Izrada informacijskog sustava za zemljišne knjige i katastar

Sršen, Pero

SVEUČILIŠTE U DUBROVNIKU

ODJEL ZA ELEKTROTEHNIKU I RAČUNARSTVO

PERO SRŠEN

IZRADA INFORMACIJSKOG SUSTAVA ZA ZEMLJIŠNE KNJIGE I KATASTAR

ZAVRŠNI RAD

Dubrovnik, srpanj, 2023.

SVEUČILIŠTE U DUBROVNIKU

ODJEL ZA ELEKTROTEHNIKU I RAČUNARSTVO

IZRADA INFORMACIJSKOG SUSTAVA ZA ZEMLJIŠNE KNJIGE I KATASTAR

ZAVRŠNI RAD

Studij: Primijenjeno/poslovno računarstvo

Kolegij: Analiza i projektiranje računalom

Mentor: doc. dr. sc. Ivona Zakarija

Komentor: Ivan Grbavac, dipl. Ing.

Student: Pero Sršen

Dubrovnik, srpanj, 2023.

SAŽETAK

Prilikom izrade informacijskih sustava u oblaku programeri ponekad koriste nativni pristup izrade. Takvim načinom izrade ne koriste se nikakva razvojna okruženja ili programske biblioteke, već samo osnovni programski jezici potrebni za izradu aplikacije poput PHP-a i JavaScript-a. Kod ovakvog pristupa izrade sustava stvara se problem ne modularnosti odnosno ponovne iskoristivosti koda. U današnje doba, kada su vrijeme razvoja i ponovna iskoristivost koda iznimno važni parametri izrade takav način nije preporučljiv. Upotrebom različitih biblioteka i razvojnih okruženja mrežni sustavi postaju kompliciraniji, ali se skraćuje vrijeme izrade uz enkapsulaciju pojedinih aspekata.

U ovom završnom radu napravljena je analiza koncepta, jezika, razvojnih okruženja i alata koji mogu biti korišteni prilikom izrade informacijskog sustava u oblaku. Analizirani su i zahtjevi informacijskog sustava za zemljišne knjige i katastar. Sintezom opisanih tehnologija, a na osnovu prikupljenih zahtjeva, razvijen je i opisan informacijski sustav za zemljišne knjige i katastar.

Ključne riječi: mrežna aplikacija, informacijski sustav za katastar, zemljišne knjige

ABSTRACT

When developing cloud-based information systems, developers sometimes use a native development approach. This approach does not involve the use of development environments or software libraries, but only basic programming languages such as PHP and JavaScript necessary for application development. However, this method creates a problem of lack of modularity and code reusability. In today's age, where development time and code reusability are crucial parameters, this approach is not recommended. By utilizing different libraries and development environments, web systems become more complex but reduce development time by encapsulating specific aspects.

This final thesis conducted an analysis of concepts, languages, development environments, and tools that can be used in the development of cloud-based information systems. The requirements of the land registry and cadastral information system were also analyzed. Through a synthesis of the described technologies and based on the gathered requirements, an information system for land registry and cadastral purposes was developed and described.

Keywords: web application, land registry information system, cadastral records

SADRŽAJ

1 UVOD

Katastar je službena evidencija nekretnina koja se izrađuje na temelju geodetskog premjera zemljišta i njegova razvrstavanja po namjeni i kakvoći. Svaka čestica zemljišta svrstava se prema opisu i načinu iskorištavanja, određuje im se razred te se na osnovu tih karakteristika utvrđuje zemljišni porez. Katastarski izvadak sadržava podatke o položaju, obliku, razredu i namjeni određenih čestica zemljišta, o vlasniku i svrsi korištenja. Na temelju katastarske evidencije razvija se zemljišna knjiga kao registar prava. To su javne knjige u kojima se bilježe podaci o pravnom stanju nekretnina – vlasništvu, koje je bitno za bilo kakav pravni promet istih. [1]

Prvi potpuni premjer i evidencija parcela na teritoriju današnje Republike Hrvatske i Federacije Bosne i Hercegovine izrađen je za vrijeme Austrougarske vlasti. Na tim prostorima uspostavlja se katastar, a potom zemljišne knjige, koji se s vremenom unaprjeđuju na dobrobit svih građana i institucija kojima služe.

Katastarsku evidenciju vode područni uredi, katastri, a za vođenje zemljišnih knjiga nadležni su zemljišnoknjižni odjeli općinskih sudova. Svi poslovi i postupci koji se odrađuju u katastru i zemljišnim knjigama propisani su zakonima. [2][3]

Od 90-ih godina do danas Republika Hrvatska razvija informacijski sustav katastra i zemljišnih knjiga, te je i dalje usmjerena ka naprednijim tehnologijama financirana zajmom Međunarodne banke za obnovu i razvoj. [4]

Uspostavljanjem službenog informacijskog sustava evidencije nekretnina Federacija Bosna i Hercegovina nastoji biti u korak s Republikom Hrvatskom i drugim europskim zemljama. Neke od općina, u ovom slučaju općina Neum, rade na usklađivanju podataka katastra i zemljišnih knjiga, a time im se ujedno otvara i mogućnost objedinjavanja katastarskih i zemljišnih podataka u jedinstvenu evidenciju, bazu podataka.

Digitalizacijom službenog sustava evidencije nekretnina katastra i zemljišnih knjiga dobivaju se kvalitetnije usluge. Poslovni procesi su brži i jednostavniji, a korisnicima digitalnih usluga s ovoga informacijskog sustava omogućava se jednostavni pregled podataka, podnošenje zahtjeva i ishođenje potrebne dokumentacije. [4]

Osim same modernizacije procesa, prethodno navedeni razlozi su dodatni motiv za izradu i implementaciju informacijskog sustava evidencije nekretnina katastra i zemljišnih knjiga za Bosnu i Hercegovinu, odnosno Općinu Neum koji je opisan u ovom radu..

Drugo poglavlje rada bavi se analizom postojećega sustava u Republici Hrvatskoj koji bi se mogao implementirati u Bosni i Hercegovini. Treće poglavlje sadržava tablice sa zahtjevima koje sustav treba imati. Četvrto poglavlje opisiva tehnologije korištene tijekom izrade sustava i opis baze podataka. Peto poglavlje opisiva izgled i funkcionalnost mrežne aplikacije.

2 POSTOJEĆI SUSTAVI

Hrvatski sustav registriranja nekretnina i prava na njima temelji se na dva registra – katastru i zemljišnim knjigama. U katastru se nekretnine opisuju prema njihovim tehničkim karakteristikama. Katastarski podaci o česticama podloga su za osnivanje, vođenje i održavanje zemljišnih knjiga. U zemljišnim knjigama se podacima o katastarskim česticama pridružuju podaci o nositeljima prava na njima. [4]

Dvojna evidencija, evidencija katastra i evidencija zemljišnih knjiga je komplicirana i teško održiva, ali s pojavom novih tehnologija dolazi do efikasnijih rješenja koja osiguravaju pravnu sigurnost vlasnika i brže prikupljanje informacija.

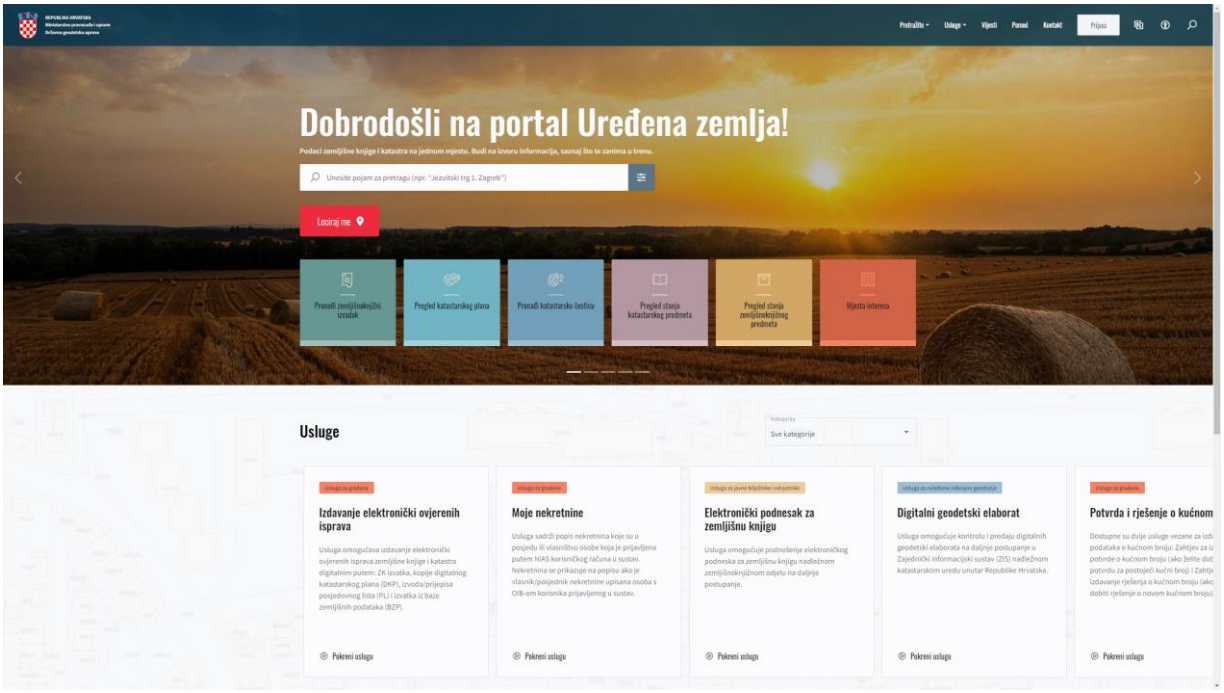
Republika Hrvatska je 2003. godine pokrenula Nacionalni program sređivanja zemljišnih knjiga i katastra, pod zajedničkim nazivom Uređena zemlja. Uspostavljanjem ovoga sustava ubrzana je registracija nekretnina u katastarskom i zemljišnoknjižnom sustavu te je time povećana razina sigurnosti u pravnom prometu nekretnina i zaštite prava. Oba sustava su racionalizirana, poslovni procesi su pojednostavljeni, a uvođenjem elektroničkih sustava poboljšan je odnos s korisnicima što rezultira kvalitetom pružanja usluga. Naprednijim sustavom katastra i zemljišnih knjiga omogućen je brži i jednostavniji postupak registriranja nekretnina i prava na njima, a sve informacije dostupne su elektroničkim putem.

U okviru nacionalnog programa Uređena zemlja razvijen je i Zajednički informacijski Sustav zemljišnih knjiga i katastra (*ZiS*). Realizacijom ZiS-a stvoren je jedinstven registar katastra i zemljišnih knjiga, koji povezuje i razmjenjuje podatke o nekretninama. Jednostavnije rečeno, uspostavljena je zajednička baza podataka i aplikacija za vođenje i održavanje katastarskih i zemljišnoknjižnih podataka, što donosi brojne koristi korisnicima. Osim što je vrijeme potrebno za pristup podacima i provođenje upisa značajno skraćeno, građani sada mogu na jednom mjestu vidjeti vlasničku strukturu nekretnina, njihovu lokaciju i druge funkcionalnosti.

Dio ZIS-a je "One-Stop-Shop" (OSS) - jedinstveno mjesto za pristup podacima zemljišne knjige i katastra. OSS se sastoji od dvije komponente:

- Javni OSS - dostupan svim korisnicima bez registracije, omogućuje pretragu i pregled osnovnih podataka zemljišne knjige i katastra.

- Privatni OSS - dostupan samo registriranim korisnicima, omogućuje pregled podataka, podnošenje zahtjeva za izdavanje javnih isprava te rješavanje poslova u zemljišnoknjižnim odjelima i katastarskim uredima, kao i primanje službenih dokumenata.



