleksibilnost u korištenju zbog velikog broja dostupnih biblioteka i razvojnih okvira.
- c) Besplatan je i otvorenog koda. Zbog svoje velike popularnosti, Python ima veliki broj resursa koji mogu pomoći korisniku u učenju.

Za potrebe diplomskog rada, korišteno je nekoliko biblioteka: *playwright*, *dataclasses*, *pandas* i *os*. *Playwright* je biblioteka koja omogućuje pokretanje virtualnog web-preglednika i radnji nad njim (URL 20). *Dataclass* je vrsta klase u Pythonu koja je pogodna za spremanje različitih vrsta podataka (URL 21).

4. IZRADA WEB-STRANICE I WEB-KARTE

U ovom poglavlju objašnjen je postupak izrade web-stranice i web-karte. Cijelo sučelje stranice izrađeno je u razvojnom okviru Angular, no opisan je i postupak prikupljanja podataka i spremanja istih u bazu podataka. Postupak izrade stranice može se podijeliti u tri potpoglavlja:

1. Prikupljanje podataka;
2. Kreiranje baze podataka;
3. Izrada web-stranice i web-karte.

4.1 PRIKUPLJANJE PODATAKA

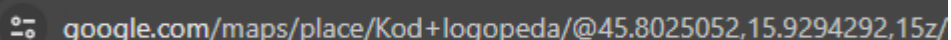
Razvoj skripte za web-struganje započinje uvozom potrebnih biblioteka: *playwright*, *dataclasses*, *pandas* i *os* (slika 4.1).

```
from playwright.sync_api import sync_playwright
from dataclasses import dataclass, asdict, field
import pandas as pd
import os
```

Slika 4.1. Uvoz biblioteka u skripti za web-struganje

Nakon uvoza potrebnih biblioteka, definiraju se dvije klase: *Kabinet* i *KabinetList*. Klasa *Kabinet* služi za spremanje svih dostupnih podataka o pojedinom logopedskom kabinetu. S druge strane, klasa *KabinetList* služi kao lista svih dostupnih kabineta. Također, unutar klase *KabinetList* nalazi se funkcija *save_to_csv* pomoću koje se svi prikupljeni podatci o kabinetima spremaju u datoteku formata *.CSV. Prije pisanja funkcije web-struganja, dodana je još jedna pomoćna funkcija nazvana *extract_coordintes* (slika 4.2).

Budući da se na dijelu web-stranice gdje su ispisani svi podaci o kabinetu ne ispisuju koordinate istoga, potrebno ih je izvesti na drugi način. Klikom na pojedini kabinet unutar *Google Maps*, koordinate se ispisuju unutar URL stranice. Iz tog URL-a se onda izvlače koordinate (slika 4.3).



Slika 4.3. Koordinate unutar URL-a

```

@dataclass
class Kabinet:
    name: str = None
    address: str = None
    website: str = None
    phone_number: str = None
    latitude: float = None
    longitude: float = None

@dataclass
class KabinetList:
    kabinet_list: list[Kabinet] = field(default_factory=list)
    save_at = 'output'

    def dataframe(self):
        return pd.json_normalize(
            (asdict(kabinet) for kabinet in self.kabinet_list), sep="_"
        )

    def save_to_csv(self):
        if not os.path.exists(self.save_at):
            os.makedirs(self.save_at)
        self.dataframe().to_csv(f"output/kabineti.csv")

def extract_coordinates(url: str) -> tuple[float, float]:
    coordSubstring = url.split('!3d')[-1]
    latitude = coordSubstring.split('!4d')[0]
    longitude = coordSubstring.split('!4d')[1].split('!')[0]
    return float(latitude), float(longitude)

```

Slika 4.2. Pomoćne klase i funkcije prije web-struganja

Nakon napisanih pomoćnih klasa i funkcija, potrebno je napisati glavnu funkciju *main* (slika 4.4). Prvo se poziva biblioteka *playwright* s kraticom *p*. Pomoću te kratice koristi se sve što *playwright* omogućuje. Prvo se otvara novi web-preglednik, u ovom slučaju Google Chrome. Nakon toga, otvara se stranica unutar virtualnog preglednika i ide na *Google karte* (*Google Maps*) te traži pojam „logoped zagreb“. Budući da se pri svakom paljenju virtualnog web-preglednika otvara novi preglednik bez ikakvih prethodnih podataka, potrebno je prihvatiti Googleove uvjete poslovanja (slika 4.5). Prihvaćaju se odabiranjem gumba koji sadrži tekst „Prihvati sve“, te pozivanjem funkcije *click*. Tako se dolazi do liste kabineta.

```

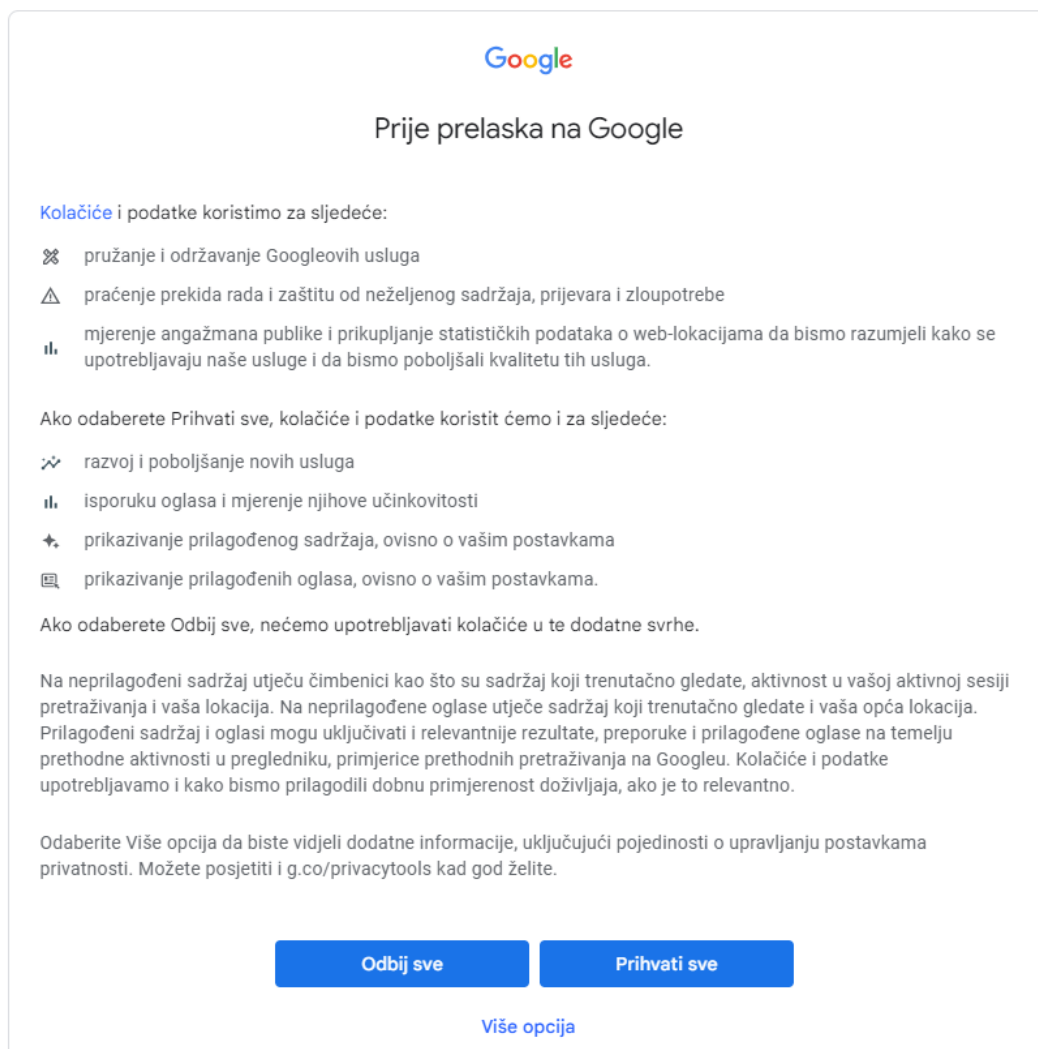
def main():
    with sync_playwright() as p:
        browser = p.chromium.launch()
        page = browser.new_page()

        page.goto("https://www.google.com/maps/search/logoped+zagreb/",
        timeout=60000)

        page.get_by_role("button", name="Prihvati sve").click()
        page.wait_for_timeout(500)

```

Slika 4.4. Funkcija main web-strugača



Slika 4.5. Prihvaćanje uvjeta poslovanja prije ulaska na Google karte (Google Maps)

U sljedećem koraku implementirana je funkcionalnost listanja kroz sve kabinete. To se izvodi na način da se prvo virtualni miš stavi preko prvog kabineta u listi pomoću funkcije *hover*. Nakon toga, započinje *while* petlja unutar koje se listaju kabineti sve dok se ne dođe do dijela liste gdje će se ispisati poruka „Reached the end of the list.“. Kada se to dogodi, petlja se prekida te se može nastaviti dalje (slika 4.6). Na slici 4.7 prikazana je lista kabineta u virtualnom web-pregledniku s virtualnim mišem.

```

page.hover('(/a[@class="hfpxyz"])[1]')

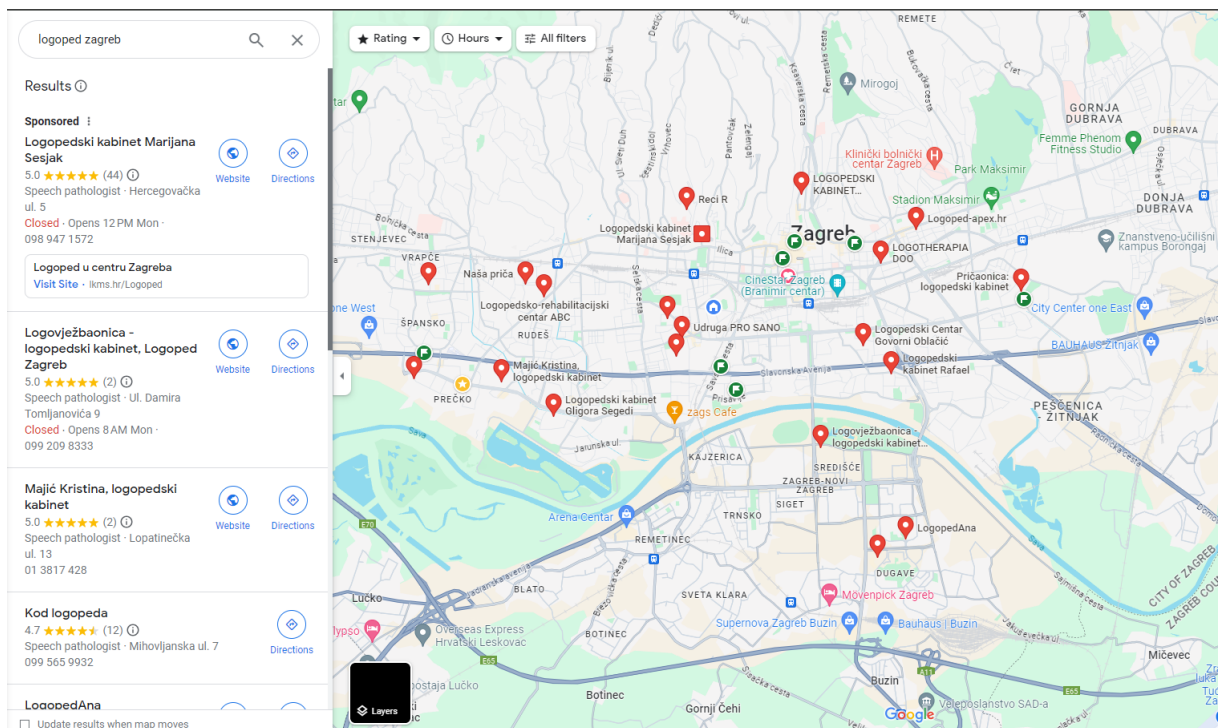
while True:
    page.mouse.wheel(0, 10000)
    page.wait_for_timeout(3000)

end_of_list = page.query_selector('//div[contains(@class,
"tLjsW")])

if end_of_list:
    print("Reached the end of the list.")
    break
else:
    print("Scrolling...")

