**UNIVERZA V LJUBLJANI**

**FAKULTETA ZA UPRAVO**

Diplomsko delo

**RAZVOJ APLIKACIJE ZA OZNAČEVANJE LOKACIJ DIVJIH ODLAGALIŠČ**

Študent: Tine Pretnar

Vpisna številka: 04170151

Študijski program: Upravna informatika

Mentorica: doc. dr. Tina Jukić

Ljubljana. Maj 2023

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Podpisani Tine Pretnar, študent 1. bolonjske stopnje smeri Upravna Informatika, z vpisno številko 04170151, sem avtor diplomskega dela z naslovom Razvoj aplikacije za označevanje divjih odlagališč.

S svojim podpisom zagotavljam, da:

* je predloženo delo izključno rezultat mojega lastnega raziskovalnega dela,
* sem poskrbel, da so dela in mnenja drugih avtorjev oziroma avtoric, ki jih uporabljam v predloženem delu, navedena oziroma citirana v skladu s fakultetnimi navodili
* sem poskrbel, da so vsa dela in mnenja drugih avtorjev oziroma avtoric navedena v seznamu literature in virov, ki so sestavni element predloženega dela in je zapisan v skladu s fakultetnimi navodili,
* sem pridobil vsa dovoljenja za uporabo avtorskih del, ki so v celoti prenesena v predloženo delo in sem to tudi jasno zapisal v predloženem delu,
* se zavedam, da je plagiatorstvo-predstavljanje tujih del, bodisi v oblika citata bodisi v obliki skoraj dobesednega parafraziranja bodisi v grafični obliki, s katerim so tuje misli oziroma ideje predstavljene kot moje lastne-kaznivo po zakonu (Zakon o avtorskih in sorodnih pravicah, Uradni list RS, št. 21/95), kršitev pa se sankcionira tudi z ukrepi po pravilih Univerze v Ljubljani in Fakultete za upravo,
* se zavedam posledic, ki jih dokazano plagiatorstvo lahko predstavlja za predloženo delo in za moj status na Fakulteti za upravo,
* soglašam z objavo dela v Repozitoriju Univerze v Ljubljani.

Diplomsko delo je lektoriral .

Ljubljana, datum oddaje dela

Podpis avtorja:\*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

POVZETEK

Besedilo

Besedilo

Besedilo

**Ključne besede**:

ABSTRACT

DEVELOPMENT OF AN APPLICATION FOR RECORDING OF ILLEGAL DUMPING SITE LOCATIONS

Text

Text

Text

**Keywords**:

KAZALO

[IZJAVA O AVTORSTVU DIPLOMSKEGA DELA i](#_Toc143428267)

[POVZETEK ii](#_Toc143428268)

[ABSTRACT iii](#_Toc143428269)

[KAZALO PONAZORITEV v](#_Toc143428270)

[1 UVOD 1](#_Toc143428271)

[2 DIVJA ODLAGALIŠČA V SLOVENIJI 2](#_Toc143428272)

[2.1 NASTANEK DIVJIH ODLAGALIŠČ 2](#_Toc143428273)

[2.2 PREPREČEVANJE DIVJIH ODLAGALIŠČ 3](#_Toc143428274)

[3 TRENUTNI SISTEM ZA PRIJAVO DIVJIH ODLAGALIŠČ 4](#_Toc143428275)

[3.1 PRISTOJNI INŠPEKTORATI 4](#_Toc143428276)

[3.2 DRUGI NAČINI PRIJAVE DIVJIH ODLAGALIŠČ 7](#_Toc143428277)

[4 SPECIFIKACIJA ZAHTEV 7](#_Toc143428278)

[4.1 UPORABNIŠKE VLOGE 7](#_Toc143428279)

[4.2 FUNKCIONALNE ZAHTEVE 8](#_Toc143428280)

[4.3 NEFUNKCIONALNE ZAHTEVE 15](#_Toc143428281)

[4.4 SLOVAR POJMOV 16](#_Toc143428282)

[4.5 DIAGRAM PRIMEROV UPORABE 17](#_Toc143428283)

[4.6 OPIS VMESNIKOV 18](#_Toc143428284)

[5 ZASNOVA INFORMACIJSKE REŠITVE ZA PROCES PRIJAVE DIVJEGA ODLAGALIŠČA 19](#_Toc143428285)

[5.1 PROCESNI MODEL PRIJAVE DIVJEGA ODLAGALIŠČA V NOVEM SISTEMU 19](#_Toc143428286)

[5.2 PODATKOVNA BAZA 21](#_Toc143428287)