Slika 1 Prikaz početne stranice portala Uređena zemlja s uslugama [5]

Neke od javnih isprava koje se mogu preuzeti putem OSS-a Zajedničkog informacijskog sustava katastra i zemljišnih knjiga su:

- kopija katastarskog plana,
- prijepis/izvod iz posjedovnog lista,
- izvod iz baze zemljišnih podataka,
- razna uvjerenja i potvrde,
- zemljišnoknjižni izvadak.

NESLUŽBENA KOPIJA

REPUBLIKA HRVATSKA
DRŽAVNA GEODETSKA UPRAVA
PODRUČNI URED ZA KATASTAR
DUBROVNIK

Stanje na dan: 27.06.2023. 11:19

PRIJEPIS POSJEDOVNOG LISTA

Katastarska općina: BABINO POLJE (Mbr. 306100)
Posjedovni list: 416

Udio	Prezime i ime odnosno tvrtka ili naziv, prebivalište odnosno sjedište upisane osobe	OIB
1/1	SRŠEN MILJENKO, P. GRGURA, SRŠENOVIĆI 5, BABINO POLJE, HRVATSKA (VLASNIK)	02948003224

Podaci o katastarskim česticama

Zgr	Dio	Broj katastarske čestice	Adresa katastarske čestice/Način uporabe katastarske čestice/Način uporabe zgrade, naziv zgrade, kućni broj zgrade	Površina/ m2	Broj D.L.	Posebni pravni režimi	Primjedba
*		**191**	**SRŠENOVIĆI**	**58**	**11**		
			KUĆA, SRŠENOVIĆI	58			
			Ukupna površina katastarskih čestica	**58**			

NAPOMENA: Ovaj prijepis posjedovnog lista nije dokaz o vlasništvu na katastarskim česticama upisanim u posjedovnom listu.

Slika 2 Neslužbena kopija prijepisa posjedovnog lista u Hrvatskoj [6]

3 ZAHTJEVI PROGRAMA

U zahtjevima projekta navedene su i opisane razne vrste zahtjeva koje je, prema prioritetu, neophodno integrirati u mrežnu aplikaciju. Zahtjeve koji se implementiraju dijele se na funkcionalne i nefunkcionalne zahtjeve i zahtjeve za sučelje.

3.1 Funkcionalni zahtjevi

Funkcionalni zahtjevi su zahtjevi koji opisuju funkcionalne elemente na mrežnoj aplikaciji.

Tablica 1 Funkcionalni zahtjevi

ID zahtjeva	Naziv funkcionalnog zahtjeva	Prioritet
Opis funkcionalnog zahtjeva		
F01	Autentikacijski sustav za administriranje aplikacije (*eng*. *Content Managment System*)	1
Omogućava autoriziranom korisniku pristup u sustav za administriranje aplikacije. Pristup se omogućava unosom traženih vjerodajnica.		
F02	Prikaz usluga	1
Ulaskom u sustav korisniku se, na početnoj stranici, prikazuju postojeći podaci i mogućnosti odabira ostalih usluga unutar sustava.		
F04	Dodavanje nove čestice	1
Unutar mrežne aplikacije, odabirom usluge *Čestice*, korisniku se nudi opcija *Dodaj* koja služi za dodavanje nove čestice u sustav.		
F05	Izmjena podataka čestice	1

Unutar mrežne aplikacije, odabirom usluge *Čestice*, korisniku se odabirom tražene čestice nudi opcija *Izmjena* koja služi za izmjenu podataka čestice u sustavu.		
F06	Brisanje čestica	1
Unutar mrežne aplikacije, odabirom usluge *Čestice*, korisniku se odabirom tražene čestice nudi opcija *Brisanje* koja služi za izmjenu podataka čestice u sustavu.		
F07	Dodavanje novog posjedovnog lista	1
Unutar mrežne aplikacije, odabirom usluge *Posjedovni listovi*, korisniku se nudi opcija *Dodaj* koja služi za dodavanje novog posjedovnog lista u sustav.		
F08	Izmjena posjedovnog lista	1
Unutar mrežne aplikacije, odabirom usluge *Posjedovni listovi*, korisniku se odabirom traženog posjedovnog lista nudi opcija *Izmjena* koja služi za izmjenu podataka posjedovnog lista u sustavu.		
F09	Brisanje posjedovnog lista	1
Unutar mrežne aplikacije, odabirom usluge *Posjedovni listovi*, korisniku se odabirom traženog posjedovnog lista nudi opcija *Brisanje* koja služi za brisanje podataka posjedovnog lista iz sustava.		
F10	Slanje elektroničke obavijesti nakon odabranih akcija	2
Mrežna aplikacija šalje obavijest elektroničkom poštom o napravljenim akcijama svakoj upisanoj osobi u posjedovnom listu.		
F11	Ispis odabranih podataka	2
Unutar mrežne aplikacije, unutar odabrane usluge, korisniku se nudi opcija *Ispis* koja ispisiva odabrane stavke.		

F12	Dnevnik izmjena	3
Mrežna aplikacija sprema podatke svake promjene napravljene unutar sustava u dnevnik promjena u *.txt* formatu.		

3.2 Zahtjevi za strojna sučelja

Zahtjevi za strojna sučelja opisuju specifikacije i funkcionalnosti koje ta sučelja trebaju podržavati kako bi se omogućila ispravna i učinkovita komunikacija s vanjskim uređajima.

Tablica 2 Zahtjevi za strojna sučelja

ID zahtjeva	Naziv zahtjeva	Prioritet
Opis zahtjeva		
S01	Prilagodba mrežne aplikacije rezoluciji uređaja	2
Mrežna aplikacija se prilagođava rezoluciji uređaja na kojemu je pokrenuta, poželjno je da uređaj bude računalo.		

3.3 Nefunkcionalni zahtjevi

Nefunkcionalni zahtjevi odnose se na zahtjeve koji opisuju karakteristike sustava ili projekta koje nisu izravno povezane s njegovim funkcionalnostima, već utječu na njegovu kvalitetu, performanse ili upotrebljivost. Ovi zahtjevi često opisuju ograničenja, standarde ili ciljeve koje sustav ili projekt moraju ispuniti.

Tablica 3 Iskoristivost

ID zahtjeva	Naziv zahtjeva	Prioritet
Opis zahtjeva		
I01	Mrežna aplikacija katastarskog sustava, izgled	2
Mrežna aplikacija treba biti jednostavnog i izravnog izgleda, korisniku je sve pregledno i bez ikakvih mogućih smetnji.		

4 KORIŠTENE TEHNOLOGIJE I ARHITEKTURA

4.1 HTML

HTML (*eng. Hyper-Text Markup Language*) je standardizirani jezik koji se koristi za izradu mrežnih stranica. Tvorci HTML su Tim Berners-Lee i njegov tim na CERN-u.. Najnovija verzija, HTML5, objavljena je 2008. godine. Ovaj jezik igra ključnu ulogu u izradi mrežne stranica jer omogućava programerima da strukturiraju i organiziraju sadržaj na mreži. Zahvaljujući poboljšanjima u vizualnim stilovima (*CSS*) i objektnim jezicima (*JavaScript*), HTML je postao snažan alat za stvaranje interaktivnih i atraktivnih mrežnih stranica.

Princip HTML-a leži u korištenju elemenata za opisivanje dokumenata i podataka koje oni sadrže. Za razliku od većine programskih jezika, HTML se fokusira na prikazivanje podataka, umjesto na njihovu obradu. Osnovne funkcionalnosti HTML-a uključuju definiranje položaja, boje i fontova teksta, postavljanje veličina slika te povezivanje s drugim dokumentima.

Najčešće korištene komponente HTML-a su "oznake". Ove unaprijed definirane funkcije imaju određena imena i služe za obavljanje operacija s podacima. Oznake se često pišu u parovima, gdje prvi pojavljivanje označava početak naredbe, a drugo pojavljivanje označava kraj. Primjeri takvih oznaka uključuju "body" i "paragraph". Osim toga, oznake često mogu imati i neobavezne parametre poput "style", "title" i "height" itd. Također postoje i ne uparene oznake poput "img" i "br". Svaki HTML dokument započinje oznakom "<html>" na početku i završava oznakom "</html>" na kraju.

Iako HTML sam po sebi može stvoriti funkcionalne mrežne stranice i organizirati sadržaj, on se često kombinira s drugim programskim jezicima koji pružaju naprednije mogućnosti vizualnog prikaza podataka i obavljanja složenih operacija. Unatoč svojoj jednostavnosti, HTML ostaje temelj svake mrežne stranice, bilo da se koristi zajedno s CSS-om, PHP-om, JavaScriptom ili drugim tehnologijama. Čak i moderni alati koji se koriste za izradu mrežnih stranica poput WordPress-a i Wix-a koriste HTML kao temelj, iako je sam HTML često skriven iza intuitivnog sučelja i predložaka.

Ukratko, HTML je jezik koji omogućuje programerima da strukturiraju i organiziraju sadržaj na mrežnim stranicama. Kombinacijom HTML-a s drugim tehnologijama, moguće je stvoriti atraktivne, interaktivne i funkcionalne mrežne stranice koje oblikuju online iskustvo korisnika.

U primjeru koda broj 1 prikazuje se kako izgleda struktura HTML koda. Na početku koda, uvijek je važno deklarirati vrstu dokumenta kao HTML kako bi program mogao prepoznati kod koji se piše. Nakon deklaracije, može se započeti pisanje samog koda dodavanjem različitih komponenti u obliku "oznaka" koje će biti prikazane na mrežnoj stranici.

```
<!DOCTYPE html>
<html>
<body>
<h1>Moj prvi naslov</h1>
<p>Moj prvi paragraf.</p>
</body>
</html>
```

Primjer koda 1 Struktura HTML koda.

- Kao što je spomenuto ranije, HTML sadrži mnogo "oznaka" koje se koriste za izradu mrežnih stranica. Te "oznake" se razvrstavaju u različite kategorije sadržaja, uključujući kategoriju tijeka, zaglavlja, grupiranja, interaktivnih, ugrađenih, fraziranja i kategoriju meta podataka. Važno je napomenuti da su sve ove kategorije pod skupovi kategorije tijeka. fraziranja, elemente ugrađivanja, interaktivne elemente i elemente vezane za forme. Primjeri takvih elemenata su sidro (<a>), podebljano (<b>) i diobeni blok ( ). [7]

- Kategorija grupiranja je podskup kategorije tijeka i može se koristiti bilo gdje se očekuje sadržaj tijeka. Elementi koji pripadaju ovoj kategoriji stvaraju sekciju unutar trenutne strukture, koja definira opseg elemenata zaglavlja (<header>), elemenata podnožja (<footer>) i sadržaja naslova. Primjeri elemenata koji

pripadaju ovoj kategoriji su članak (<article>), sa strane (<aside>), navigacija (<nav>) i sekcija (<section>). [7]

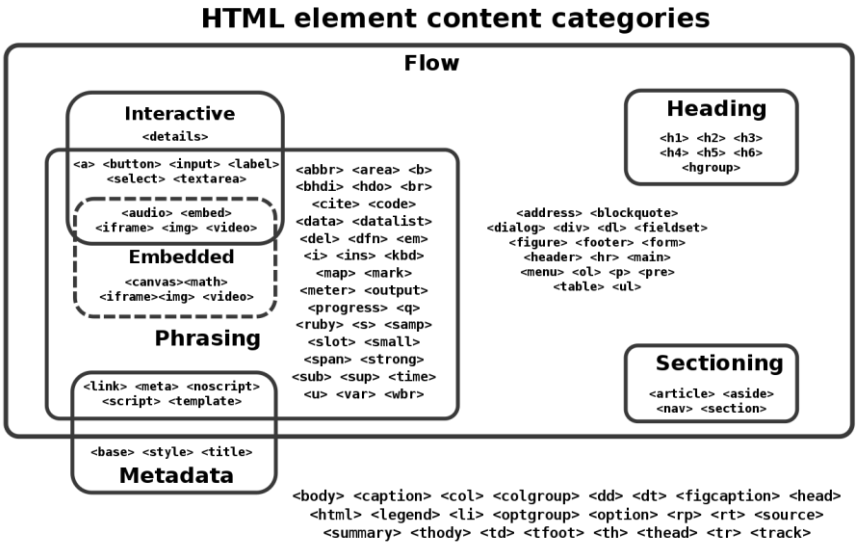
- Kategorija zaglavlja definira naslov sekcije, bilo da je označena eksplicitnim sadržajem grupiranja ili implicitno definiran samim sadržajem zaglavlja. Sadržaj zaglavlja može se koristiti bilo gdje se očekuje sadržaj tijeka. Primjeri elemenata koji pripadaju ovoj kategoriji su zaglavlje (<h1>) i grupa zaglavlja (<hgroup>). [7]