```

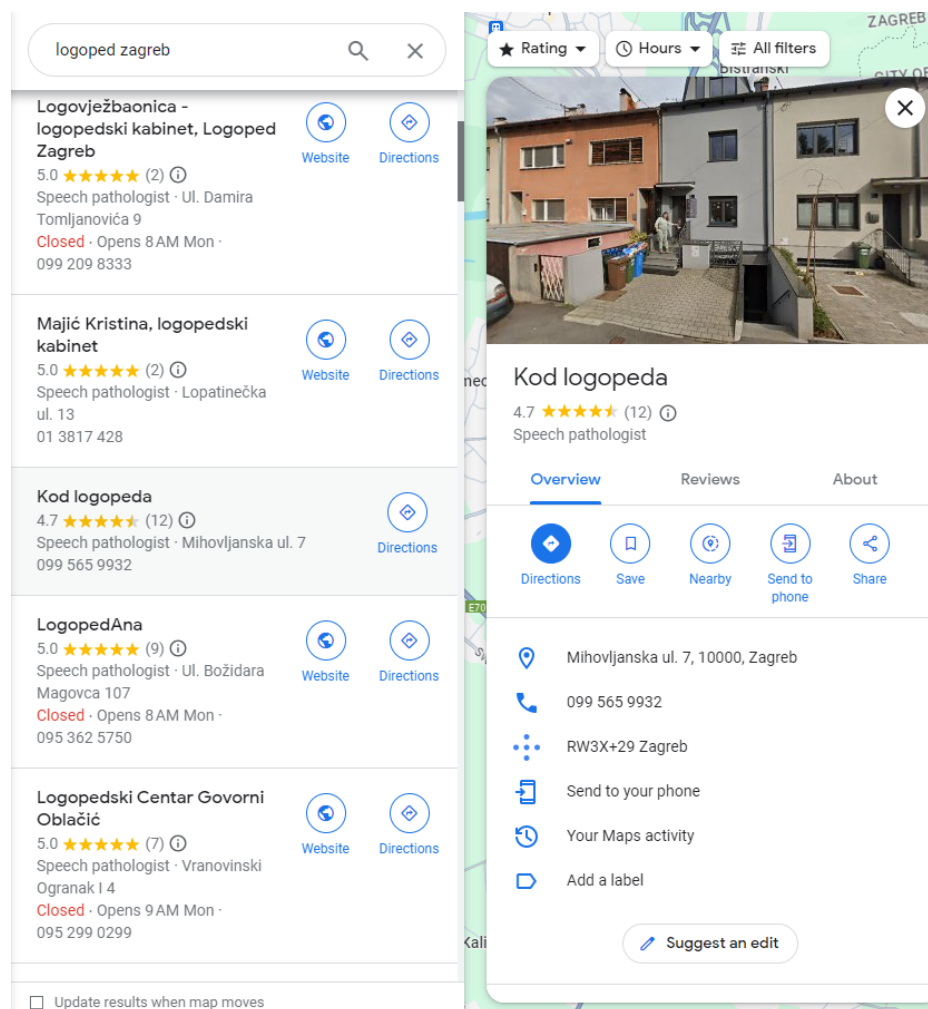
Slika 4.6. Dio skripte za listanje po kabinetima



Slika 4.7. Lista kabineta koja se otvara u virtualnom web-pregledniku s virtualnim mišem

U posljednjem dijelu skripte spremaju se podaci za svaki pojedini kabinet. Pomoću funkcije *locator* i *all* svi kabineti se spremaju u novu klasu *KabinetList*. Prije prolaska kroz svaki kabinet potrebno je definirati gdje se u *Google kartama* (*Google Maps*) može preuzeti informacija o adresi, imenu, web-stranici i službenom broju svakog kabineta. Zato se prethodno definiraju te lokacije kako bi se mogli pretražiti prilikom iteracije kroz svaki kabinet.

Pomoću *for* petlje provodi se iteracija kroz svaki kabinet. Pozivom naredbe *click* na pojedini kabinet otvara se dio stranice gdje su pokazani svi relevantni podaci o kabinetu (slika 4.8). Zatim se kreira nova klasa *Kabinet* u koju se spremaju podaci. Ako neki od podataka nedostaje, onda se ostavlja prazno polje. Kada se svi podaci spreme, klasa *Kabinet* dodaje se u prethodno stvorenu klasu *KabinetList*. Nakon što je zadnji kabinet dodan u klasu *KabinetList*, svi podaci unutar liste spremaju se u datoteku formata *.CSV te se virtualni preglednik zatvara (slika 4.9).



Slika 4.8. Dio Google karata odakle web-strugač preuzima podatke


```

listings = page.locator('//a[@class="hfpxzc"]').all()
kabinet_list = KabinetList();
name_attribute = 'aria-label'
address_xpath = '//button[@data-item-
id="address"]//div[contains(@class, "fontBodyMedium")]\'
website_xpath = '//a[@data-item-id="authority"]//div[contains(@class,
"fontBodyMedium")]\'
phone_number_xpath = '//button[contains(@data-item-id,
"phone:tel:")]//div[contains(@class, "fontBodyMedium")]\'
list_length = len(listings)
print(f"Length of list: {list_length}")
for listing in listings:
    listing.click()
    page.wait_for_timeout(5000)
    kabinet = Kabinet()
    if len(listing.get_attribute(name_attribute)) >= 1:
        kabinet.name = listing.get_attribute(name_attribute)
    else:
        kabinet.name = ""
    if page.locator(address_xpath).count() > 0:
        kabinet.address =
page.locator(address_xpath).all()[0].inner_text()
    else:
        kabinet.address = ""
    if page.locator(website_xpath).count() > 0:
        kabinet.website =
page.locator(website_xpath).all()[0].inner_text()
    else:
        kabinet.website = ""
    if page.locator(phone_number_xpath).count() > 0:
        kabinet.phone_number =
page.locator(phone_number_xpath).all()[0].inner_text()
    else:
        kabinet.phone_number = ""

    kabinet.latitude, kabinet.longitude =
extract_coordinates(page.url)
    kabinet_list.kabinet_list.append(kabinet)

    print(f"Entry added: {kabinet.name}")

kabinet_list.save_to_csv()
browser.close()

```

Slika 4.9. Dohvaćanje podataka za svaki kabinet, spremanje u listu i u CSV datoteku

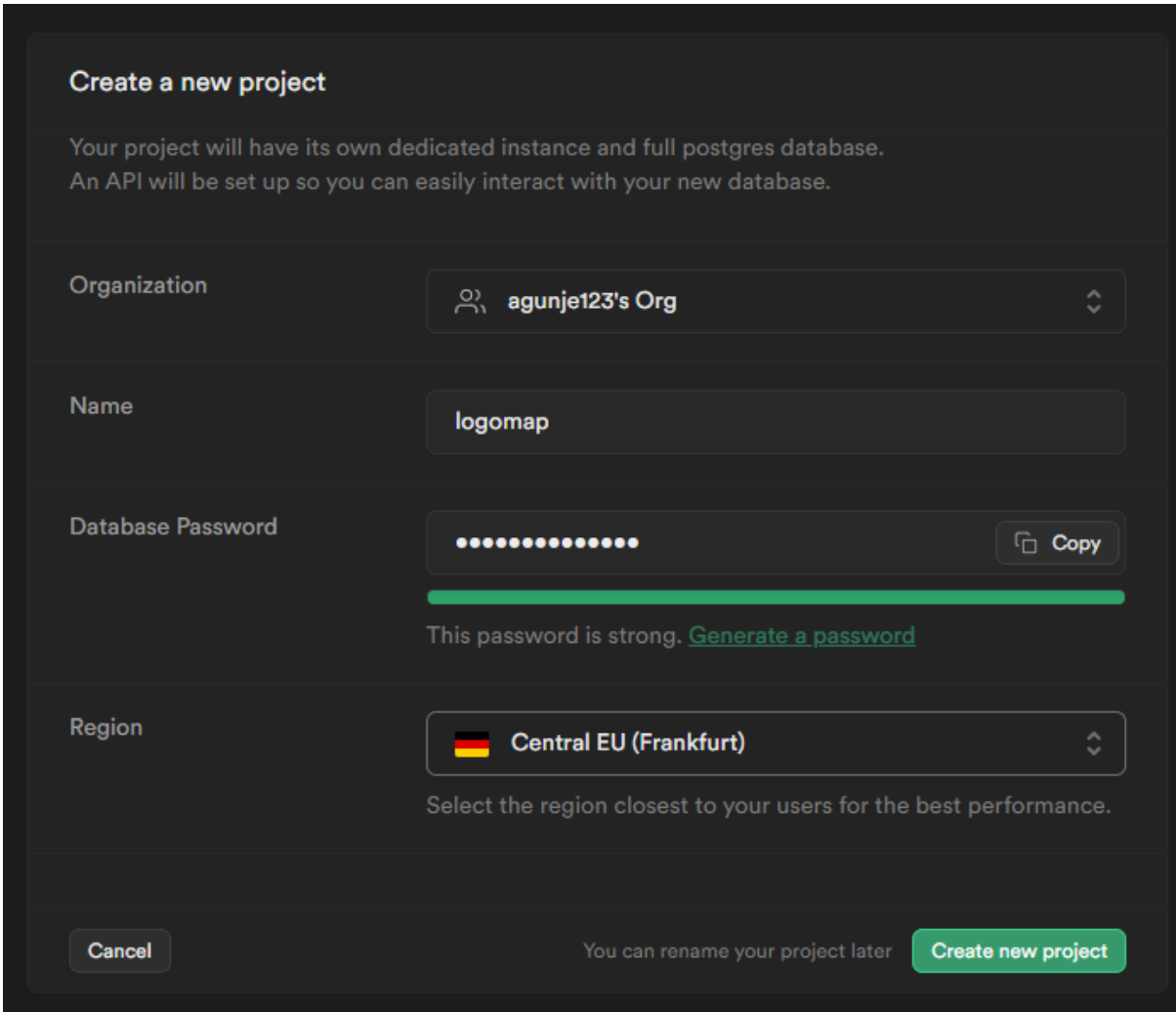
Isti rezultat može se postići ručnim prepisivanjem tih podataka u dokument. Ovim načinom osigurava se eliminacija ljudskog faktora, primarno grešaka pri prepisivanju. U tablici 4.1 prikazano je prvih pet redaka dobiveni iz skripte za web-struganje. Može se vidjeti kako neki kabineti nemaju sve podatke, poput web-stranice ili službenog broja.