[5.3 ANGULAR(FRONTEND) 22](#_Toc143428288)

[5.4 SPRING BOOT(BACKEND) 22](#_Toc143428289)

[LITRETURA IN VIRI 23](#_Toc143428290)

KAZALO PONAZORITEV

KAZALO SLIK

[**Slika 1: UML diagram za aplikacijo** 17](#_Toc144058778)

[**Slika 2: Zaslonska maska-prikaz divjih odlagališč** 19](#_Toc144058779)

[**Slika 3: Zaslonska maska-prikaz podatkov o izbranem divjem odlagališču** 19](#_Toc144058780)

[**Slika 4: Zaslonska maska-prijava uporabnika v sistem** 20](#_Toc144058781)

[**Slika 5: Celoten proces prijave divjega odlagališča** 22](#_Toc144058782)

[**Slika 6: Začetek razreda Odlagališča** 26](#_Toc144058783)

[**Slika 7: Koda za HTTP zahtevo** 28](#_Toc144058784)

[**Slika 8: Prikaz divjih odlagališč na zemljevidu** 29](#_Toc144058785)

[**Slika 9: Prikaz podrobnosti o divjih odlagališčih** 31](#_Toc144058786)

KAZALO TABEL

[**Tabela 1: legenda simbolov procesnega modela** 19](#_Toc143428316)

# 1 UVOD

Črna odlagališča, predstavljajo pomembno grožnjo okolju, javnemu zdravju in varnosti. Ta odlagališča nastanejo, ko posamezniki ali podjetja odložijo odpadke na neprimernih lokacijah, kot so odprta polja, gozdovi, reke ali zapuščena zemljišča. Nezakonito odlaganje odpadkov lahko povzroči sproščanje škodljivih kemikalij in onesnaževanja tal, zraka in vode, kar lahko privede do onesnaženja ekosistema in morebitnih zdravstvenih tveganj za ljudi in živali.

Poleg tega so črna odlagališča moteča in zmanjšujejo vrednost nepremičnin ter zemljišč, odvračajo od gospodarskega razvoja in negativno vplivajo na turizem. Čiščenje črnih odlagališč je lahko tudi drago in dolgotrajno, pogosto zahteva sodelovanje več agencij in deležnikov.

V Sloveniji je največja zbirka črnih odlagališč register, ki pa je star, neprijazen uporabnikom in nepopoln. Register ni državni, temveč je last neprofitne organizacije Ekologi brez meja, ki pa nima dovolj finančnih virov za vzdrževanje podatkov in nadgradnjo sistema. Baza podatkov, ki dokumentira črna odlagališča, je ključnega pomena, saj zagotavlja dragocene informacije, ki lahko pomagajo identificirati vire in vrste odpadnih materialov, spremljati obseg in lokacijo problema ter nam lahko olajšajo čiščenje oz. sanacijo onesnaženih območij.

Pomanjkanje ustrezne baze podatkov, ki dokumentira črna odlagališča, lahko tudi ovira prizadevanja pravosodnih organov pri preiskovanju in kaznovanju odgovornih za nezakonito odlaganje odpadkov. Brez podrobnih zapisov o lokacijah in vrstah odpadnih materialov, ki se odlagajo, je težje identificirati in kaznovati posameznike ali podjetja, ki ustvarjajo nova in polnijo stara črna odlagališča.

V diplomski nalogi želim narediti analizo trenutnega sistema za prijavo divjih odlagališč, raziskati trenutno stanje divjih odlagališč v Sloveniji in razviti novo rešitev. Raziskal bom tudi katero ministrstvo je pristojno za nadzor oziroma sanacijo divjih odlagališč, saj je letos(2023) prišlo do sprememb in se je ministrstvo za okolje in prostor razdelilo na dva ministrstva. Razvil bi aplikacijo, za prijavljanje in označevanje črnih odlagališč, aplikacija pa lahko deluje tudi kot baza podatkov. Lokacije črnih odlagališč bodo lahko imele tudi vrste odpadkov na odlagališču, slike in komentarje uporabnikov, če bodo oni to želeli. Prav tako pa bi lahko komunalne službe oz. pristojni za čiščenje divjih odlagališč označili, katere lokacije so že očiščene. Pri razvoju aplikacije bom izdelal tudi vso potrebno dokumentacijo, ki bo naredila aplikacijo bolj razumljivo in jasno ter bo poskrbela, da aplikacijo lahko uporabljajo tudi drugi. V izdelani aplikaciji bodo vključeni tudi avtomatizirani testi, ki bodo preverjali delovanje aplikacije, vendar pa bom poleg teh testov omogočil tudi preizkus aplikacije s strani uporabnikov. Zanesljiva baza podatkov lahko olajša pridobivanje finančnih sredstev za čiščenje in obnovo onesnaženih območij in zmanjša stroške sanacije. Ocenjujem, da bi aplikacija pripomogla k čistoči, zdravju in blaginji Slovenije.