- Sadržaj ugrađivanja omogućuje uvoz drugih resursa ili umetanje sadržaja iz drugih označnih (*eng. markup*) jezika u dokument. Može se koristiti bilo gdje se očekuje sadržaj tijeka. Primjeri elemenata koji pripadaju ovoj kategoriji su zvuk (<audio>), video (<video>), slika (<picture>). [7]

- Sadržaj fraziranja se odnosi na tekst i označni dio koji se nalazi unutar elemenata. Može se koristiti bilo gdje se očekuje sadržaj tijeka. Nizovi fraziranog sadržaja čine odlomke. Primjeri elemenata koji pripadaju ovoj kategoriji su tipka (<button>), novi red (
), bold (<b>), slika (<img>). [7]

- Interaktivni sadržaj uključuje elemente koji su posebno dizajnirani za korisničku interakciju. Može se koristiti bilo gdje se očekuje sadržaj tijeka. Primjeri elemenata koji pripadaju ovoj kategoriji su sidro (<a>), tipka (<button>), oznaka (<label>). [7]

- Sadržaj meta podataka sadrži elemente koji modificiraju prezentaciju ili ponašanje dokumenta, postavljaju linkove i druge dokumente ili prenose druge vrste informacija. Primjeri elemenata koji pripadaju ovoj kategoriji su link (<link>), skripta (<script>), titula (<title>). [7]



Slika 3 Izgled sadržaja kategorija elemenata HTML [8]

Na slici 1. prikazan je prikaz organizacije kategorija HTML-a i veze između njih, odnosno kako su "oznake" međusobno povezane.

4.2 CSS

CSS (eng. *Cascading Style Sheets*) je jezik za stiliziranje koji značajno nadopunjuje HTML. Glavna mu je svrha pomoći u vizualno atraktivnom prikazu sadržaja mrežne stranice. U osnovi, djeluje kao jezik za oblikovanje HTML-a putem vlastitih "oznaka" i atributa.

Razvijen je zajedno s HTML-om 1996. godine, od strane Håkon Wium Liea, kao odgovor na potrebu za dosljednim stiliziranjem mrežnih stranica i omogućavanjem različitih stilova istog HTML dokumenta bez izmjene kôda za preglednike, ispis ili slanje poštom.

Unatoč tome što CSS koristi vlastite stilističke datoteke, potpuno je ovisan o HTML-u i ne može postojati samostalno bez povezanog HTML dokumenta. Također koristi sličan sustav "oznaka".

U biti, CSS koristi datoteku (ili kod napisan izravno u HTML dokumentu) u kojoj se definiraju stilovi za pojedinačne oznake, određene dijelove ili odjeljke HTML dokumenta. Ti stilovi mogu biti naslijeđeni od strane drugih oznaka i dati im određeni prioritet. To je najveća prednost CSS-a jer omogućuje stiliziranje dijelova dokumenta zajedno, bez potrebe za pojedinačnim izmjenama. Na primjer, boja, font ili veličina svih naslova unutar dokumenta mogu se promijeniti unutar CSS dokumenta za oznaku `<h1>`, bez potrebe za mijenjanjem sadržaja HTML dokumenta na mjestima gdje se pojavljuje pojedinačna oznaka `<h1>`. [9]

```
h1 {
color: #252525;
font-size: 3em;
line-height: 50px;
}
<h1>Ovo je naslov</h1>
```

Primjer koda 2 Struktura koda CSS-a.

Primjer koda 2. prikazuje izgled koda u CSS-u. Prije korištenja CSS-a potrebno je stvoriti datoteku s nastavkom .css u kojem će se upisati svi potrebni parametri za oznaku koju želimo preurediti. U primjeru je prikazana linija koda koja se odnosi na CSS stilove za oznaku <h1>. Konkretno, definira se stil za naslov prvog reda tako što boju teksta mijenja na tamno sivo, određuje veličinu fonta naslova na 3 puta veličinu osnovne veličine fonta i definira visinu linije naslova na 50 piksela, što utječe na razmak između redova teksta.

4.3 JavaScript

JavaScript (JS) je najpopularniji jezik za programiranje širom svijeta, često korišten pri izradi mrežnih stranica, a njegovo učenje je jednostavno. Koristi se "u hodu" (tijekom izvršavanja) prilikom prijevoda koda. Iako je najpoznatiji kao skriptni jezik za mrežne stranice, on se također koristi u mnogim drugim okruženjima kao što su Node.js, Apache i Adobe Acrobat. JavaScript je nastao 1995. godine, a njegov tvorac je Brendan Eich. Ovaj jezik se grana iz programiranja temeljenog na prototipu i podržava više paradigmi. Radi na jednoj niti i karakterizira ga dinamičnost. JavaScript podržava objektno orijentirano, imperativno i deklarativno programiranje. Prilikom stvaranja mrežnih stranica, JavaScript ima iznimnu važnost jer određuje ponašanje sadržaja na mrežnoj stranici. Osim toga, u ovoj implementaciji su korištene različite JavaScript knjižnice kao što su Bootstrap, JQuery, JSON, Ajax, Axios i Vuetify, koje pružaju dodatne funkcionalnosti i olakšavaju razvoj mrežnih aplikacija.

```
<!DOCTYPE html>
<html>
<body>
<h2>Moj prvi JavaScript</h2>
<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Prikaz vremena i datuma.</button>
<p id="demo"></p>
</body>
</html>
```

Primjer koda 3 Struktura koda JavaScript-a.

U primjeru koda 3. imamo jednostavnu JavaScript skriptu koja stvara gumb na mrežnoj stranici te klikom na njega pokreće funkciju prikaza datuma i vremena.

4.3.1 Axios

Axios je popularna JavaScript biblioteka koja olakšava slanje HTTP zahtjeva s klijenta na poslužitelj. Služi kao moderna alternativa XMLHttpRequest-u i nudi jednostavno sučelje za izvođenje HTTP zahtjeva. Axios podržava višeplatformski rad, što znači da može raditi i u preglednicima i na poslužitelju (npr. uz pomoć Node.js). Također podržava funkcionalnosti poput automatskog pretvaranja podataka u JSON format, upravljanja pogreškama i postavljanja zaglavlja zahtjeva. Axios omogućuje slanje različitih vrsta zahtjeva, poput GET, POST, PUT, DELETE itd., i prihvaća razne konfiguracijske opcije za prilagođavanje zahtjeva.

```
const axios = require('axios');

axios.get('http://webcode.me').then(resp => {

    console.log(resp.data);
});
```

Primjer koda 4 Struktura koda Axios-a.

Primjer koda 4. prikazuje kod koji koristi Axios biblioteku za izvođenje HTTP zahtjeva, u ovom slučaju koristi se zahtjev GET koji će pritom ispisati dobivene podatke u konzolu.

4.3.2 Ajax

AJAX je tehnika koja koristi JavaScript za slanje asinkronih HTTP zahtjeva na poslužitelj i primanje podataka u XML formatu. Međutim, danas se umjesto XML-a češće koristi JSON (JavaScript Object Notation). AJAX koristi XMLHttpRequest objekt za komunikaciju s poslužiteljem i osigurava da se mrežna stranica ne blokira tijekom čekanja na odgovor od servera. Ova tehnika omogućuje ažuriranje samo određenih dijelova stranice, poboljšavajući korisničko iskustvo.

```
xhttp.onload = function() {
    document.getElementById("demo").innerHTML =
    this.responseText;
  }
  xhttp.open("GET", "ajax_info.txt");
  xhttp.send();
}
</script>
</body>
</html>
```

Primjer koda 5 Struktura koda Ajax-a.

Primjer koda 5. predstavlja HTML stranicu s integriranom JavaScript skriptom koja koristi gumb za izvršavanje AJAX zahtjeva kako bi se promijenio sadržaj teksta na mrežnoj stranici.

4.3.3 jQuery

jQuery je popularna JavaScript biblioteka koja olakšava manipulaciju HTML dokumentima, obradu događaja, animacija i izvršavanje AJAX zahtjeva na mrežnim

stranicama. Glavna svrha jQuery-a je pojednostaviti pisanje JavaScript koda i olakšati interakciju s elementima na mrežnoj stranici.

Dizajniran je kako bi pojednostavio rad s JavaScript-om, omogućio pisanje manje koda i brži razvoj mrežnih stranica. Iako je jQuery i dalje široko korišten, treba napomenuti da s modernim preglednicima i napretkom JavaScripta, mnoge funkcionalnosti koje je jQuery pružao, postale dostupne izravno kroz JavaScript jezik te i nove biblioteke i okvire.

```
$(document).ready(function(){
  $("p").click(function(){
    $(this).hide();
  });
});
```

Primjer koda 6 Struktura koda jQuery-a.

U primjeru koda 6. pritiskom na učitane paragrafe pokreće se skripta koja skriva kliknuti paragraf.

4.3.4 JSON

JSON (*eng. JavaScript Object Notation*) je popularan format za razmjenu podataka. On se koristi za čitanje i pisanje podataka, a koristi jednostavnu strukturu objekata i nizova. Objekti su kolekcije parova ključ-vrijednost, dok su nizovi uređene liste vrijednosti. JSON podržava osnovne podatkovne tipove poput nizova, brojeva, boolean vrijednosti i NULL vrijednosti. Ovaj format se često koristi za prijenos podataka između mrežnog poslužitelja i klijentskih aplikacija. JSON je popularan zbog svoje jednostavnosti i široke prihvaćenosti, a mnogi programski jezici imaju ugrađene funkcije za rad s njim.

```
'{"name":"John", "age":30, "car":null}'
let personName = obj.name;
let personAge = obj.age;
```

Primjer koda 7 Struktura koda JSON-a.

Primjer koda 7.prikazuje JSON niz koji se dekodira i kojemu se onda pridružuje vrijednost te na kraju pridružuje zadanoj varijabli.

4.3.5 Bootstrap

Bootstrap je popularan razvojni okvir za izradu korisničkog sučelja mrežnih stranica. Temelji se na uzvratnom dizajnu i koristi rešetkasti sustav za jednostavno raspoređivanje elemenata na različitim uređajima. Sadrži mnogo unaprijed definiranih komponenata i stilova koji olakšavaju izgradnju funkcionalnosti i dosljedan izgled. Također uključuje JavaScript komponente za interaktivnost. Bootstrap ima detaljnu dokumentaciju i veliku zajednicu korisnika. Sve ovo ga čini moćnim alatom za brzo stvaranje modernih, mobilno prilagodljivih mrežnih stranica.

```
<div class="col-sm-4">
      <h1>Column 1</h1>
      <p>Lorem ipsum dolor..</p>
</div>
```

Primjer koda 8 Struktura koda Bootstrap-a.

Primjer koda 8. prikazuje korištenje Bootstrap razvojnog okvira za jedan stupac koji se definira klasom „col-sm-4“.