Tablica 4.1. Primjer podataka o logopedskim kabinetima dobiveni iz skripte za web-struganje

ID	NAZIV	ADRESA	WEB-STRANICA	BROJ	GEOGRAF. ŠIRINA	GEOGRAF. DUŽINA
1	LOGOTHERAPIA D.O.O.	Ul. Marijana Derenčina 27, 10000, Zagreb	logotherapia.hr	01 4618 600	45.8107543	15.9221728
2	LOGOKOR d.o.o.	Medulićeva ul. 14/3, 10000, Zagreb	logokor.hr	01 4610 261	45.8107543	15.9221728
3	Ruzicasti Oblak Logopedski Kabinet	Prisavlje 10, 10000, Zagreb	ruzicastioblak.hr		45.8107543	15.9221728
4	Logovježbaonica - logopedski kabinet, Logoped Zagreb	Ul. Damira Tomljanovića 9, 10020, Zagreb	logovjezbaonica.hr	099 209 8333	45.8107543	15.9221728
5	Kraljev govor	Importanne Galleria, Trg Drage Iblera 10, 10000, Zagreb			45.8107543	15.9221728

4.2 KREIRANJE BAZE PODATAKA

Nakon što su prikupljeni potrebni podaci, može se kreirati PostgreSQL baza podataka. Potrebno je zadati ime projekta, šifru baze podataka i područje gdje će biti locirana baza podataka (slika 4.10). Nakon nekoliko minuta čekanja, nova baza podataka je kreirana u odabranom području. Budući da je u interesu što brža internetska veza prema bazi podataka, odabran je Frankfurt kao lokacija gdje će se baza nalaziti.



Create a new project

Your project will have its own dedicated instance and full postgres database.
An API will be set up so you can easily interact with your new database.

Organization: agunje123's Org

Name: logomap

Database Password: [masked] [Copy](#)

This password is strong. [Generate a password](#)

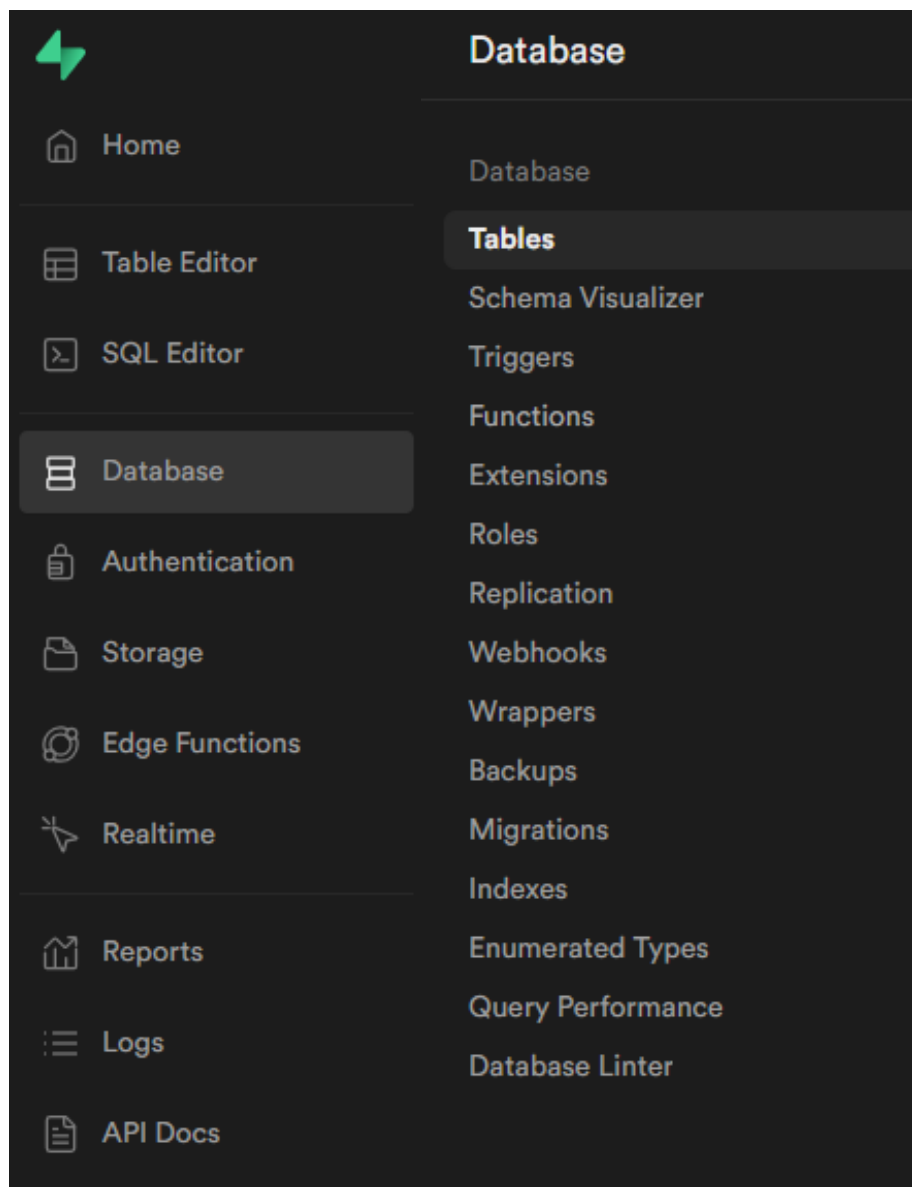
Region: Central EU (Frankfurt)

Select the region closest to your users for the best performance.

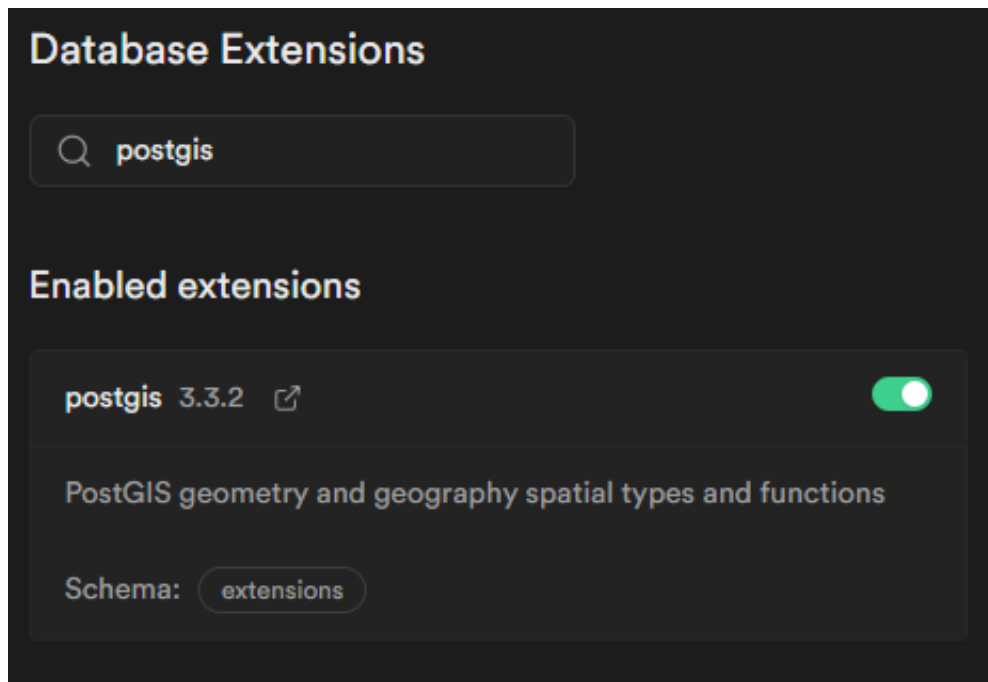
[Cancel](#) You can rename your project later [Create new project](#)

Slika 4.10. Početne postavke za bazu podataka

Nakon kreiranja baze podataka, pojavljuje se bočna navigacijska traka. Na njoj se nalazi sve što je moguće provoditi unutar stranice Supabase. Za potrebe diplomskog rada, korištena je samo baza podataka. Prije kreiranja tablice potrebne za web-stranicu, potrebno je instalirati PostGIS proširenje u PostgreSQL bazu podataka. Pritiskom na *Database* potom na *Extensions* (slika 4.11) otvara se lista svih mogućih proširenja. Pretragom pojma *postgis* dobiva se pet različitih PostGIS proširenja. Za kreiranje baze podataka u okviru diplomskog rada korišten je samo običan PostGIS (slika 4.12).



Slika 4.11. Bočna navigacijska traka



Slika 4.12. Instalirano PostGIS proširenje

Nakon instaliranja PostGIS-a, kreira se tablica potrebna za web-stranicu. Pritiskom na *Database* pa *SQL Editor* otvara se prazni uređivač teksta gdje se mogu pisati *SQL* skripte. Na temelju navedenoga, kreirana je tablica s poljima koja su dohvaćena pomoću web-strugača (slika 4.13).

```
CREATE TABLE kabineti (  
  id SERIAL PRIMARY KEY,  
  name VARCHAR(100),  
  address VARCHAR(255),  
  website VARCHAR(255),  
  phone_number VARCHAR(20),  
  geometry GEOMETRY(Point, 4326)  
);
```

Slika 4.13. Naredba za kreiranje nove tablice

Na kraju je potrebno stvoriti novu funkciju koja će se pozivati kako bi se pozvala lista kabineta. U istom dijelu stranice gdje je napisana skripta za dohvaćenje tablice, napisana je i funkcija za dohvaćanje kabineta (slika 4.14).

```

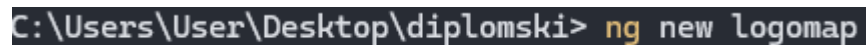
create or replace function get_kabinet_list()
returns table (
  id integer,
  name text,
  address text,
  website text,
  phone_number text,
  longitude float,
  latitude float
) language sql as $$
select
  id,
  name,
  address,
  website,
  phone_number,
  ST_Y(geometry::geometry) AS longitude,
  ST_X(geometry::geometry) AS latitude
FROM
  public.kabineti;
$$;

```

Slika 4.14. Naredba za stvaranje nove funkcije

4.3 IZRADA WEB-STRANICE I WEB-KARTE

Prije početka pisanja koda za izradu web-stranice, potrebno je generirati novu Angular aplikaciju. To se izvodi pomoću naredbe `ng new` (slika 4.15). Nakon pokretanja naredbe, cijela struktura Angular aplikacije generira se automatski te se može započeti s izradom web-stranice (slika 4.16).