4.3.6 Vue.js

Vue.js je jednostavan i progresivan JavaScript okvir za izgradnju korisničkih sučelja. Nudi reaktivnost, komponentnu arhitekturu i jednostavan šablonski sustav poput Vuetify.js-a. Sa svojim bogatim ekosustavom i aktivnom zajednicom, Vue.js pruža jednostavan način za izgradnju skalabilnih i interaktivnih mrežnih aplikacija. S njegovom direktivama, konfiguriranjem i bibliotekom Vuex za upravljanje stanjem, Vue.js olakšava razvoj i održavanje aplikacija. Bez obzira na razinu iskustva, Vue.js je izvrstan izbor za moderni mrežni razvoj.

```
<div id="app">
<h1>{{ message }}</h1>
</div>
<script>var myObject = new Vue({
    el: '#app', data: {message: 'Hello Vue!'}
}) </script>
```

Primjer koda 9 Struktura koda Vue.js-a.

U primjeru koda 9. koristi se Vue.js biblioteka koja stvara jednostavnu aplikaciju koja prikazuje poruku „Hello Vue!“.

4.3.6.1 Vuetify.js

Vuetify.js je popularna biblioteka komponenata otvorenog koda za Vue.js okvir. Sa svojim bogatim skupom gotovih komponenata, Vuetify.js omogućuje brzo i jednostavno stvaranje modernih i atraktivnih korisničkih sučelja. Ova biblioteka se temelji na principima materijalnog dizajna, pružajući uzvratne, prilagodljive i estetski privlačne elemente. S Vuetify.js, mogu se koristiti navigacijske trake, kartice, oblike, dijaloge, padajuće izbornike, kalendare i mnoge druge komponente kako bi se izgradile profesionalne mrežne aplikacije. Također, Vuetify.js pruža mogućnost prilagodbe izgleda, teme i stilova kako bi se osiguralo da korisničko sučelje odgovara potrebama korisnika. Vuetify.js biblioteka olakšava razvoj i održavanje korisničkog sučelja, čineći ga izvrsnim izborom za izgradnju modernih Vue.js aplikacija.

4.4 PHP

PHP je popularan skriptni jezik za razvoj dinamičkih mrežnih aplikacija. Ovaj jezik stvorio je Rasmus Lerdorf 1994. godine. Prvotno jezik je stvoren kao koncept alata za manipulaciju osobnim mrežnim stranicama, ali je kasnije evoluirao u svestrani jezik za izgradnju složenih aplikacija. PHP se izvršava na poslužiteljskoj strani, što znači da se kod izvršava na poslužitelju prije nego što se rezultat šalje korisniku.

PHP se razlikuje od ostalih programskih jezika poput JavaScripta i HTML-a na dva bitna načina:

- Izvršavanje na serverskoj strani: Za razliku od JavaScript-a koji se izvršava na klijentskoj strani (u mrežnom pregledniku korisnika), PHP se izvršava na poslužiteljskoj strani. To znači da se PHP kod izvršava na mrežnom poslužitelju prije nego što se rezultat šalje korisniku.
- Programski jezik: PHP je samostalan programski jezik, dok su JavaScript i HTML jezici koji se koriste unutar mrežnog preglednika.

Sve .php datoteke započinju otvaranjem PHP oznake <?php, a završavaju s ?>. To omogućava označavanje dijela koda koji se treba izvršiti na poslužiteljskoj strani.

PHP ima bogatu podršku za rad s bazama podataka. Pruža razne funkcionalnosti i sučelja za povezivanje, upravljanje i manipulaciju podacima u bazama podataka. Kroz integraciju s popularnim sustavima upravljanja bazama podataka poput MySQL, PostgreSQL i SQLite, PHP omogućuje izradu dinamičkih mrežnih aplikacija koje mogu čitati, upisivati i ažurirati podatke u bazama podataka.