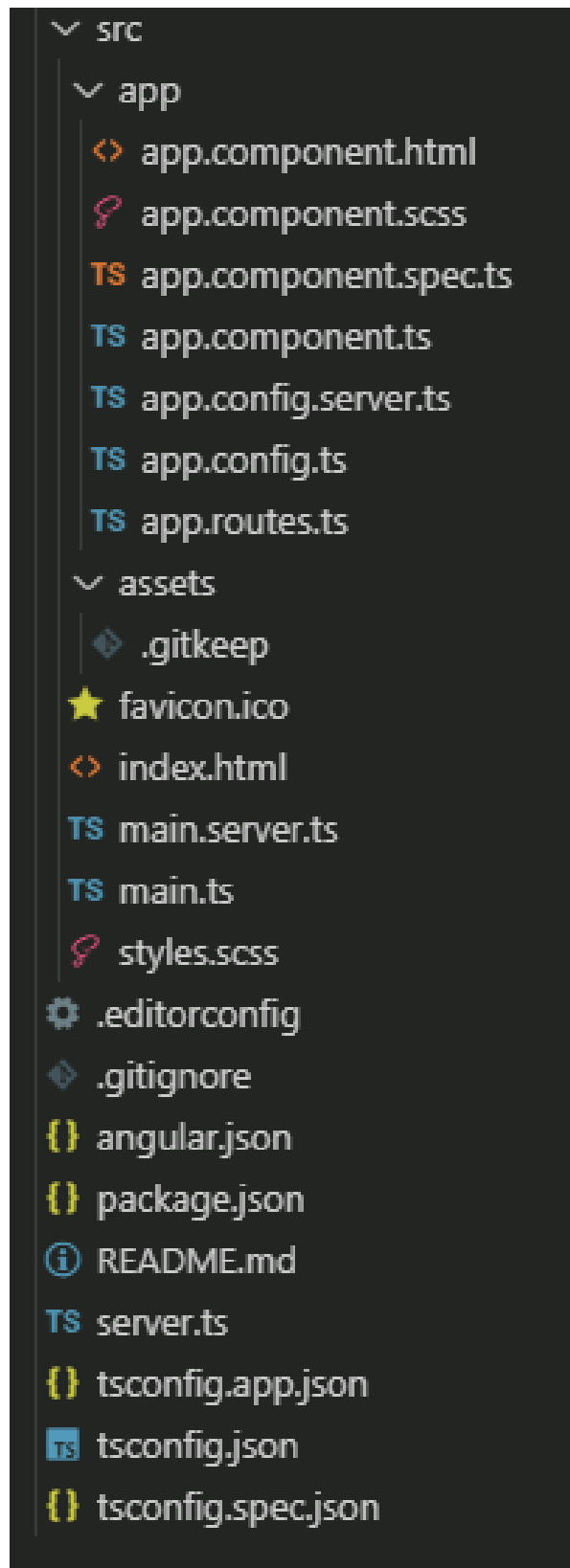


```

C:\Users\User\Desktop\diplomski> ng new logomap

```

Slika 4.15. Naredba za stvaranje nove Angular aplikacije



Slika 4.16. Generirana struktura Angular aplikacije

4.3.1 Servis Supabase

Za stvaranje naredbi za slanje i dohvaćanje podataka o kabinetima iz baze podataka potrebno je izraditi servis gdje će se sve potrebne funkcije zapisati. Servis se generira pomoću naredbe *ng generate*. Nakon naredbe, treba definirati što je potrebno generirati. U ovom slučaju to je servis i njegovo ime (slika 4.17).

```
C:\Users\User\Desktop\diplomski> ng generate service supabase
```

Slika 4.17. Naredba za stvaranje servisa

U generirani servis zapisuje se sve potrebno za bazu podataka. Prvo se zapisuje poveznica preko koje se direktno spaja na bazu podataka. Osim toga, potrebno je zapisati ključ za poveznicu kako bi baza podataka znala autentificirati zahtjeve za slanje i dohvaćanje podataka sa web-stranice. Na kraju, instancira se veza na bazu podataka i ista se može koristiti u budućim zahtjevima (slika 4.18).

```
protected supabase: SupabaseClient;  
  
constructor() {  
  this.supabase = createClient(  
    environment.supabaseUrl,  
    environment.supabaseKey  
  );  
}
```

Slika 4.18. Poveznica i ključ za bazu podataka

U istome servisu nalaze se i dvije funkcije: jedna koja se poziva kako bi se prethodno prikupljena lista kabineta spremila u bazu podataka i druga koja se poziva kako bi se ti podaci dohvatili direktno iz baze podataka.

Funkcija *pushKabinetList* za argument koristi listu kabineta. Svaki kabinet unutar te liste se mijenja na način da se stvara novo polje *geometry*, koje koristi geografsku širinu i dužinu svakog kabineta te ih pretvara u naredbu *Point* prikladnu za PostGIS (slika 4.19).

```
async pushKabinetList(kabinetList: Kabinet[]) {
  const insertData = kabinetList.map((kabinet) => {
    return {
      name: kabinet.name,
      address: kabinet.address,
      website: kabinet.website,
      phone_number: kabinet.phone_number,
      geometry: `POINT(${kabinet.longitude} ${kabinet.latitude})`,
    };
  });
  return this.supabase.from('kabineti').insert(insertData).select();
}
```

Slika 4.19. Funkcija *pushKabinetList*

Funkcija *getKabinetList* nema nikakvih argumenata. Poziva se funkcija *get_kabinet_list()* u bazi podataka (slika 4.14). Kada baza podataka pošalje listu svih kabineta, ta lista se prosljeđuje dalje u aplikaciju (slika 4.20).

```
async getKabinetList() {
  let { data, error } = await this.supabase.rpc('get_kabinet_list');
  if (error) {
    console.error(error);
  } else {
    this.kabinetListSub.next(data);
  }
}
```

Slika 4.20. Funkcija *getKabinetList*

4.3.2 Komponenta za tablicu

Za potrebe web-stranice stvorene su dvije komponente. Prva od njih je komponenta koja sadržava tablicu sa svim kabinetima i gumb pomoću kojega se raščlanjuje CSV datoteka svih kabineta u listu kabineta koja se može slati u bazu podataka. Nova komponenta također se može generirati pomoću naredbe `ng generate`. Međutim, potrebno je naglasiti da se stvara komponenta a ne servis (slika 4.21).

```
C:\Users\User\Desktop\diplomski> ng generate component table
```

Slika 4.21. Naredba za stvaranje komponente

Za prikaz svih kabineta unutar tablice i kasnije unutar web-karte, potrebno je raščlaniti retke unutar CSV datoteke. Prva funkcija `parseCsvRow` služi za raščlanjivanje pojedinog retka CSV datoteke, pretvarajući ga u niz stupaca. Funkcija za argument koristi cijeli redak. Stvara se prazan niz `columns` koji sadržava vrijednosti stupaca iz CSV retka. Prolazi se kroz svaki znak u retku i provjerava se je li trenutni znak zarez, a nije unutar navodnika. To je potrebno kontrolirati jer se u nekim nazivima kabineta nalaze zarezi. Ako zarez nije unutar navodnika, to znači da se došlo do kraja trenutnog stupca. Dodaje se trenutni stupac u niz `columns`, očisti se `currentColumn` za sljedeći stupac, i nastavlja se s iteracijom. Na kraju, funkcija vraća niz `columns` koji sadrži sve stupce iz raščlanjenog retka CSV datoteke (slika 4.22).

```
parseCsvRow(row: string): string[] {
  const columns = [];
  let currentColumn = '';
  let withinQuotes = false;
  for (let i = 0; i < row.length; i++) {
    const char = row[i];
    if (char === ',' && !withinQuotes) {
      columns.push(currentColumn.trim());
      currentColumn = '';
    } else if (char === '"' && (i === 0 || row[i - 1] !== '\\')) {
      withinQuotes = !withinQuotes;
    } else {
      currentColumn += char;
    }
  }
  columns.push(currentColumn.trim());
  return columns;
}
```

Slika 4.22. Funkcija `parseCsvRow`

Druga funkcija *parseCsv* služi za čitanje cijelog sadržaja CSV datoteke. Prvo se učitava datoteka i pretvara u tekst. Taj tekst se zatim sprema u varijablu *csv* koja se razdvaja na retke i sprema u varijablu *rows*. Zatim se za svaki redak poziva prethodno objašnjena funkcija *parseCsvRow* kako bi se razdvojili stupci i dobio niz vrijednosti. Stupci se zatim zapisuju u novi kabinet koji predstavlja jedan podatak o logopedskom kabinetu. Nakon što se svi podaci obrade, podaci se prikazuju u tablici i poziva se funkcija *pushKabinetList* kako bi se podaci zapisali u bazu podataka (slika 4.23).

```
parseCsv(file: File) {
  const reader = new FileReader();
  reader.readAsText(file);
  reader.onload = () => {
    const csv = reader.result as string;
    const rows = csv.split('\n');
    for (let i = 1; i < rows.length; i++) {
      const row = rows[i];
      const columns = this.parseCsvRow(row);
      const kabinet: Kabinet = {
        id: parseInt(columns[0]) + 1,
        name: columns[1],
        address: columns[2],
        website: columns[3],
        phone_number: columns[4],
        longitude: parseFloat(columns[5]),
        latitude: parseFloat(columns[6]),
      };
      if (kabinet.id) {
        this.data.push(kabinet);
      }
    }
    this.dataSource.data = this.data;
    this.supabaseService.pushKabinetList(this.data);
  };
  reader.onerror = () => {
    console.error('Error reading CSV file');
  };
}
```

Slika 4.23. Funkcija *parseCsv*

Nakon završetka funkcije *parseCsv*, podatke više nije potrebno učitavati jer su zapisani u bazu podataka. U svrhu prikazivanja podataka, stvorena je tablica koristeći biblioteku gotovih komponenti *Angular material* (slika 4.24). Ta biblioteka omogućuje brži razvoj tablice jer sadrži već stilizirane komponente koje je lako moguće modificirati potrebama web-stranice. Upotrijebljena je gotova tablica *mat-table* i u njoj se prikazuju podatci koji se dohvaćaju pomoću funkcije *getKabinetList* (slika 4.20).



Ime Logopedskog Kabineta	Adresa	Web-stranica	Kontakt broj
Logovjezbaonica - logopedski kabinet, Logoped Zagreb	Ul. Damira Tomljanovića 9, 10020, Zagreb	logovjezbaonica.hr	099 209 8333
Logopedski kabinet Rafael	Lastovska ul. 2A, 10000, Zagreb	logoped-rafael.hr	099 734 6399
LogopedAna	Ul. Božidara Magovca 107, 10000, Zagreb		095 362 5750
Logopedski Centar Govorni Oblačić	Vranovinski Ogranak I 4, 10000, Zagreb	logopedski-centar.com	095 299 0299
LOGOTHERAPIA D.O.O.	Ul. Marijana Derenčina 27, 10000, Zagreb	logotherapia.hr	01 4618 600
Jezik i govor - Logopedska dijagnostika i terapija	Vajdin vijenac 2, 10000, Zagreb		01 6600 156
ProLogo kabinet	Ul. Davora Zbiljskog 36, 10000, Zagreb		098 954 8966
Majić Kristina, logopedski kabinet	Lopatinečka ul. 13, 10000, Zagreb	logopedskikabinet.com	01 3817 428
Centar za jezik i govor Vrutak	Ulica Miroslava Kraljevića 6, 10000, Zagreb	centar-vrutak.eu	01 2929 140

Slika 4.24. Tablica s podacima o logopedskim kabinetima

4.3.3 Komponenta za kartu

Druga komponenta sadržava kartu sa svim lokacijama kabineta. Kako bi se prikazali svi logopedski kabineti na području Grada Zagreba, prvo je potrebno instancirati *Leaflet* kartu. Prije instanciranja definirane su varijable koje pomažu pri izradi karte (slika 4.25). Varijabla *map* omogućuje rukovanje nad elementima karte, varijabla *centroid* su koordinate centroida područja koje je prikazano na karti. Za potrebe izrade ove karte zadane su koordinate centroida Grada Zagreba. Na kraju se definira varijabla *markerIcon* koja određuje izgled znaka kojim će se prikazivati svaki logopedski kabinet.

```
private map!: L.Map;
private centroid: L.LatLngExpression = [45.81, 15.98];
private markerIcon = {
  icon: L.icon({
    iconSize: [25, 41],
    iconAnchor: [10, 41],
    popupAnchor: [2, -40],
    iconUrl: 'https://unpkg.com/leaflet@1.5.1/dist/images/marker-icon.png',
    shadowUrl:
      'https://unpkg.com/leaflet@1.5.1/dist/images/marker-shadow.png',
  }),
};
```