PHP je postao jedan od najpopularnijih jezika za mrežni razvoj diljem svijeta zbog svoje jednostavnosti, fleksibilnosti i velike zajednice programera koja pruža podršku i razvija mnoge biblioteke i okvire koji olakšavaju izgradnju složenih mrežnih aplikacija.

```
<?php
$servername = "localhost";
$username = "rentacar_pero";
$password = "132S56s2!";
$database = "rentacar_baza";

$conn = new mysqli($servername, $username, $password, $database);

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

Primjer koda 10 Struktura koda PHP-a.

Primjer koda 10. prikazuje PHP kod koji se koristi za uspostavljanje veze s bazom podataka putem MySQL poslužitelja pomoću pruženih podataka o poslužitelju. U slučaju da veza nije uspostavljena prikazuje se poruka koja sadrži informacije o grešci vraćene od strane baze podataka.

4.5 LARAVEL

Laravel je popularan okvir otvorenog koda za razvoj mrežnih aplikacija pisan u programskom jeziku PHP. Ovaj okvir je stvoren kako bi olakšao i ubrzao proces razvoja mrežnih aplikacija pružajući programerima intuitivnu sintaksu i bogat skup alata. Laravel je prvi put objavljen 2011. godine, a njegov glavni tvorac je Taylor Otwell.

Jedna od ključnih karakteristika Laravela je elegantna sintaksa koja omogućuje brzo i čitljivo pisanje koda. Okvir promovira čiste i modularne arhitekture, olakšavajući organizaciju koda i održavanje aplikacija. Laravel pruža mnoge ugrađene funkcionalnosti kao što su autentifikacija, autorizacija, konfiguriranje, upravljanje bazom podataka i predmemorijski sustav. Također podržava i integraciju s popularnim bazama podataka poput MySQL, PostgreSQL i SQLite.

Laravel promiče upotrebu MVC (*eng. Model-View-Controller*) arhitekture, koja omogućuje razdvajanje logike aplikacije od prikaza. Ovo pruža veću fleksibilnost i ponovnu upotrebljivost koda, olakšava testiranje i održavanje aplikacija. Laravel također dolazi s ugrađenim alatom za testiranje, koji olakšava pisanje i izvođenje jediničnih i funkcionalnih testova.

Jedna od najjačih strana Laravela je velika i aktivna zajednica programera koja pruža podršku, dijeli znanje i razvija dodatne biblioteke i proširenja. Laravelov paketni menadžer "Composer" omogućuje jednostavno dodavanje vanjskih biblioteka i proširenja u aplikaciju.

Uz sve navedeno, Laravel nudi i mnoge druge napredne značajke poput planiranja poslova, slanja elektroničke-pošte, obrade slika i rad s API-jima. Ovaj okvir je postao jedan od najpopularnijih izbora za razvoj mrežnih aplikacija zbog svoje moćne funkcionalnosti, intuitivnog sučelja i brzog rasta zajednice.

4.6 MAPBOX

Mapbox je američka tehnološka tvrtka specijalizirana za razvoj digitalnih karata, lokacijskih usluga i geoprostornih alata. Tvrtka je osnovana 2010. godine i ima sjedište u Washingtonu, SAD. Mapbox pruža napredne alate, SDK-ove (*eng. Software Development Kits*) i API-je (*eng. Application Programming Interfaces*) koji omogućuju programerima integraciju interaktivnih karata i geoprostornih funkcionalnosti u svoje aplikacije i platforme. Ključni proizvod tvrtke je Mapbox Platform, koja obuhvaća nekoliko ključnih komponenti:

- Mapbox Maps: Pruža širok spektar karata koje se mogu prilagoditi prema potrebama korisnika. Ove karte podržavaju različite slojeve podataka, stilove, interaktivnost i vizualne prilagodbe.

- Mapbox Studio: Alat koji omogućuje dizajniranje i prilagodbu karata putem intuitivnog sučelja. Korisnici mogu kontrolirati stilove, boje, simbole i mnoge druge vizualne aspekte kako bi stvorili karu prilagođenu svom brendu ili dizajnu.
- Mapbox Navigation: Pruža napredne lokacijske usluge za razvoj navigacijskih aplikacija. Uključuje funkcionalnosti poput smjernica za vožnju, upozorenja o prometu, navigacije pješaka i drugih navigacijskih značajki.
- Mapbox Geocoding: Omogućuje pretvaranje adresa ili geografskih koordinata u lokacijske podatke (geo-kodiranje) te obrnutu pretvorbu (obrnuto geo-kodiranje). Korisnici mogu dobiti precizne informacije o lokacijama na temelju unesenih adresa ili koordinata.

Mapbox je popularan izbor za razvoj mobilnih i mrežnih aplikacija koje zahtijevaju kartografske i geoprostorne funkcionalnosti. Tvrtka je poznata po pružanju visokokvalitetnih karata s prilagodljivim stilovima i naprednim mogućnostima interakcije. Mapbox je integriran u mnoge poznate platforme i aplikacije, uključujući aplikacije poput Foursquare, Pinterest i Snapchat.

Važno je napomenuti da Mapbox koristi besplatni probni model poslovanja, gdje pruža besplatni pristup ograničenim mogućnostima, ali nudi i naprednije značajke uz plaćanje prema upotrebi. Ovo omogućuje programerima da iskoriste temeljne funkcionalnosti Mapbox-a bez potrebe za velikim ulaganjima, dok tvrtke koje imaju velike potrebe za geoprostornim uslugama mogu prilagoditi svoje planove i pakete prema svojim potrebama.

Mapbox aktivno doprinosi otvorenim izvorima podataka i podržava OpenStreetMap, globalnu zajednicu koja stvara i održava slobodne i detaljne geoprostorne podatke. Otvoreni karakter Mapbox-a omogućuje programerima i korisnicima da prilagode i poboljšaju svoje karte i lokacijske usluge na temelju otvorenih podataka i zajedničkog razvoja.

4.7 ARHITEKTURA

Arhitektura baze podataka predstavlja strukturu baze koja se sastoji od tri "sloja" i interakcija među njima. Tri nivoa apstrakcije su sljedeći: fizički nivo, globalni logički nivo i lokalni logički nivo. Fizički nivo obuhvaća fizički prikaz i raspored podataka na

vanjskim memorijskim jedinicama. Globalni logički nivo odnosi se na logičku strukturu cijele baze podataka, dok se lokalni logički nivo odnosi na logički prikaz dijela baze koji je specifičan za određenu aplikaciju.

Jedan popularan arhitekturni model koji se često koristi u razvoju aplikacija je MVC model (Model-View-Controller). Ovaj model pomaže u organizaciji i strukturiranju aplikacija.

U kontekstu baze podataka, baza aplikacije je izgrađena prema MVC modelu. To znači da baza podataka predstavlja "model" aplikacije, koji sadrži podatke i logiku pristupa tim podacima. Ostale komponente, poput prikaza i kontrolera, koriste se za upravljanje prikazom podataka i obradu korisničkih zahtjeva. MVC model omogućuje bolju organizaciju i odvajanje odgovornosti unutar aplikacije, što olakšava održavanje i skalabilnost.

4.7.1 Baza podataka

Pojam baze podataka odnosi se na strukturiranu kolekciju međusobno povezanih podataka. Baza podataka predstavlja ne ponavljajući skup informacija o stanju poslovne organizacije, koji je opisan u shemi baze podataka. Sadrži podatke o entitetima, njihovim vezama i atributima, koji su definirani u shemi baze podataka. Arhitektura baze podataka sastoji se od tri sloja: fizičkog, globalnog logičkog i lokalnog logičkog sloja. Fizički sloj odnosi se na fizički prikaz i organizaciju podataka na vanjskim memorijama. Globalni logički sloj odnosi se na logičku strukturu cijele baze podataka, dok se lokalni logički sloj odnosi na logički prikaz dijela baze podataka koji koristi određena aplikacija.

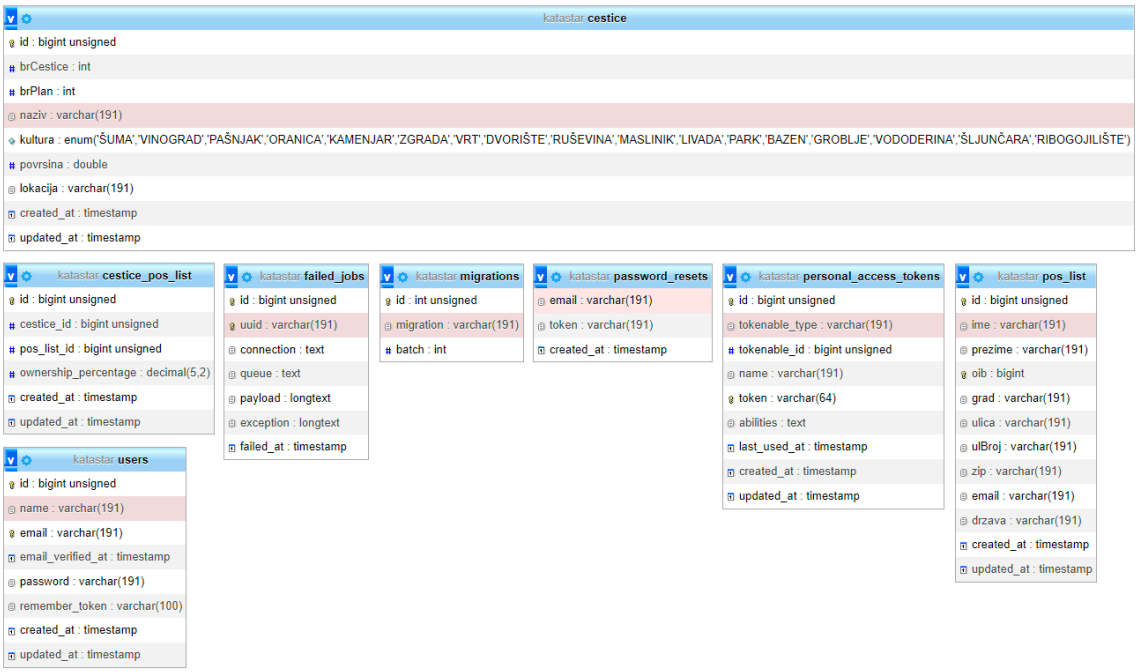
Na slici 2. se vidi izgled baze podataka Katastar. Sve tablice se kreiraju kroz programski okvir Laravel. Sustav koristi osam tablica unutar baze podataka: *cestice, cestice_pos_list, failed_jobs, migrations, password_resets, personal_access_tokens, pos_list* I *users.*

Unutar tablice *cestice* nalaze se svi podaci o katastarskim česticama.

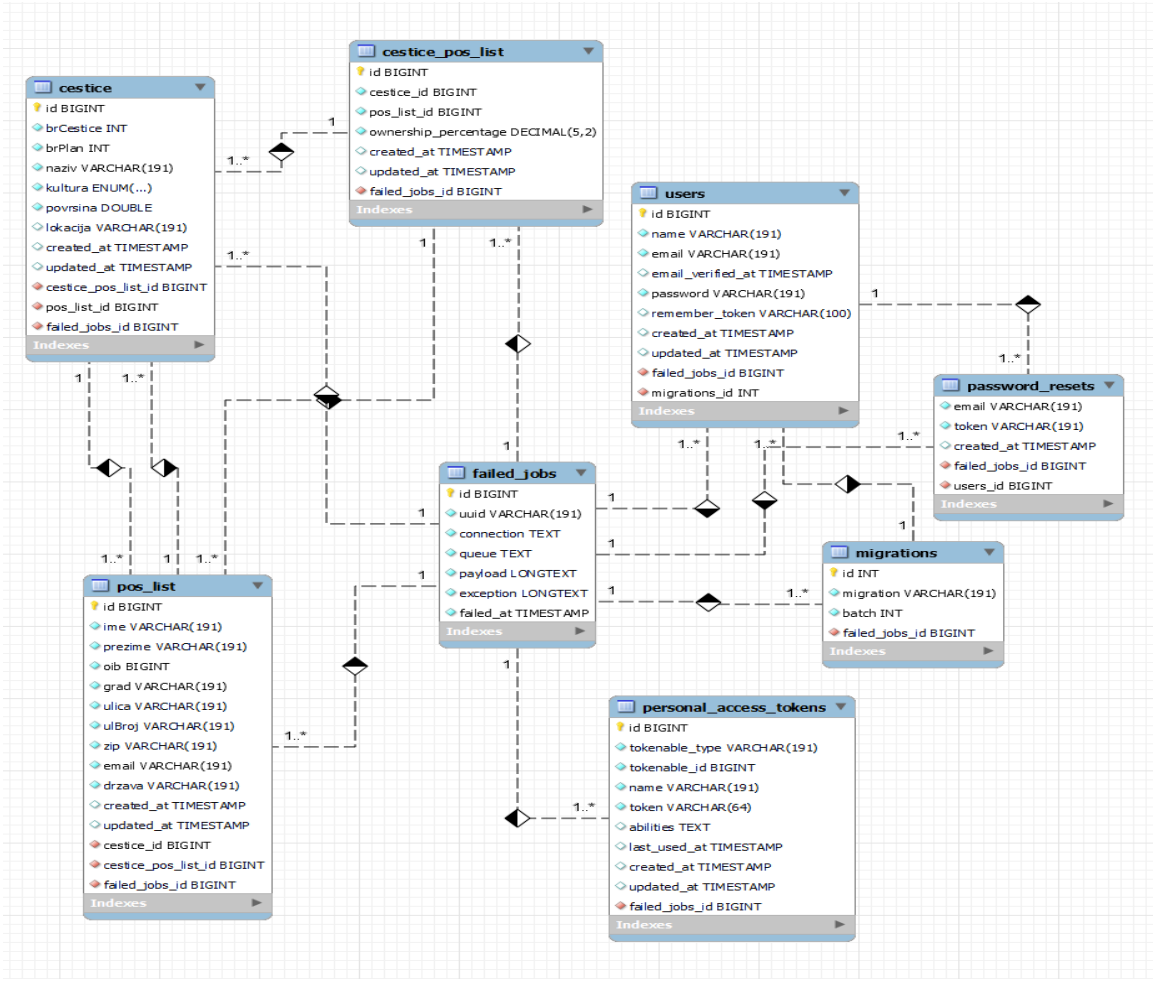
U tablici *cestice_pos_list* nalaze se podaci vlasništva i udjela nad pojedinačnim katastarskim česticama.

U tablici *pos_list* nalaze se svi podaci vlasnika katastarskih čestica.

U tablici *users* se nalaze podaci svih korisnika koji imaju autorizirani pristup sustavu.



Slika 4 Prikaz baze podataka *katastar*



Slika 5 Prikaz relacija i ključeva tablica baze podataka *katastar*

4.7.2 MVC Model

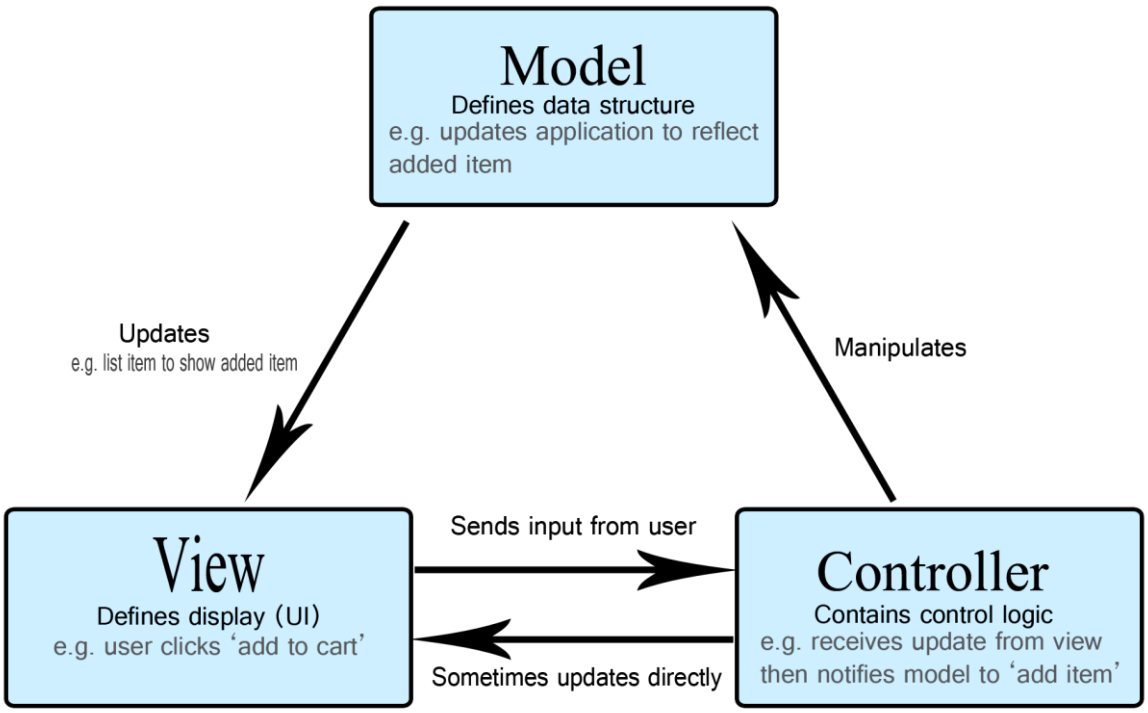
MVC (*eng. Model-View-Controller*) je arhitekturni obrazac koji se koristi u razvoju softvera za organizaciju i strukturiranje aplikacija. Ovaj obrazac je posebno popularan u razvoju mrežnih aplikacija, uključujući i razvoj upotrebom okvira poput Laravel-a. MVC arhitektura razdvaja aplikaciju na tri ključne komponente:

- Model (*eng. Model*): Model predstavlja sloj podataka u aplikaciji. Ovdje se obrađuju podaci, pristupa im se, te se podaci ažuriraju i pohranjuju. Model predstavlja logiku poslovnih pravila i pristupa podacima. On može sadržavati definicije baze podataka, upite za dohvat i spremanje podataka te poslovnu logiku koja se odnosi na obradu tih podataka.
- Pogled (*eng. View*): Pogled je sloj koji se odnosi na prezentaciju podataka korisniku. Čini ga korisničko sučelje koje prikazuje informacije iz modela na

način razumljiv korisniku. Pogled može biti HTML stranica, grafički sučelje ili bilo koja druga metoda vizualizacije podataka. Pogled ne bi trebao sadržavati poslovnu logiku ili izravno pristupati podacima, već samo prikazati podatke dobivene iz modela.

- Kontroler (*eng. Controller*): Kontroler je posrednik između modela i pogleda. On prima zahtjeve od korisnika i obrađuje ih prema poslovnoj logici aplikacije. Kontroler dohvaća potrebne podatke iz modela, a zatim ih prosljeđuje pogledu kako bi se prikazali korisniku. Kontroler također upravlja interakcijom korisnika s aplikacijom i izvršava odgovarajuće radnje.

MVC arhitektura omogućuje jasnu razdiobu odgovornosti između različitih komponenti aplikacije. To rezultira boljom organizacijom koda, poboljšanom modularnošću i olakšava testiranje i održavanje aplikacija. Također omogućuje višestruku upotrebu i skalabilnost, jer promjene u jednoj komponenti ne utječu na ostale. Ovaj obrazac omogućuje razdvajanje brige o podacima (model), prezentaciji podataka (pogled) i upravljanju korisničkim zahtjevima (kontroler). To olakšava suradnju timova razvojnih programera, dizajnera i drugih stručnjaka koji rade na različitim aspektima aplikacije. MVC je postao standard u razvoju mrežnih aplikacija zbog svoje efikasnosti, organizacije i skalabilnosti.



Slika 6 Prikaz veza između dijelova u arhitekturi MVC-a [25]

Prednosti MVC arhitekture obuhvaćaju mogućnost višestrukog istovremenog rada različitih programera na modelima, pogledima i kontrolerima. Također, pruža logičko grupiranje povezanih akcija unutar kontrolera. Modeli mogu biti povezani s više pogleda, što pruža fleksibilnost pri prikazu podataka.

Nedostaci ove arhitekture uključuju moguću kompleksnost navigacije unutar razvojnog okruženja zbog uvođenja dodatnih slojeva apstrakcije. Također, korisnicima se može tražiti prilagođavanje određenim kriterijima dekompozicije MVC arhitekture. Korištenje MVC zahtijeva od programera da budu vješti u korištenju više tehnologija istovremeno, što podrazumijeva da imaju znanje i iskustvo s tim tehnologijama kako bi efikasno implementirali arhitekturu.

5 IMPLEMENTACIJA INFORMACIJSKOG SUSTAVA

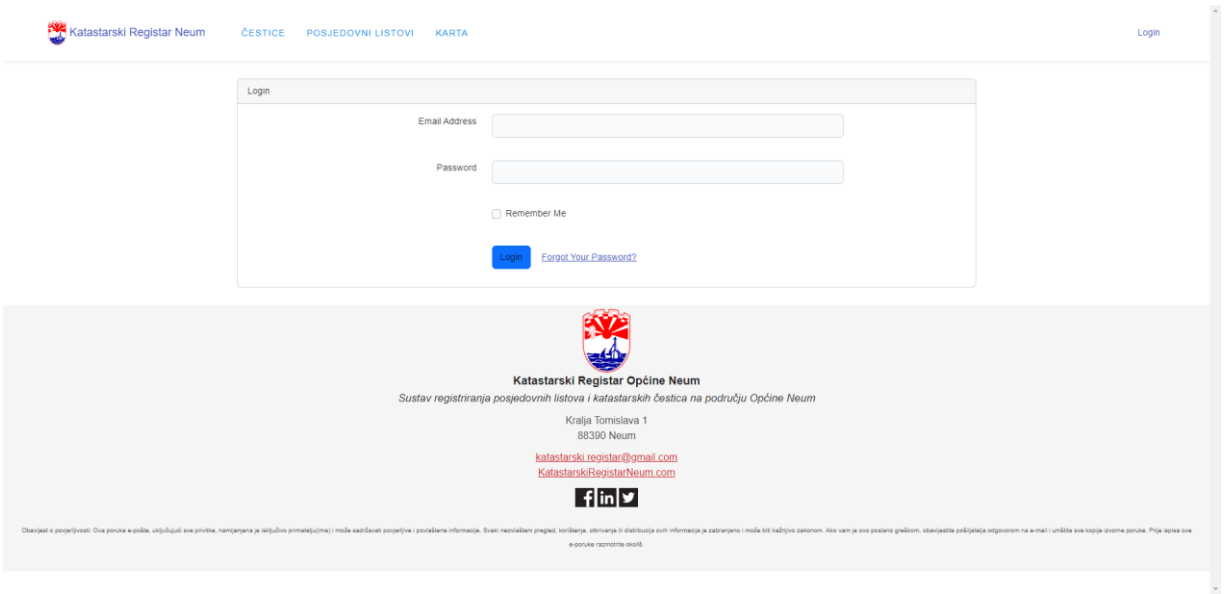
Ova cjelina opisuje izgled i funkcionalnost informacijskog sustava, odnosno mrežne aplikacije, svih njenih elemenata i sadržaja. Detaljnije će se opisati važne funkcionalnosti koje su važne za korisnika.

5.1 Korisničko sučelje aplikacije

Unutar korisničkog sučelja, nalaze se svi elementi aplikacije i sve potrebne podstranice kojima korisnik može pristupiti klikom na njih. Sama aplikacija je maksimalno prilagođena korisnicima i daje dobar pregled svih elemenata na njoj.

5.1.1 Pristup sustavu – Prijava

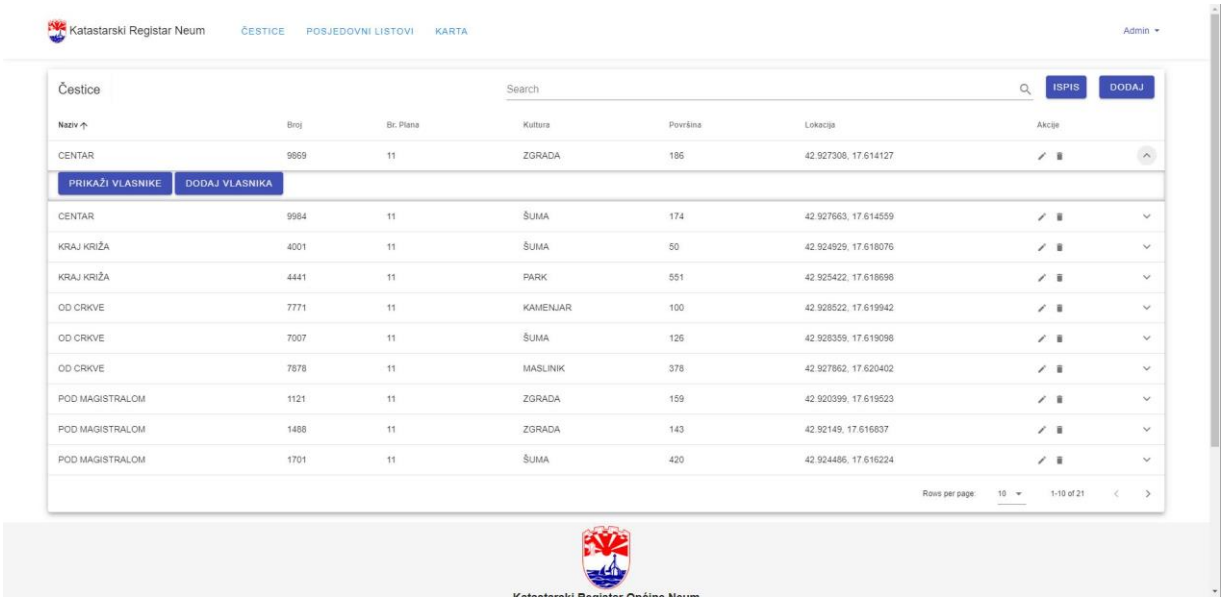
Slika 7. prikazuje stranicu za prijavu u sustav katastra nekretnina. Na slici se nalazi jednostavna forma za prijavu koja traži korisničke podatke za prijavu. Nakon unosa korisničkih podataka potrebno je pritisnuti gumb za prijavu. U slučaju zaboravljene lozinke, potrebno je pritisnuti gumb za zaboravljenu lozinku. Navigacijska traka je identična onoj u ostalim dijelovima sustava, no ovoj su onemogućene funkcionalnosti, a podnožje je identično na svim stranicama sustava.