Slika 4.25. Pomoćne varijable za izradu karte

Također, prije inicijalizacije karte, definira se nekoliko pomoćnih funkcija. Prva funkcija zove se *createPopup* (slika 4.26). Unutar te funkcije definira se tekst koji se prikazuje prilikom odabira pojedinog kabineta na karti. Za argument funkcija koristi jedan kabinet te za svaki taj kabinet automatski dodjeljuje ime i adresu kabineta, jer svaki kabinet ima te vrijednosti. Nakon toga, provjerava se je li kabinet ima web-stranicu i broj mobitela. Ako ih ima, onda se dodaju na postojeći tekst, a ako nema, onda se taj tekst ne prikazuje.

```

createPopup(kabinet: Kabinet) {
  let popupContent =
    `

Ime Kabineta: ${kabinet.name}</div>` +
    `

Adresa: ${kabinet.address}</div>`;

  if (kabinet.website) {
    popupContent += `

Web-stranica: ${kabinet.website}</div>`;
  }

  if (kabinet.phone_number) {
    popupContent += `

Kontakt broj: ${kabinet.phone_number}</div>`;
  }

  return popupContent;
}


```

Slika 4.26. Funkcija *createPopup*

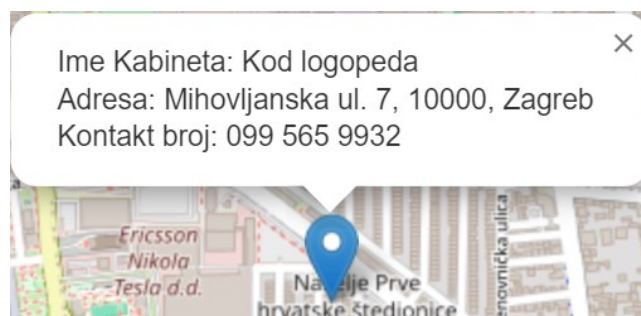
Druga pomoćna funkcija zove se *createMarkerWithPopup* (slika 4.27). Funkcija prima kabinet kao argument. Iz tog kabineta iščitavaju se koordinate te se na točku s koordinatama kabineta postavlja znak koji je prethodno definiran pomoću varijable *markerIcon*. Također, stvara se tekst koji se prikazuje klikom na kabinet pomoću prethodno definirane funkcije *createPopup* (slika 4.28).

```

createMarkerWithPopup(kabinet: Kabinet) {
  let marker = L.marker(
    [kabinet.latitude!, kabinet.longitude!],
    this.markerIcon
  );
  let popup = this.createPopup(kabinet);
  marker.addTo(this.map);
  marker.bindPopup(popup);
}

```

Slika 4.27. Funkcija *createMarkerWithPopup*



Slika 4.28. Primjer znaka kabineta sa svim dostupnim podacima

Pomoću prethodno definiranih varijabli i funkcija, može se pristupiti inicijalizaciji *Leaflet* karte. Za tu svrhu stvorena je funkcija *initMap*. Unutar funkcije postavljeno je da se karta prikazuje na prethodno definiranoj varijabli *centroid*, na određenoj razini povećanja. Preuzimaju se slojevi *tile* koji su potrebni za prikazivanje karte, određuje se maksimalna i minimalna razina mogućeg povećanja karte i pripisuju potrebne zasluge stranici *OpenStreetMap* odakle su preuzeti potrebni slojevi. Slojevi se dodaju na prethodno instanciranu kartu te se prikazuju (slika 4.29).

```
initMap() {
  this.mapService.kabinetSub.subscribe({
    next: (kabinet) => {
      this.panToKabinet(kabinet);
    },
  });

  this.map = L.map('map', {
    center: this.centroid,
    zoom: 13,
  });

  const tiles = L.tileLayer(
    'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
    {
      maxZoom: 18,
      minZoom: 12,
      attribution:
        '&copy; <a
href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>',
    }
  );

  tiles.addTo(this.map);

  this.supabaseService.getKabinetList();
}
```

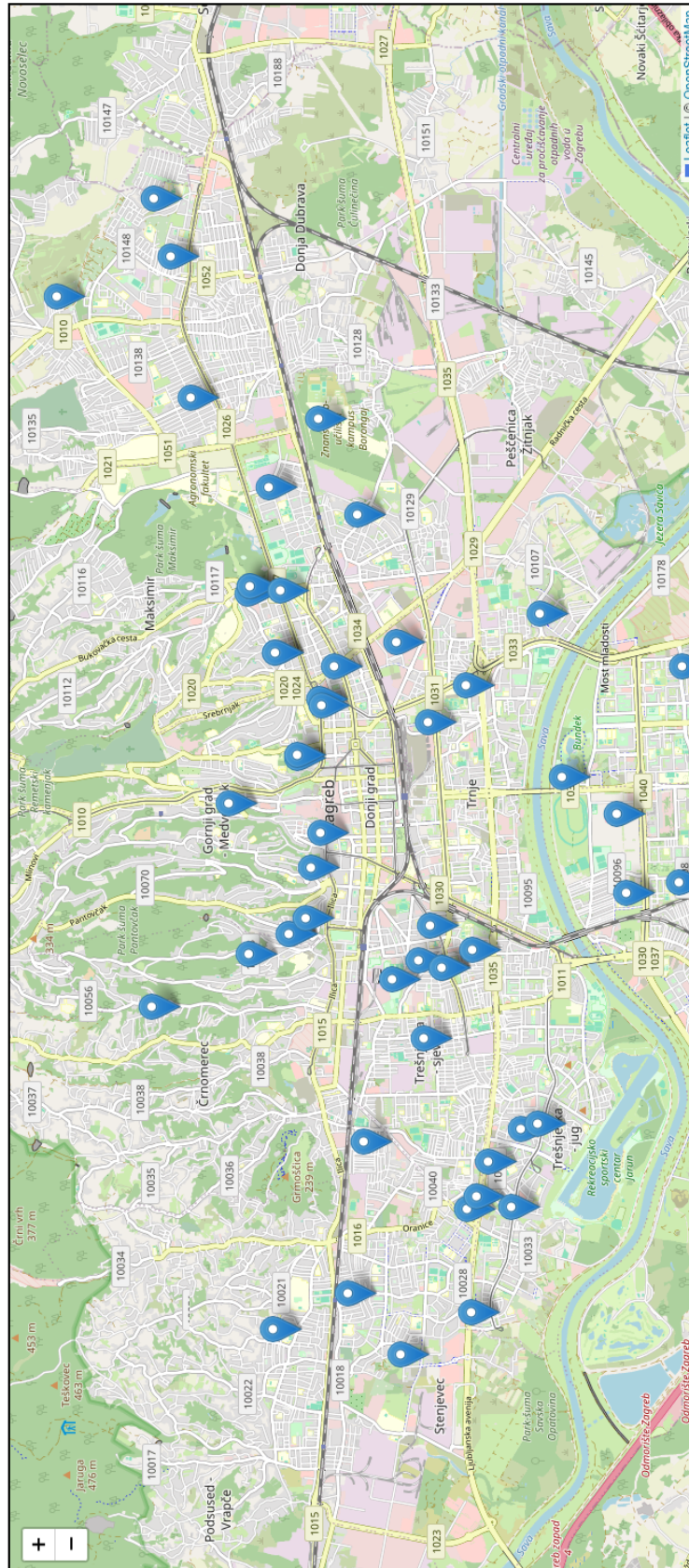
Slika 4.29. Funkcija *initMap*

Važan dio funkcije *initMap* je pretplata na Subject *kabinetSub*. Koristi se kada se klikne na određeni redak u tablici da se pozove funkcija *panToKabinet*, koja pomiče kartu na odabrani kabinet. Također, tek na kraju funkcije *initMap* poziva se funkcija *getKabinetList* kako bi se osiguralo da se karta pokrene prije nego se na nju dodaju svi kabineti.

Kako bi se osiguralo da se svi kabineti prikažu na karti, stvorena je funkcija *ngOnInit* koja se automatski pokreće pri pokretanju karte (slika 4.30). Unutar te funkcije nalazi se pretplata na Subject *kabinetListSub*. Nakon što se kabineti dohvate iz baze podataka, unutar pretplate se za svaki kabinet poziva metoda *createMarkerWithPopup*. Na kraju funkcije *ngOnInit*, poziva se funkcija *initMap* kojom se prikazuje karta (slika 4.31).

```
ngOnInit(): void {
  this.subs.add(
    this.supabaseService.kabinetListSub.subscribe({
      next: (kabineti) => {
        kabineti.forEach((kabinet) => {
          this.createMarkerWithPopup(kabinet);
        });
      },
    })
  );
  this.initMap();
}
```

Slika 4.30. Funkcija *ngOnInit*



Slika 4.31. Karta logopedskih kabineta

5. REZULTATI I RASPRAVA

U sklopu diplomskog rada izrađene su tri stavke: web-strugač, baza podataka i web-stranica s web-kartom. Cijeli kod javno je dostupan na URL 22 i URL 23.

5.1 WEB-STRUGAČ

Prvi dio diplomskog rada je napisana skripta za web-struganje (slike 5.1 i 5.2). Web-strugač je izrađen na način da se može promijeniti područje pretraživanja. Trenutno se unutar skripte pretražuje pojam *logoped zagreb*, ali to se može prilagoditi za bilo koje područje od interesa. Tako se na jednostavan način može doći do podataka o logopedskim kabinetima u drugim gradovima. Također, postoji mogućnost nadogradnje skripte kako bi se prikupljalo više podataka.

Prednost ovog pristupa je fleksibilnost skripte koja se izvršava, brzina izvršavanja i lako obradiva datoteka s podacima formata *.CSV.

Nedostatak web-strugača je ovisnost o listi kabineta dostupnih na *Google Mapsu*. Korištenjem skripte više puta dolazi do problema gdje *Google Maps* ne prikazuje uvijek iste podatke, već varira u broju pronađenih kabineta. Takav problem onemogućuje automatizaciju skripte koja bi se izvodila periodički i mijenjala postojeće podatke unutar baze podataka.

5.2 BAZA PODATAKA

Drugi dio izrađen u sklopu diplomskog rada je baza podataka. Unutar web-aplikacije Supabase nalazi se baza podataka sa svim potrebnim podacima o logopedskim kabinetima: ime, adresa, adresa web-stranice, telefonski broj te lokacija u koordinatnom sustavu WGS84 (slika 5.3). Budući da se koristi PostGIS dodatak za PostgreSQL bazu podataka, podaci se mogu na razne načine dohvaćati i prikazivati. Uz to, mijenja se format lokacije iz PostGIS geometrije u geografsku širinu i dužinu. Ti podaci se koriste i prikazuju unutar web-stranice. Slanje kabineta u bazu podataka izvodi se pomoć datoteke *supabase.service.ts* (slika 5.4). U toj datoteci definirane su funkcije slanja i dohvaćanja podataka iz baze.

Prednost korištenja baze podataka za web-stranicu je ažuriranje podataka u stvarnom vremenu. Korištenjem CSV datoteke dobivene iz web-strugača zahtjevalo bi ponovno postavljanje web-stranice pri svakom mijenjanju podataka.

Nedostatak korištenja baze podataka su sigurnosni rizici o kojima se treba ozbiljno brinuti. Zbog neovlaštenih pristupa potrebno je više predznanja o mogućim sigurnosnim rizicima i stalno učenje o potencijalno novim vrstama napada.

5.3 WEB-STRANICA

Web-stranica izrađena u Angularu sastoji se od dva dijela: karte i tablice kabineta.

U prvoj komponenti *map.component* (slike 5.5, 5.6, 5.7 i 5.8) prikazana je karta sa svim kabinetima (slika 5.9). Karta je u potpunosti interaktivna. Na karti je moguće pomicati, približavati ili udaljavati područje od interesa, odabirati željene kabinete te pregledavati njihove podatke. Sve to se dinamički dohvaća iz baze podataka. Kad bi se dodalo više kabineta unutar baze podataka, isti bi bili i prikazani na karti skupa sa svojim podacima.