Slika 7 Prijava u sustav

5.1.2 Čestice

Nakon prijave u sustav prikazuje se stranica *Čestice* kao na slici 8.. Na slici se odmah uočava lista čestica katastra nekretnina. Svaka čestica u listi ima svoj naziv, broj, broj plana, kulturu, površinu, koordinate lokacije i akcije. Akcije pružaju opciju brisanja ili izmjene podataka čestice. Uz navedeno klikom na malu strjelicu sa strane otvaraju se dvije nove mogućnosti rada s odabranom česticom, funkcionalnosti prikaza postojećeg/ih vlasnika katastarske čestice i mogućnost dodavanja novog vlasnika. Unutar okvira liste čestica postoje mogućnosti Ispisa i Dodavanja nove čestice u sustav. Navigacijski okvir, nakon prijave u sustav, postaje potpuno funkcionalan unutar cijelog sustava te omogućava korisniku brzu i jednostavnu navigaciju sustavom. Klikom na ime korisnika sustava, unutar navigacijskog okvira, nudi se mogućnost odjave.

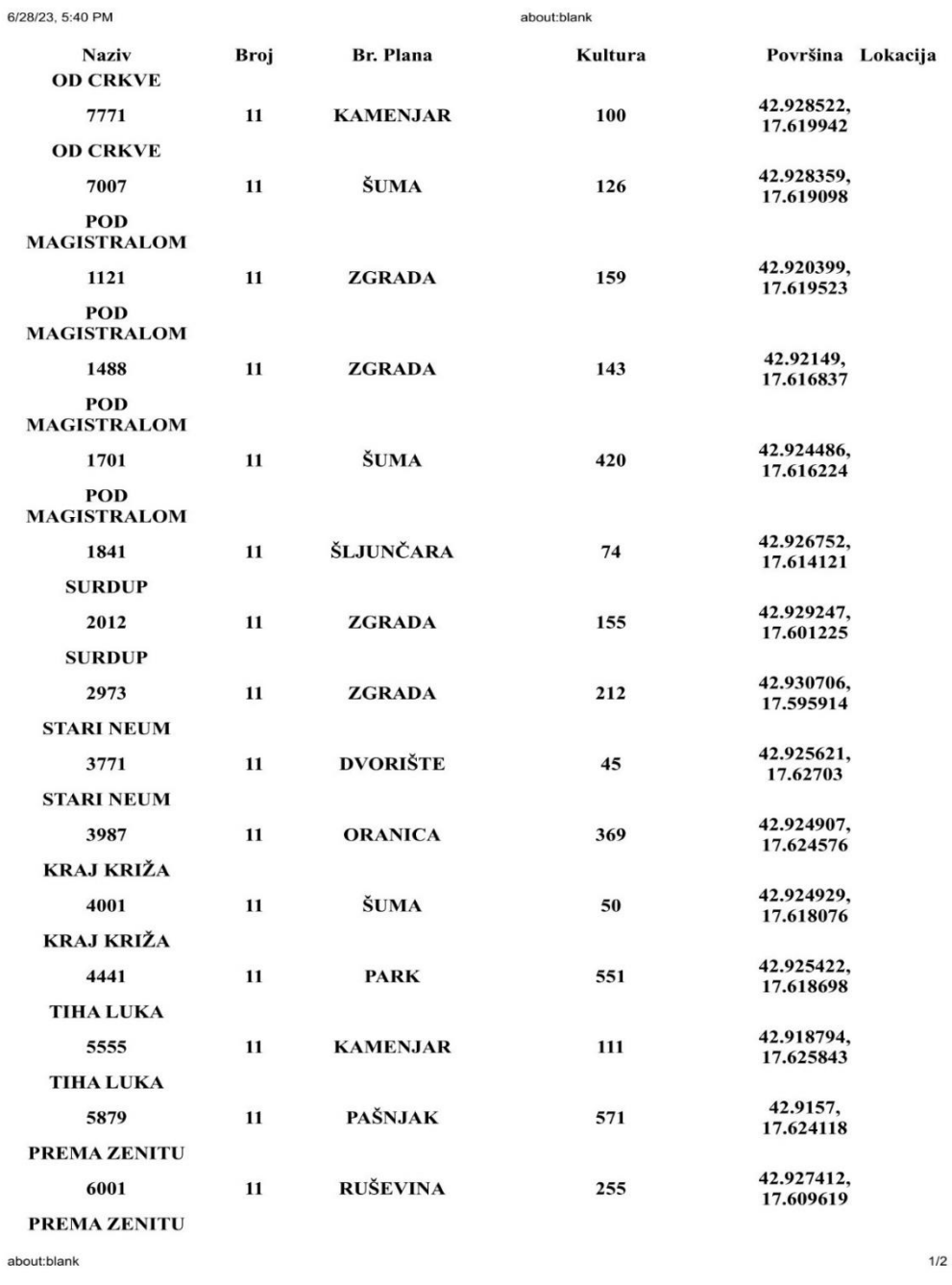


Slika 8 Izgled stranice Čestice

5.1.2.1Ispis

Klikom gumba za ispis, na stranici *Čestice* (*slika 8.*), otvara se preglednik-ov generički prozor za ispis na kojemu je potrebno odabrati pisač, izgled i broj stranica koje se želi

ispisati ili spremiti u .pdf format. Izgled ispisa podataka čestica katastra nekretnina je prikazan na slici 9.



6/28/23, 5:40 PM about:blank

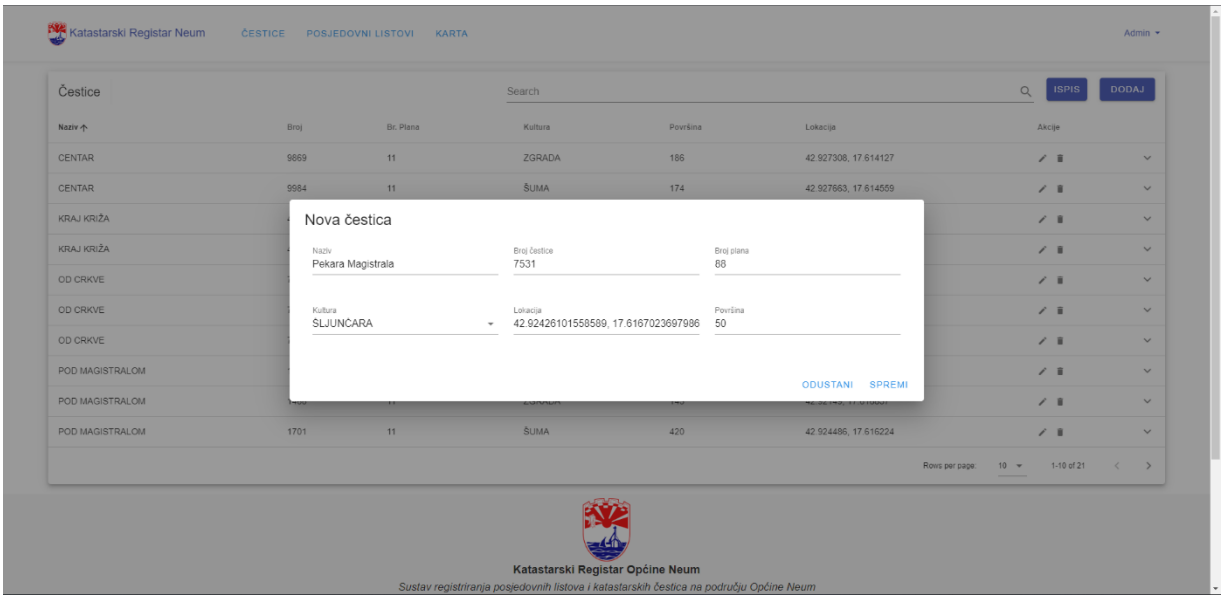
Naziv	Broj	Br. Plana	Kultura	Površina	Lokacija
OD CRKVE 7771	11	KAMENJAR	100	42.928522, 17.619942	
OD CRKVE 7007	11	ŠUMA	126	42.928359, 17.619098	
POD MAGISTRALOM 1121	11	ZGRADA	159	42.920399, 17.619523	
POD MAGISTRALOM 1488	11	ZGRADA	143	42.92149, 17.616837	
POD MAGISTRALOM 1701	11	ŠUMA	420	42.924486, 17.616224	
POD MAGISTRALOM 1841	11	ŠLJUNČARA	74	42.926752, 17.614121	
SURDUP 2012	11	ZGRADA	155	42.929247, 17.601225	
SURDUP 2973	11	ZGRADA	212	42.930706, 17.595914	
STARI NEUM 3771	11	DVORIŠTE	45	42.925621, 17.62703	
STARI NEUM 3987	11	ORANICA	369	42.924907, 17.624576	
KRAJ KRIŽA 4001	11	ŠUMA	50	42.924929, 17.618076	
KRAJ KRIŽA 4441	11	PARK	551	42.925422, 17.618698	
TIHA LUKA 5555	11	KAMENJAR	111	42.918794, 17.625843	
TIHA LUKA 5879	11	PAŠNJAK	571	42.9157, 17.624118	
PREMA ZENITU 6001	11	RUŠEVINA	255	42.927412, 17.609619	
PREMA ZENITU					

about:blank 1/2

Slika 9 Izgled ispisa čestica katastra nekretnina

5.1.2.2 Dodavanje nove čestice

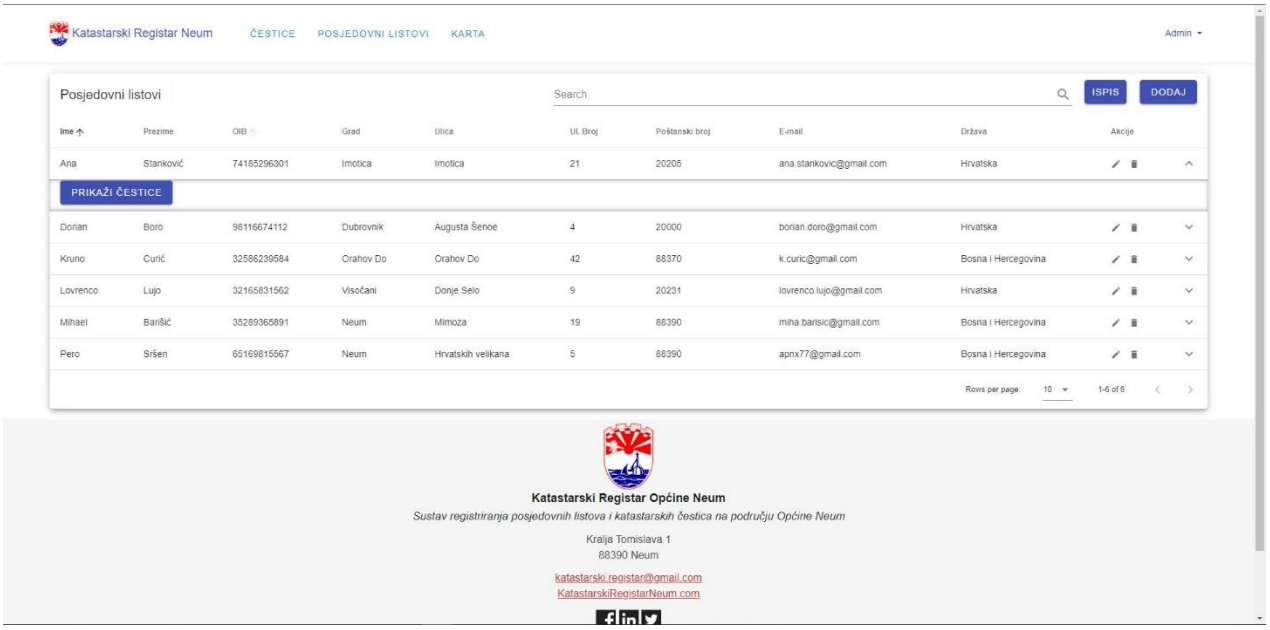
Klikom gumba *Dodaj*, na stranici *Čestice* (*slika 8.*), otvara se prozor *Nova čestica* (*slika 10.*) koji služi za unos podataka o novoj čestici. Forma dodavanja nove čestice zahtjeva unos podataka o čestici, a to su naziv, broj čestice, broj plana, kultura, koordinatna lokacija i površina. Nova čestica se ne može unijeti u sustav ako nije potpuna ili podaci nisu točno uneseni.



Slika 10 Prikaz prozora dodavanja nove čestice

5.1.3 Posjedovni list

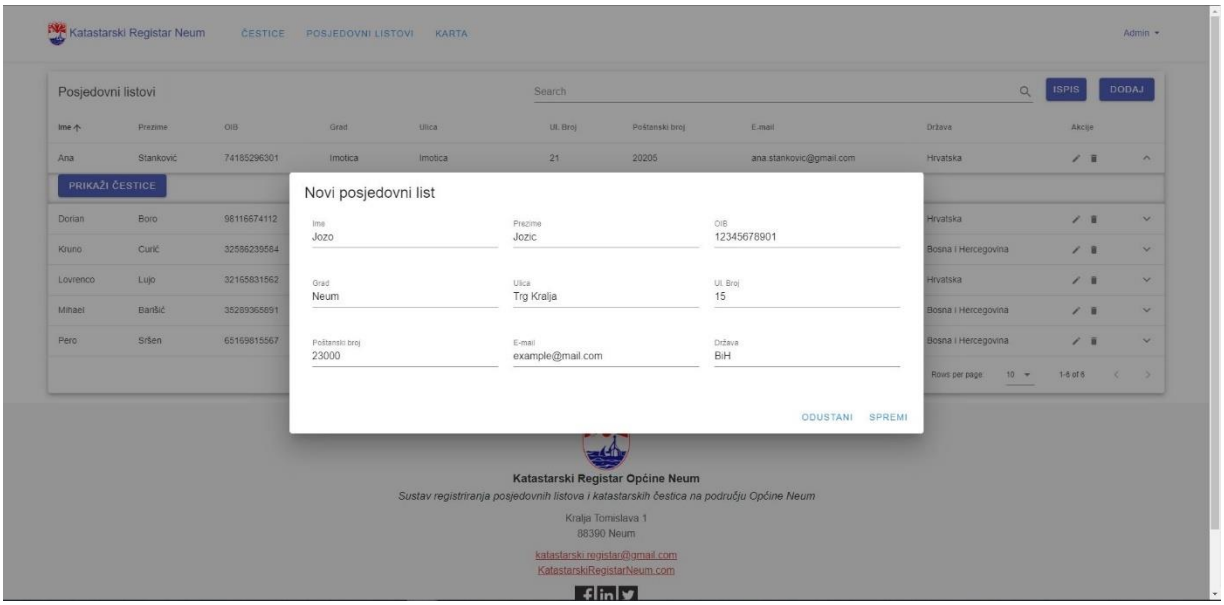
Otvaranjem stranice *Posjedovni listovi* (*slika 11.*) prikazuje se lista osoba koji posjeduju česticu katastarske nekretnine u sustavu i funkcionalnosti slične kao na stranici *Čestice*. Svaka osoba, u ovom slučaju vlasnik, unutar liste ima navedeno ime, prezime, OIB, grad, ulicu, ulični broj, poštanski broj, elektroničku poštu, državu i dodatne akcije. Akcije pružaju opciju brisanja ili izmjene podataka osobe, tj. vlasnika. Klikom na malu strjelicu sa strane otvara se dodatna funkcionalnost *Prikaza čestica* koja prikazuje sve čestice u posjedu odabrane osobe. Unutar samog okvira liste vlasnika su identične usluge za ispis i dodavanje novog posjedovnog lista. Treba napomenuti da je izgled ispisa traženih podataka posjedovnog lista sličan izgledu kao kod ispisa podataka čestica (*slika 9.*).



Slika 11 Izgled stranice Posjedovni listovi

5.1.2.1Dodavanje novog posjedovnog lista

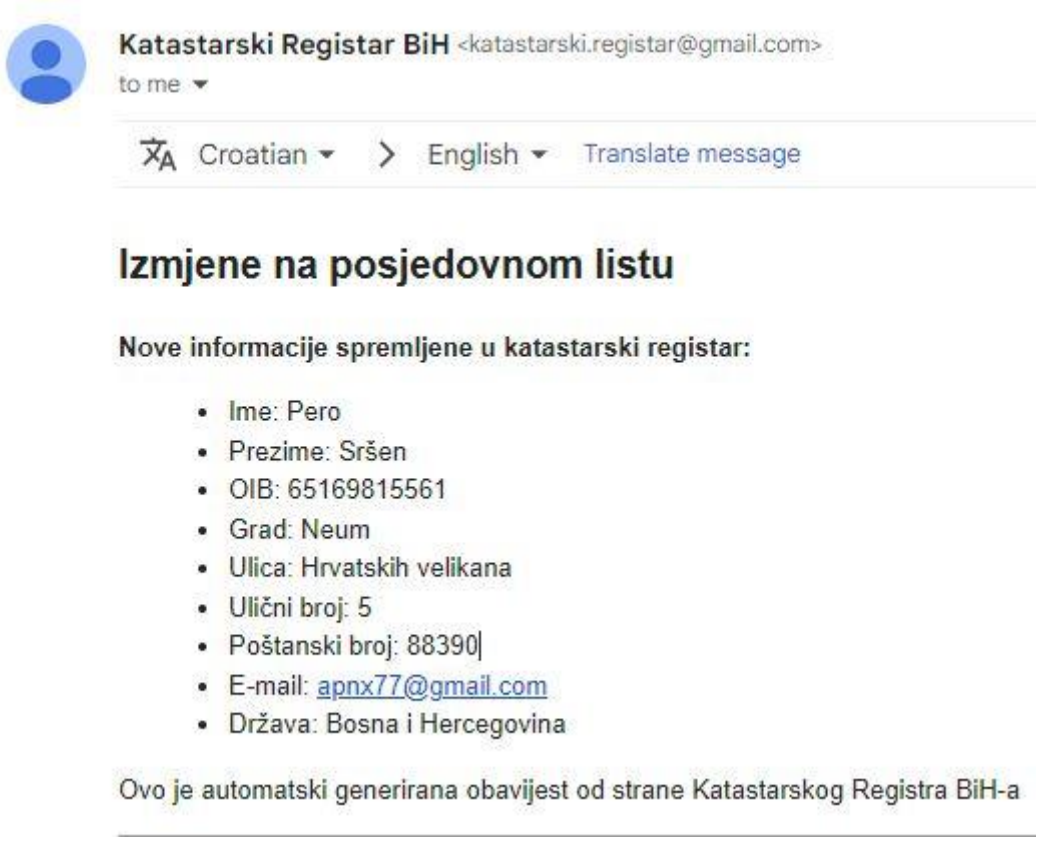
Klikom na *Dodaj,* na stranici *Posjedovni listovi* (*slika 10.*), otvara nam se prozor *Novi posjedovni list* (*slika 12.*) koji služi za unos podataka o novom posjedovnom listu. Forma dodavanja novog posjedovnog lista zahtjeva unos podataka o vlasniku čestice. To su ime i prezime osobe, OIB, grad, ulica, ulični broj, poštanski broj, adresa elektroničke pošte i država. Novi posjedovni list se ne može unijeti ako svi uvjeti nisu zadovoljeni.



Slika 12 Prikaz prozora dodavanja novog posjedovnog lista

5.1.2.2 Izgled obavijesti elektroničkom poštom

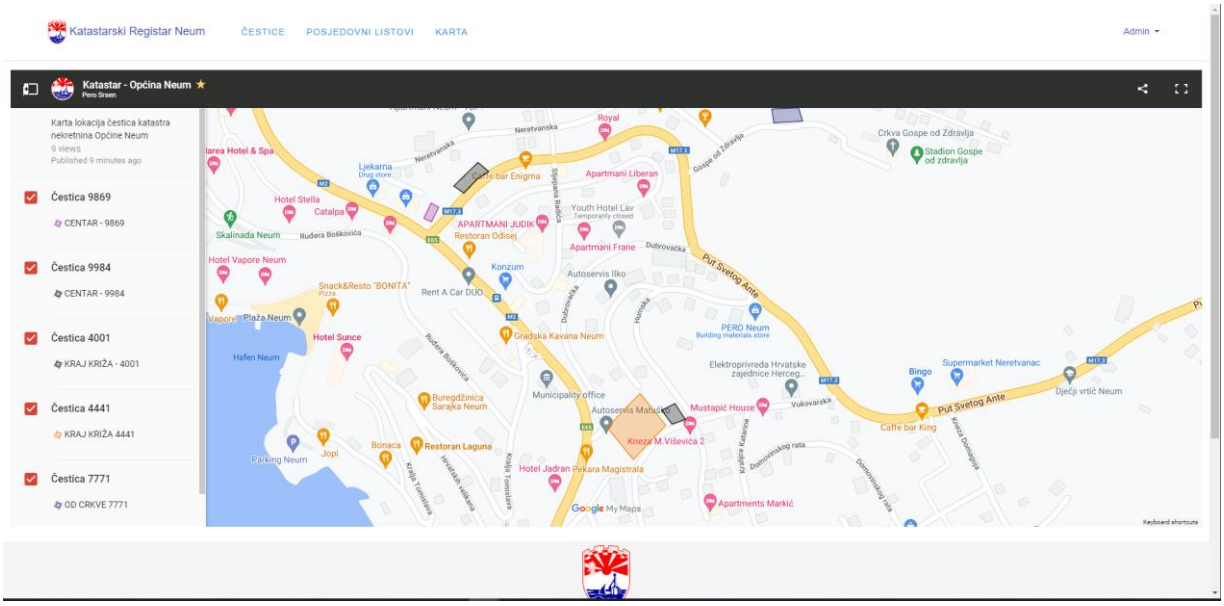
Na slici 13. prikazan je primjer obavijesti elektroničkom poštom. Obavijest se generira automatski od strane sustava i pritom obavještava osobu evidentiranu unutar sustava o bilo kakvoj promjeni, dodavanju ili brisanju podataka unutar sustava čestica katastra nekretnina. Ista funkcionalnost je kao i kod izmjene, dodavanja ili brisanja čestice unutar sustava.



Slika 13 Izgled obavijesti o izmjeni informacija unutar posjedovnog lista poslane elektroničkom poštom

5.1.3 Karta - lokacije čestica katastra nekretnina

Otvaranjem stranice *Karta* prikazuje se karta sa lokacijom čestica evidentiranih unutar sustava čestica katastra nekretnina (*slika 14.*). S lijeve strane otvorenog prozora dan je prikaz liste čestica na koje se može kliknuti te se posljedično na karti prikaže točna lokacija čestice. Za ovaj dio sustava trebao se koristiti Mapbox API, no zbog plaćanja usluge taj dio nije inkorporiran u sustav, pa se pri tom koristi besplatni Google Maps.



Slika 14 Izgled stranice Karta

6 ZAKLJUČAK

Implementacija informacijskog sustava za zemljišne knjige i katastar s mrežnom aplikacijom predstavlja značajan napredak u upravljanju i evidentiranju nekretnina. Ovaj sustav donosi brojne prednosti i olakšice kako korisnicima, tako i svim sudionicima uključenima u proces.