Druga komponenta *table.component* (slike 5.10, 5.11, 5.12 i 5.13) sastoji se od tablice sa svim kabinetima (slika 5.14) radi preglednijeg traženja željenog kabineta. Unutar tablice nalaze se svi podaci koji su prikupljeni i zapisani u bazu podataka, osim koordinata. Također, klikom na redak nekog kabineta stranica koristi datoteku *map.service.ts* (slika 5.15) i automatski prikazuje odabrani kabinet na karti (slika 5.16).

Prednosti ovog pristupa izradi web-stranice i web-karte je brzi razvoj web-stranice. Razvojni okvir Angular je pogodan za izrađivanje robusnih web-aplikacija, pogotovo jer postoje brojni materijali na internetu iz kojih se može učiti.

Nedostatak ovog pristupa je brzina same web-stranice. Iako je Angular odličan izbor za brz razvoj web-aplikacija, nije nužno potreban za ovako kompaktne web-stranice. Budući da Angular u pozadini ima dosta biblioteka i kodova koji su u web-stranici izrađenoj u okviru diplomskog rada nepotrebni, to znači da je stranica sporija nego što bi trebala biti.

```

1 from playwright.sync_api import sync_playwright
2 from dataclasses import dataclass, asdict, field
3 import pandas as pd
4 import os
5
6 @dataclass
7 class Kabinet:
8     name: str = None
9     address: str = None
10    website: str = None
11    phone_number: str = None
12    latitude: float = None
13    longitude: float = None
14
15 @dataclass
16 class KabinetList:
17     kabinet_list: list[Kabinet] = field(default_factory=list)
18     save_at = 'output'
19
20     def dataframe(self):
21         return pd.json_normalize(
22             (asdict(kabinet) for kabinet in self.kabinet_list), sep="_"
23         )
24
25     def save_to_csv(self):
26         if not os.path.exists(self.save_at):
27             os.makedirs(self.save_at)
28         self.dataframe().to_csv(f"output/kabineti.csv")
29
30     def extract_coordinates(url: str) -> tuple[float, float]:
31         coordSubstring = url.split('!3d')[-1]
32         latitude = coordSubstring.split('!4d')[0]
33         longitude = coordSubstring.split('!4d')[1].split('!')[0]
34         return float(latitude), float(longitude)
35
36     def main():
37         with sync_playwright() as p:
38             browser = p.chromium.launch()
39             page = browser.new_page()
40
41             page.goto("https://www.google.com/maps/search/logoped+zagreb/", timeout=60000)
42
43             page.get_by_role("button", name="Prihvati sve").click()
44             page.wait_for_timeout(500)
45
46             page.hover('//a[@class="hfpxzc"][1]')
47
48             while True:
49                 page.mouse.wheel(0, 10000)
50                 page.wait_for_timeout(3000)
51
52                 end_of_list = page.query_selector('//div[contains(@class, "tLjsW")]')
53
54                 if end_of_list:
55                     print("Reached the end of the list.")
56                     break
57                 else:
58                     print("Scrolling...")

```

Slika 5.1. Prvi dio skripte za web-struganje

```

1 listings = page.locator('//a[@class="hfpxzc"]').all()
2 kabinet_list = KabinetList();
3
4 name_attribute = 'aria-label'
5 address_xpath = '//button[@data-item-id="address"]//div[contains(@class, "fontBodyMedium")]'
6 website_xpath = '//a[@data-item-id="authority"]//div[contains(@class, "fontBodyMedium")]'
7 phone_number_xpath = '//button[contains(@data-item-id, "phone:tel:")]//div[contains(@class, "fontBodyMedium")]'
8
9 list_length = len(listings)
10 print(f"Length of list: {list_length}")
11
12 for listing in listings:
13     listing.click()
14     page.wait_for_timeout(5000)
15
16     kabinet = Kabinet()
17
18     if len(listing.get_attribute(name_attribute)) >= 1:
19         kabinet.name = listing.get_attribute(name_attribute)
20     else:
21         kabinet.name = ""
22     if page.locator(address_xpath).count() > 0:
23         kabinet.address = page.locator(address_xpath).all()[0].inner_text()
24     else:
25         kabinet.address = ""
26     if page.locator(website_xpath).count() > 0:
27         kabinet.website = page.locator(website_xpath).all()[0].inner_text()
28     else:
29         kabinet.website = ""
30     if page.locator(phone_number_xpath).count() > 0:
31         kabinet.phone_number = page.locator(phone_number_xpath).all()[0].inner_text()
32     else:
33         kabinet.phone_number = ""
34
35     kabinet.latitude, kabinet.longitude = extract_coordinates(page.url)
36
37     kabinet_list.kabinet_list.append(kabinet)
38
39     print(f"Entry added: {kabinet.name}")
40
41     kabinet_list.save_to_csv()
42
43     browser.close()
44
45     print("Script finished!")
46
47 if __name__ == "__main__":
48     main()

```

Slika 5.2. Drugi dio skripte za web-struganje

id	name	address	website	phone_number	geometry
66	Logovježbaonica - logopedski kabinet, Logoped Zagreb	UJ. Damira Tomljanovića 9, 10020, Zagreb	logovjzbaonica.hr	099 209 8333	0101000020E6100000929AD1EA
67	Logopedski kabinet Rafael	Lastovska ul. 2A, 10000, Zagreb	logoped-rafael.hr	099 734 6399	0101000020E610000009F2B0D7C
68	LogopedAna	UJ. Božidara Magoeca 107, 10000, Zagreb	EMPTY	095 362 5750	0101000020E610000007028D76F
69	Logopedski Centar Govorni Oblačić	Vranovinski Ogranak 14, 10000, Zagreb	logopedski-centar.com	095 299 0299	0101000020E6100000B6B52A2E
70	LOGOTHERAPIA D.O.O.	UJ. Marijana Derenčina 27, 10000, Zagreb	logotherapia.hr	01 4618 600	0101000020E6100000B2C8B01CC
71	Jezik i govor - Logopedska dijagnostika i terapija	Vajdin vijenac 2, 10000, Zagreb	EMPTY	01 6600 156	0101000020E6100000004F40C2E
72	ProLogo kabinet	UJ. Davora Zbiljskog 36, 10000, Zagreb	EMPTY	098 954 8966	0101000020E6100000098CC672F
73	Majjić Kristina, logopedski kabinet	Lopatinečka ul. 13, 10000, Zagreb	logopedskikabinet.com	01 3817 428	0101000020E6100000042F2295B
74	Centar za jezik i govor Vrutak	Ulica Miroslava Kraljevića 6, 10000, Zagreb	centar-vrutak.eu	01 2929 140	0101000020E610000004474B5A

Slika 5.3. Dio tablice unutar baze podataka

```

1 import { Injectable } from '@angular/core';
2 import { createClient, SupabaseClient } from '@supabase/supabase-js';
3 import { environment } from '../environment';
4 import { Kabinet } from '../model/kabinet';
5 import { Subject } from 'rxjs';
6
7 @Injectable({
8   providedIn: 'root',
9 })
10 export class SupabaseService {
11   kabinetListSub = new Subject<Kabinet[]>();
12
13   protected supabase: SupabaseClient;
14
15   constructor() {
16     this.supabase = createClient(
17       environment.supabaseUrl,
18       environment.supabaseKey
19     );
20   }
21
22   async pushKabinetList(kabinetList: Kabinet[]) {
23     const insertData = kabinetList.map((kabinet) => {
24       return {
25         name: kabinet.name,
26         address: kabinet.address,
27         website: kabinet.website,
28         phone_number: kabinet.phone_number,
29         geometry: `POINT(${kabinet.longitude} ${kabinet.latitude})`,
30       };
31     });
32     return this.supabase.from('kabineti').insert(insertData).select();
33   }
34
35   async getKabinetList() {
36     let { data, error } = await this.supabase.rpc('get_kabinet_list');
37     if (error) {
38       console.error(error);
39     } else {
40       this.kabinetListSub.next(data);
41     }
42   }
43 }

```

Slika 5.4. Datoteka supabase.service.ts

```

1 import { Component, OnDestroy, OnInit } from '@angular/core';
2 import * as L from 'leaflet';
3 import { MapService } from '../services/map.service';
4 import { CommonModule } from '@angular/common';
5 import { MatButtonModule } from '@angular/material/button';
6 import { SupabaseService } from '../services/supabase.service';
7 import { Subscription } from 'rxjs';
8 import { Kabinet } from '../model/kabinet';
9
10 @Component({
11   selector: 'app-map',
12   standalone: true,
13   imports: [CommonModule, MatButtonModule],
14   templateUrl: './map.component.html',
15   styleUrls: ['./map.component.scss'],
16 })
17 export class MapComponent implements OnInit, OnDestroy {
18   private map!: L.Map;
19   private centroid: L.LatLngExpression = [45.81, 15.98];
20   private markerIcon = {
21     icon: L.icon({
22       iconSize: [25, 41],
23       iconAnchor: [10, 41],
24       popupAnchor: [2, -40],
25       iconUrl: 'https://unpkg.com/leaflet@1.5.1/dist/images/marker-icon.png',
26       shadowUrl:
27         'https://unpkg.com/leaflet@1.5.1/dist/images/marker-shadow.png',
28     }),
29   };
30
31   subs: Subscription = new Subscription();
32
33   constructor(
34     private supabaseService: SupabaseService,
35     private mapService: MapService
36   ) {}
37
38   ngOnInit(): void {
39     this.subs.add(
40       this.supabaseService.kabinetListSub.subscribe({
41         next: (kabineti) => {
42           kabineti.forEach((kabinet) => {
43             this.createMarkerWithPopup(kabinet);
44           });
45         },
46       })
47     );
48     this.initMap();
49   }
50
51   ngOnDestroy(): void {
52     this.subs.unsubscribe();
53   }

```

Slika 5.5. Prvi dio datoteke map.component.ts


```

1  initMap() {
2    this.mapService.kabinetSub.subscribe({
3      next: (kabinet) => {
4        this.panToKabinet(kabinet);
5      },
6    });
7
8    this.map = L.map('map', {
9      center: this.centroid,
10     zoom: 13,
11   });
12
13   const tiles = L.tileLayer(
14     'https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png',
15     {
16       maxZoom: 18,
17       minZoom: 12,
18       attribution:
19         '&copy; <a href="http://www.openstreetmap.org/copyright">OpenStreetMap</a>',
20     }
21   );
22
23   tiles.addTo(this.map);
24
25   this.supabaseService.getKabinetList();
26 }
27
28 createMarkerWithPopup(kabinet: Kabinet) {
29   let marker = L.marker(
30     [kabinet.latitude!, kabinet.longitude!],
31     this.markerIcon
32   );
33   let popup = this.createPopup(kabinet);
34   marker.addTo(this.map);
35   marker.bindPopup(popup);
36 }
37
38 createPopup(kabinet: Kabinet) {
39   let popupContent =
40     `

Ime Kabineta: ${kabinet.name}</div>` +
41     `

Adresa: ${kabinet.address}</div>`;
42
43   if (kabinet.website) {
44     popupContent += `

Web-stranica: ${kabinet.website}</div>`;
45   }
46
47   if (kabinet.phone_number) {
48     popupContent += `

Kontakt broj: ${kabinet.phone_number}</div>`;
49   }
50
51   return popupContent;
52 }
53
54 panToKabinet(kabinet: Kabinet) {
55   this.map.panTo(new L.LatLng(kabinet.latitude!, kabinet.longitude!));
56 }
57 }
58