Prvo, važno je istaknuti da digitalni katastarski sustav omogućuje efikasno dodavanje, izmjenu i brisanje čestica te posjedovnih listova. Zahvaljujući ovim funkcionalnostima, procesi ažuriranja i upravljanja nekretninama postaju jednostavniji, brži i precizniji. Smanjuje se administrativni teret i potreba za ručnim unosom podataka, čime se minimizira mogućnost grešaka.

Karta koja vizualizira lokacije čestica pruža korisnicima intuitivni način za pregled i orijentaciju u prostoru. Vlasnici čestica mogu jednostavno prepoznati i identificirati svoju imovinu na karti, što olakšava njihovu interakciju s katastarskim sustavom. Ovaj vizualni prikaz dodatno poboljšava razumijevanje prostornih odnosa, pogotovo kod složenih i razgranatih područja.

Prednosti digitalnog katastarskog sustava proširuju se i na sve sudionike uključene u poslove vezane uz nekretnine. Za državne institucije, sustav donosi učinkovitiju evidenciju i upravljanje nekretninama, što rezultira poboljšanom administracijom i smanjenjem birokracije. Osim toga, pravna sigurnost u prometu nekretnina povećava se jer digitalni katastarski sustav osigurava pouzdane i ažurirane informacije. Nadalje, odvjetnici, javni bilježnici i drugi stručnjaci koji se bave nekretninama imaju koristi od ovog sustava jer im pruža brz pristup relevantnim podacima, čime se ubrzava proces obrade dokumenata i smanjuje vrijeme potrebno za pružanje usluga klijentima.

U konačnici, razvijeni informacijski sustav za zemljišne knjige i katastar ispunjava sve zahtjeve iz ovog rada te sa svojom mrežnom aplikacijom pruža značajne prednosti i olakšice u upravljanju nekretninama. Njegove funkcije dodavanja, izmjene i brisanja čestica te vizualizacija lokacija čestica na karti unaprjeđuju efikasnost, točnost i pristupačnost informacija. Time se osigurava bolja pravna sigurnost, smanjuje administrativni teret i olakšava rad svim korisnicima, posebno vlasnicima čestica, koji mogu jednostavno i brzo pristupiti podacima o svojoj imovini.

7 LITERATURA

[1] Vujić, Antun. *Opća i nacionalna enciklopedija u 20 knjiga.* Zagreb: PRO LEKSIS d.o.o.,2006.

[2] Narodne novine - Zakon, Zakon o državnoj izmjeri i katastru nekretnina, pristupljeno 18.6.2023. Preuzeto s https://narodne-novine.nn.hr/clanci/sluzbeni/2018_12_112_2167.html

[3] Narodne novine - Zakon, Zakon o upravnim pristojbama, pristupljeno 18.6.2023. Preuzeto s https://narodne-novine.nn.hr/clanci/sluzbeni/2016_12_115_2521.html

[4] Uređena zemlja, Uređena zemlja korisničko iskustvo, pristupljeno 18.6.2023. Preuzeto s https://oss.uredjenazemlja.hr

[5] Uređena zemlja, Uređena zemlja korisničko iskustvo izrezak - slika, pristupljeno 18.6.2023. Preuzeto s https://oss.uredjenazemlja.hr

[6] Uređena zemlja, Uređena zemlja karta, izrezak - slika, pristupljeno 18.6.2023. Preuzeto s https://oss.uredjenazemlja.hr/map

[7] MDN Web Docs, HTML content categories, pristupljeno 18.6.2023. [Mrežno] Preuzeto s: https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Content_categories

[8] HTML element content categories, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s: https://en.wikipedia.org/wiki/HTML#/media/File:HTML_element_content_categories.svg

[9] MDN Web Docs, Learn to style HTML using CSS, pristupljeno 18.6.2023. [Mrežno] Preuzeto s: https://developer.mozilla.org/en-US/docs/Learn/CSS

[10] MDN Web Docs, JavaScript, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s: https://developer.mozilla.org/en-US/docs/Web/JavaScript

[11] W3Schools, JavaScript Tutorial, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s: https://www.w3schools.com/js/

[12] ZetCode, Axios tutorial, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://zetcode.com/javascript/axios/

[13] W3Schools JS Ajax, AJAX Introduction, pristupljeno . 18.6.2023. [Mrežno] Preuzeto s https://www.w3schools.com/js/js_ajax_intro.asp

[14] W3Schools jQuery, JQuery tutorial, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://www.w3schools.com/jquery/

[15] W3Schools JS JSON, JSON-Introduction, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://www.w3schools.com/js/js_json_intro.asp

[16] W3Schools Bootstrap, Bootstrap 3 Tutorial, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://www.w3schools.com/bootstrap/

[17] Get Bootstrap, Get started with Bootstrap, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://getbootstrap.com/docs/5.3/getting-started/introduction/

[18] W3Schools WebDevelopment, What is Vue.js, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://www.w3schools.com/whatis/whatis_vue.asp

[19] Vue.js The Progressive JavaScript Framework, Vue.js Docs, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://vuejs.org/

[20] Vue Component Framework, Vuetify UI Library, pristupljeno 18.6.2023. [Mrežno] Preuzeto s https://vuetifyjs.com/en/

[21] PHP Manual, Introduction, What is PHP? Pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://www.php.net/manual/en/intro-whatis.php

[22] Laravel Documentation, Introduction, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://laravel.com/docs/4.2/introduction

[23] MAPBOX, Maps and location for developers, pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://www.mapbox.com/

[24] MDN Web Docs, MVC Model-View-Controller Pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://developer.mozilla.org/en-US/docs/Glossary/MVC

[25] MDN Web Docs, MVC Model-View-Controller slika Pristupljeno 18.6.2023. [Mrežno]. Preuzeto s https://developer.mozilla.org/en-US/docs/Glossary/MVC/model-view-controller-light-blue.png

IZJAVA

Izjavljujem pod punom moralnom odgovornošću da sam završni rad izradio samostalno, isključivo znanjem stečenim na studijima Sveučilišta u Dubrovniku, služeći se navedenim izvorima podataka i uz stručno vodstvo mentorice doc. dr. sc. Ivone Zakarije i komentora Ivana Grbavca, dipl. ing. kojima se još jednom srdačno zahvaljujem.

Pero Sršen