```

Slika 5.6. Drugi dio datoteke map.component.ts



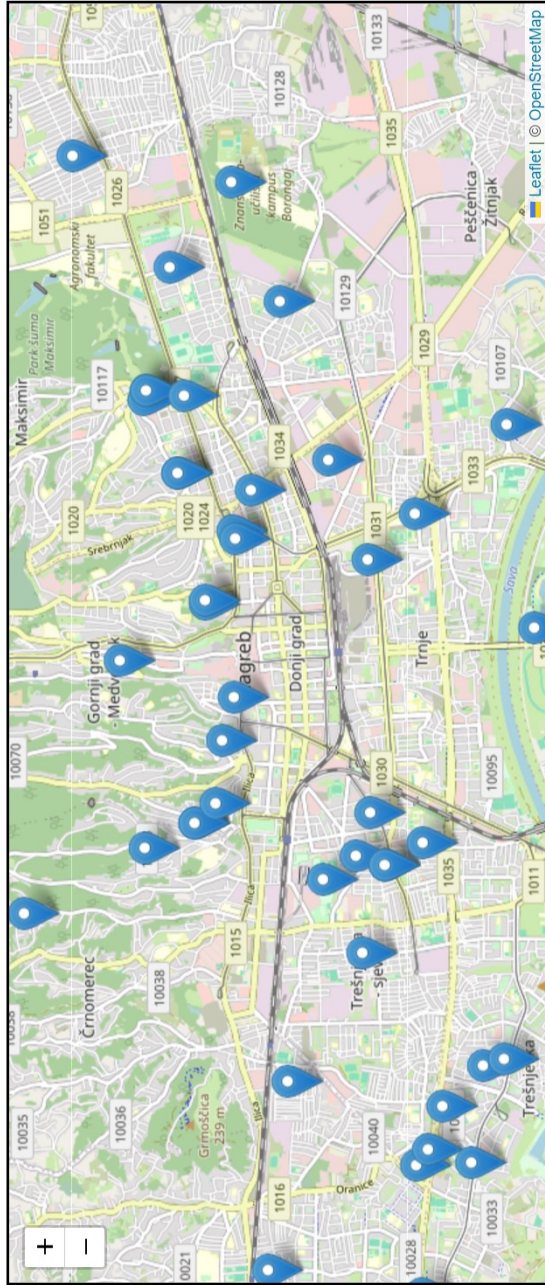
```
1 <div class="map-container">
2   <div class="map-frame">
3     <div id="map"></div>
4   </div>
5 </div>
```

Slika 5.7. Datoteka map.component.html



```
1 .map-container {
2   position: absolute;
3   top: 0;
4   left: 0;
5   right: 0;
6   bottom: 0;
7   margin: 30px;
8   margin-top: 65px;
9   display: flex;
10  justify-content: center;
11  align-items: center;
12 }
13
14 .map-frame {
15   border: 2px solid black;
16   height: 80%;
17   width: 80%;
18 }
19
20 #map {
21   height: 100%;
22 }
```

Slika 5.8. Datoteka map.component.scss



Slika 5.9. Karta u sklopu web-stranice

```

1 import { Component, OnDestroy, OnInit } from '@angular/core';
2 import { MatButtonModule } from '@angular/material/button';
3 import { MatIconModule } from '@angular/material/icon';
4 import { MatTableDataSource, MatTableModule } from '@angular/material/table';
5 import { Kabinet } from '../../model/kabinet';
6 import { SupabaseService } from '../../services/supabase.service';
7 import { Subscription } from 'rxjs';
8 import { CommonModule } from '@angular/common';
9 import { Router } from '@angular/router';
10 import { MapService } from '../../services/map.service';
11
12 @Component({
13   selector: 'app-table',
14   standalone: true,
15   imports: [MatTableModule, MatIconModule, MatButtonModule, CommonModule],
16   templateUrl: './table.component.html',
17   styleUrls: ['./table.component.scss'],
18 })
19 export class TableComponent implements OnInit, OnDestroy {
20   constructor(
21     private supabaseService: SupabaseService,
22     private mapService: MapService,
23     private router: Router
24   ) {}
25
26   displayCSV: boolean = true;
27
28   subs: Subscription = new Subscription();
29
30   dataSource = new MatTableDataSource<Kabinet>([]);
31   displayedColumns: string[] = ['name', 'address', 'website', 'phone_number'];
32   data: Kabinet[] = [];
33
34   ngOnInit(): void {
35     this.subs.add(
36       this.supabaseService.kabinetListSub.subscribe({
37         next: (kabineti) => {
38           if (kabineti.length !== 0) {
39             this.dataSource.data = kabineti;
40             this.displayCSV = false;
41           }
42         },
43       })
44     );
45     this.supabaseService.getKabinetList();
46   }
47
48   ngOnDestroy(): void {
49     this.subs.unsubscribe();
50   }

```

Slika 5.10. Prvi dio datoteke table.component.ts

```

1  onFileSelected(event: any) {
2    const file: File = event.target.files[0];
3    if (file) {
4      this.parseCsv(file);
5    }
6  }
7
8  parseCsv(file: File) {
9    const reader = new FileReader();
10   reader.readAsText(file);
11   reader.onload = () => {
12     const csv = reader.result as string;
13     const rows = csv.split('\n');
14     for (let i = 1; i < rows.length; i++) {
15       const row = rows[i];
16       const columns = this.parseCsvRow(row);
17       const kabinet: Kabinet = {
18         id: parseInt(columns[0]) + 1,
19         name: columns[1],
20         address: columns[2],
21         website: columns[3],
22         phone_number: columns[4],
23         longitude: parseFloat(columns[5]),
24         latitude: parseFloat(columns[6]),
25       };
26       if (kabinet.id) {
27         this.data.push(kabinet);
28       }
29     }
30     this.dataSource.data = this.data;
31     this.supabaseService.pushKabinetList(this.data);
32   };
33   reader.onerror = () => {
34     console.error('Error reading CSV file');
35   };
36 }
37
38 parseCsvRow(row: string): string[] {
39   const columns = [];
40   let currentColumn = '';
41   let withinQuotes = false;
42
43   for (let i = 0; i < row.length; i++) {
44     const char = row[i];
45     if (char === ',' && !withinQuotes) {
46       columns.push(currentColumn.trim());
47       currentColumn = '';
48     } else if (char === '"' && (i === 0 || row[i - 1] !== '\\')) {
49       withinQuotes = !withinQuotes;
50     } else {
51       currentColumn += char;
52     }
53   }
54   columns.push(currentColumn.trim());
55   return columns;
56 }
57
58 showKabinet(kabinet: Kabinet) {
59   this.router.navigate(['map']);
60   setTimeout(() => {
61     this.mapService.showKabinet(kabinet);
62   }, 500);
63 }
64 }

```

Slika 5.11. Drugi dio datoteke table.component.ts

```

1 <div class="header" *ngIf="displayCSV">
2   <input
3     #fileUploadSimple
4     [accept]="'.csv'"
5     type="file"
6     (change)="onFileSelected($event)"
7     hidden="true"
8   />
9   <button mat-raised-button color="primary" (click)="fileUploadSimple.click()">
10     Uvoz iz CSV
11   </button>
12 </div>
13
14 <table mat-table [dataSource]="dataSource" class="mat-elevation-z8">
15   <ng-container matColumnDef="name">
16     <th mat-header-cell *matHeaderCellDef>Ime Logopedskog Kabineta</th>
17     <td mat-cell *matCellDef="let element">{{ element.name }}</td>
18   </ng-container>
19
20   <ng-container matColumnDef="address">
21     <th mat-header-cell *matHeaderCellDef>Adresa</th>
22     <td mat-cell *matCellDef="let element">{{ element.address }}</td>
23   </ng-container>
24
25   <ng-container matColumnDef="website">
26     <th mat-header-cell *matHeaderCellDef>Web-stranica</th>
27     <td mat-cell *matCellDef="let element">{{ element.website }}</td>
28   </ng-container>
29
30   <ng-container matColumnDef="phone_number">
31     <th mat-header-cell *matHeaderCellDef>Kontakt broj</th>
32     <td mat-cell *matCellDef="let element">{{ element.phone_number }}</td>
33   </ng-container>
34
35   <tr mat-header-row *matHeaderRowDef="displayedColumns"></tr>
36   <tr
37     mat-row
38     *matRowDef="let row; columns: displayedColumns"
39     (click)="showKabinet(row)"
40   ></tr>
41 </table>

```

Slika 5.12. Datoteka table.component.html

```

1  .header {
2    margin: 20px;
3    display: flex;
4    justify-content: center;
5    align-items: center;
6    gap: 10px;
7  }
8
9  mat-header-cell,
10 th {
11   background-color: #2e7d32 !important;
12   color: whitesmoke !important;
13   font-weight: bold;
14 }
15
16 tr:hover {
17   background-color: #2e7d323a;
18   transition: all 0.4s;
19   cursor: pointer;
20 }
21

```

Slika 5.13. Datoteka table.component.scss

```

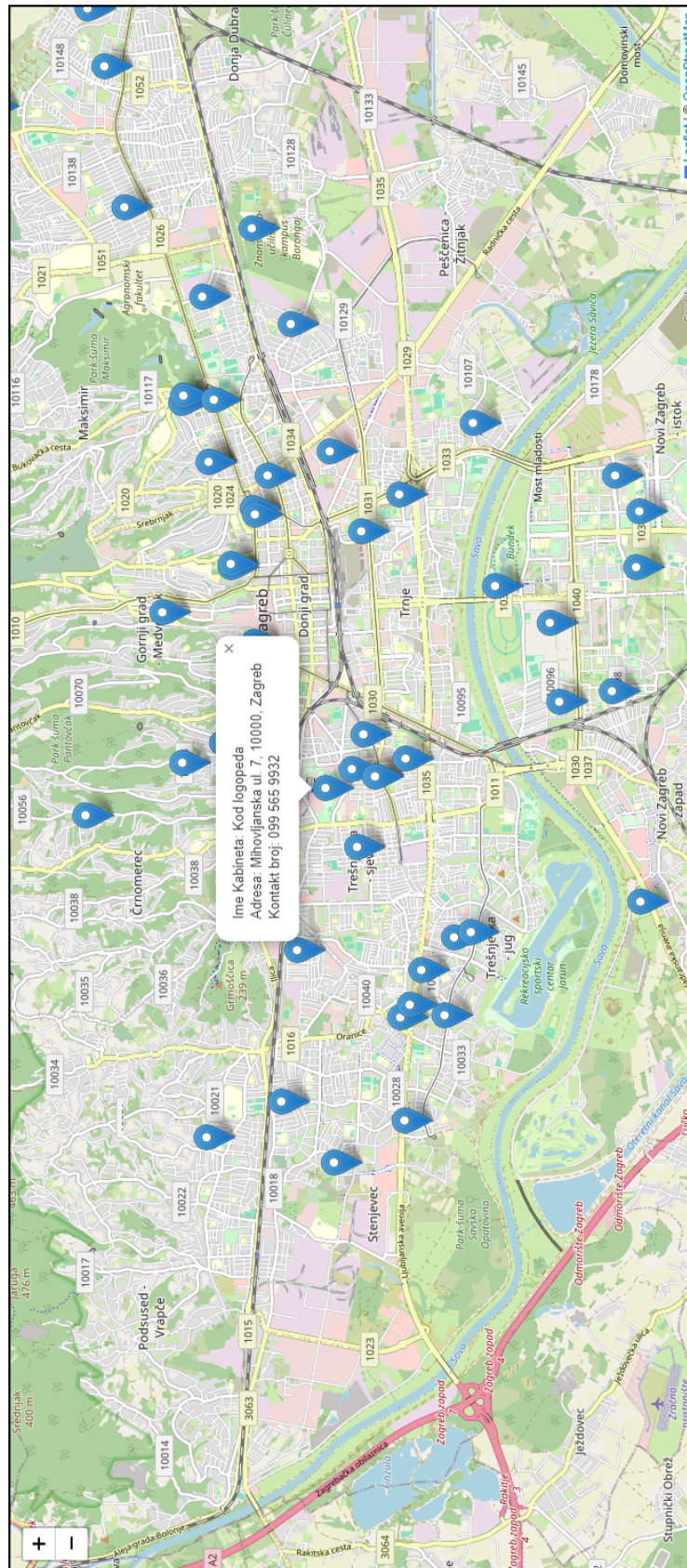
1  import { Injectable } from '@angular/core';
2  import { Subject } from 'rxjs';
3  import { Kabinet } from '../model/kabinet';
4
5  @Injectable({
6    providedIn: 'root',
7  })
8  export class MapService {
9    kabinetSub = new Subject<Kabinet>();
10
11    showKabinet(kabinet: Kabinet) {
12      this.kabinetSub.next(kabinet);
13    }
14  }

```

Slika 5.15. Datoteka map.service.ts

 Logomap 		Adresa	Web-stranica	Kontakt broj
Ime Logopedskog Kabineta				
Logovjezbaonica - logopedski kabinet, Logoped Zagreb	Uli. Damira Tomljanovića 9, 10020, Zagreb	logovjezbaonica.hr	099 209 8333	
Logopedski kabinet Rafael	Lastovska ul. 2A, 10000, Zagreb	logoped-rafael.hr	099 734 6399	
LogopedAna	Uli. Božidara Magovca 107, 10000, Zagreb		095 362 5750	
Logopedski Centar Govorni Oblačić	Vranovinski Ogranak I 4, 10000, Zagreb	logopedski-centar.com	095 299 0299	
LOGOTHERAPIA D.O.O.	Uli. Marijana Derenčina 27, 10000, Zagreb	logotherapie.hr	01 4618 600	
Jezik i govor - Logopedska dijagnostika i terapija	Vajdin vijenac 2, 10000, Zagreb		01 6600 156	
ProLogo kabinet	Uli. Davora Zbijskog 36, 10000, Zagreb		098 954 8966	
Majić Kristina, logopedski kabinet	Lopatinečka ul. 13, 10000, Zagreb	logopedskikabinet.com	01 3817 428	
Centar za jezik i govor Vrutak	Ulica Miroslava Krajijevića 6, 10000, Zagreb	centar-vrutak.eu	01 2929 140	

Slika 5.14. Dio tablice u sklopu web-stranice



Slika 5.16. Kabinet prikazan na karti nakon odabira u tablici

6. ZAKLJUČAK

Web-stranice s web-kartama predstavljaju ključni alat u suvremenom društvu, pružajući brojne prednosti i široku primjenu u različitim sektorima. Prije svega, omogućuju jednostavan i brz pristup geoinformacijama koje su korisne za svakodnevne aktivnosti. U kontekstu logopedске struke, takva web-stranica omogućuje jednostavno pronalaženje najbližeg logopeda, čime se povećava pristupačnost logopedskih usluga. Svaka točka na karti sadržava dodatne informacije o kabinetu, što pomaže u pronalasku logopedskog kabineta koji najbolje odgovara specifičnim potrebama korisnika.

Jedna od glavnih prednosti integracije web-stranice s bazom podataka je ta da se podaci mogu automatski ažurirati kada dođe do promjena u bazi podataka. Korisnici uvijek imaju pristup najnovijim informacijama bez potrebe za ručnim ažuriranjem. Također, omogućuje prikupljanje i analizu podataka o korisnicima i njihovim interakcijama sa stranicom, što može pomoći u unapređenju usluga.

Korištenje razvojnog okvira poput Angulara također ima brojne prednosti. Jedna od glavnih prednosti je velika i aktivna zajednica, što znači obilje podrške. Primjenom razvojnog okvira Angular moguće je web-stranicu nadograđivati i dodavati više funkcionalnosti. Za potrebe ovog diplomskog rada izrađena je kompaktna web-stranica. Međutim, Angular podržava izradu robusnih web-aplikacija kojima se znatno može povećati funkcionalnost izrađene stranice.

Korištenje tehnologija otvorenog koda omogućuje stvaranje dinamične i pouzdane platforme koja se može kontinuirano nadograđivati sukladno potrebama korisnika. Integracija s bazom podataka osigurava ažurne informacije i jednostavno upravljanje podacima, dok korištenje modernih okvira poput Angulara pruža izvrsno korisničko iskustvo. Korištenjem otvorenog koda web-stranica je fleksibilna za buduće nadogradnje, čime se omogućuje dugoročna korist za logopedске stručnjake i njihove korisnike.

LITERATURA

- American Speech-Language-Hearing Association (2016): Scope of Practice in Speech-Language Pathology [Scope of Practice], <https://www.asha.org/siteassets/publications/sp2016-00343.pdf> , (15.5.2024.)
- Frančula, N. (2004): Digitalna kartografija, 3. prošireno izdanje, Sveučilište u Zagrebu – Geodetski Fakultet, Zagreb.
- Miler, M. (2021): Baze prostornih podataka – Vježbe 1. dio, Sveučilište u Zagrebu – Geodetski Fakultet, Zagreb.
- Royal College of Speech & Language Therapists (n.d.): What is speech and language therapy? <https://www.rcslt.org/wp-content/uploads/media/Project/RCSLT/rcslt-what-is-slt-factsheet.pdf>, (15.5.2024.)
- Župan, R., Frangeš, S. (2015): Web-cartography, University of Zagreb – Faculty of Geodesy, Zagreb.

POPIS URL-ova

- URL 1. <https://vijesti.hrt.hr/hrvatska/u-hrvatskoj-vec-godinama-nema-dovoljno-logopeda-10871462> (8.6.2024.)
- URL 2. <https://www.erf.unizg.hr/> (15.5.2024.)
- URL 3. <https://health.u ct.ac.za/departement-dhrs/divisions-communication-sciences-disorders-about-communication-sciences-and-disorders/what-speech-language-pathologist> (15.5.2024.)
- URL 4. https://wiki.openstreetmap.org/wiki/Tag:healthcare%3Dspeech_therapist (23.3.2024.)
- URL 5. <https://www.techopedia.com/definition/5212/web-scraping> (24.3.2024.)
- URL 6. <https://blog.apify.com/is-web-scraping-legal/> (24.3.2024.)
- URL 7. <https://home.cern/science/com-puting/birth-web/short-history-web> (12.5.2024.)
- URL 8. <https://forrest.nyc/a-brief-history-of-web-maps/> (15.5.2024.)
- URL 9. <https://www.e-education.psu.edu/geog585/node/643> (15.5.2024.)
- URL 10. <https://www.typescriptlang.org/why-create-typescript/> (9.5.2024.)
- URL 11. <https://developer.mozilla.org/en-US/docs/Web/HTML> (9.5.2024.)
- URL 12. <https://developer.mozilla.org/en-US/docs/Web/CSS> (9.5.2024.)
- URL 13. <https://angular.io/guide/what-is-angular> (9.5.2024.)
- URL 14. <https://rxjs.dev/guide/overview> (17.5.2024.)
- URL 15. <https://leafletjs.com/> (10.5.2024.)
- URL 16. <https://supabase.com/> (4.4.2024.)
- URL 17. <https://www.postgresql.org/about/> (4.4.2024.)
- URL 18. <https://postgis.net/> (4.4.2024.)

- URL 19. <https://www.datacamp.com/blog/what-is-python-used-for> (10.5.2024.)
- URL 20. <https://playwright.dev/python/docs/api/class-playwright> (24.3.2024.)
- URL 21. <https://docs.python.org/3/library/dataclasses.html> (24.3.2024.)
- URL 22. <https://github.com/agunje123/logomap-diplomski> (17.5.2024.)
- URL 23. <https://github.com/agunje123/logosrape-diplomski> (17.5.2024.)

POPIS TABLICA

Tablica 4.1. Primjer podataka o logopedskim kabinetima dobiveni iz skripte za web-struganje	19
---	----

POPIS SLIKA

Slika 3.1. Vizualizacija dohvaćanja karte iz servera (URL 9).....	5
Slika 4.1. Uvoz biblioteka u skripti za web-struganje	13
Slika 4.3. Koordinate unutar URL-a.....	13
Slika 4.2. Pomoćne klase i funkcije prije web-struganja.....	14
Slika 4.4. Funkcija main web-strugača.....	15
Slika 4.5. Prihvatanje uvjeta poslovanja prije ulaska na Google karte (Google Maps)	15
Slika 4.6. Dio skripte za listanje po kabinetima	16
Slika 4.7. Lista kabineta koja se otvara u virtualnom web-pregledniku s virtualnim mišem...	16
Slika 4.8. Dio Google karata odakle web-strugač preuzima podatke.....	17
Slika 4.9. Dohvaćanje podataka za svaki kabinet, spremanje u listu i u CSV datoteku.....	18
Slika 4.10. Početne postavke za bazu podataka.....	20
Slika 4.11. Bočna navigacijska traka.....	21
Slika 4.12. Instalirano PostGIS proširenje	22
Slika 4.13. Naredba za kreiranje nove tablice	22
Slika 4.14. Naredba za stvaranje nove funkcije.....	23
Slika 4.15. Naredba za stvaranje nove Angular aplikacije	23
Slika 4.16. Generirana struktura Angular aplikacije	24
Slika 4.17. Naredba za stvaranje servisa	25
Slika 4.18. Poveznica i ključ za bazu podataka	25
Slika 4.19. Funkcija pushKabinetList	26
Slika 4.20. Funkcija getKabinetList	26
Slika 4.21. Naredba za stvaranje komponente.....	27
Slika 4.22. Funkcija parseCsvRow	27
Slika 4.23. Funkcija parseCsv	28
Slika 4.24. Tablica s podacima o logopedskim kabinetima	29
Slika 4.25. Pomoćne varijable za izradu karte.....	30

Slika 4.26. Funkcija createPopup	31
Slika 4.27. Funkcija createMarkerWithPopup	31
Slika 4.28. Primjer znaka kabineta sa svim dostupnim podacima.....	31
Slika 4.29. Funkcija initMap	32
Slika 4.30. Funkcija ngOnInit	33
Slika 4.31. Karta logopedskih kabineta	34
Slika 5.1. Prvi dio skripte za web-struganje	37
Slika 5.2. Drugi dio skripte za web-struganje	38
Slika 5.3. Dio tablice unutar baze podataka	39
Slika 5.4. Datoteka supabase.service.ts	40
Slika 5.5. Prvi dio datoteke map.component.ts	41
Slika 5.6. Drugi dio datoteke map.component.ts.....	42
Slika 5.7. Datoteka map.component.html	43
Slika 5.8. Datoteka map.component.scss	43
Slika 5.9. Karta u sklopu web-stranice	44
Slika 5.10. Prvi dio datoteke table.component.ts	45
Slika 5.11. Drugi dio datoteke table.component.ts.....	46
Slika 5.12. Datoteka table.component.html.....	47
Slika 5.13. Datoteka table.component.scss	48
Slika 5.15. Datoteka map.service.ts.....	48
Slika 5.14. Dio tablice u sklopu web-stranice	49
Slika 5.16. Kabinet prikazan na karti nakon odabira u tablici.....	50