# 2 DIVJA ODLAGALIŠČA V SLOVENIJI

Glede na najnovejše podatke iz Registra divjih odlagališč (stanje na dan 16. 6. 2023) v Sloveniji obstaja približno 8.939 neprečiščenih divjih odlagališč, od tega jih 2202 vsebuje nevarne odpadke. Največji delež, 25 %, jih je v Osrednjeslovenski statistični regiji, sledita Podravska regija z 11,6 % in Savinjska regija z 10,4 %. Ta razporeditev je posledica večje gostote prebivalstva na določenem območju, kar pomeni tudi večjo količino odpadkov. Na teh odlagališčih se predvsem kopičijo gradbeni odpadki, ki volumensko predstavljajo vsaj tri četrtine vseh odpadkov, po teži pa še več. Sledijo komunalni odpadki kot druga najpogostejša vrsta odpadkov, nato izrabljene gume ter drugi nevarni in nenevarni odpadki. Obstaja tudi nekaj odpadne električne in elektronske opreme, vendar v manjši meri. Navedene številke se nanašajo le na podrobno opisana odlagališča, vendar pa je dejstvo, da je divjih odlagališč v Sloveniji manj kot v preteklosti. Trenutno je v Sloveniji tudi 4834 očiščenih lokacij divjih odlagališč(996 divjih odlagališč je vsebovalo nevarne odpadke), kar nam pove, da se divja odlagališča sicer sanirajo ampak približno s tako hitrostjo kot nastajajo, saj je bilo decembra 2014 v Sloveniji 9437 neprečiščenih divjih odlagališč, danes pa jih je 8939. Za hitrejše zmanjšanje divjih odlagališč, je torej potrebno organizirati ali večje število sanacij ali na nek način zmanjšati število novo nastalih divjih odlagališč.

## 2.1 NASTANEK DIVJIH ODLAGALIŠČ

Divja odlagališča so pri nas začela nastajati v petdesetih letih prejšnjega stoletja, ko je industrializacija v Jugoslaviji hitro rasla. Tehnološki napredek je s seboj prinesel plastične izdelke in pa večje obilje kar se tiče količine izdelkov na trgu. Vsak si je lahko privoščil več, kar je pomenilo več odpadkov. Takrat se seveda odpadki še niso ločevali, tudi smetišča pa so bila redka in oddaljena. Tudi transport smeti je bil težji za posameznika, saj ni imel vsak avta in avti so bili manjši. Iz teh razlogov pa so ljudje začeli odlagati odpadke na skrite lokacije v naravnem okolju. To seveda ljudje počnejo še danes a manj kot včasih. Po mojem mnenju je v Slovenji tudi kaznovalna politika za odlaganje odpadkov na divja odlagališča neprimerna in prenizka.

Eden od glavnih problemov današnjega časa so izdelki iz plastike, namenjeni enkratni in kratkotrajni uporabi, kar je privedlo do velikega kopičenja in neustreznega odlaganja takšnih izdelkov. Ti izdelki se ne razgrajujejo, temveč se pod vplivom vremenskih razmer, staranjem in drugimi fizičnimi vplivi lomijo in drobijo na vse manjše delce - mikroplastiko, ki predstavlja velik problem, saj vstopa v kri živih bitij, tudi tistih, ki jih mi zaužijemo. Problem mikroplastike je, da ni dobrega načina za čiščenje mikroplastike iz narave. Divje odlagališče lahko očistimo ampak mikroplastika ostane v okolju, saj je ne moremo odstraniti, ker je v zelji in je premajhna, da bi jo odstranili. Leta 1950 je bilo na svetu proizvedenih le 1,5 milijona ton plastike, medtem ko leta 2023 ta številka že dosega 380 milijonov ton plastike v letu, pri čemer je 50 odstotkov te plastike namenjene zgolj enkratni uporabi.

Še en sodoben vir odpadkov pa so elektronski odpadki, ki se jih prav tako najde na divjih odlagališčih. Količina elektronskih odpadkov v svetu eksponentno raste, ker ima vedo več ljudi dostop do elektronskih naprav, prav tako pa elektronske naprave pogosto menjamo. Nepravilno ravnanje z elektronskimi odpadki lahko povzroči označevanje okolja, saj lahko nevarne kemikalije in težke kovine, ki se nahajajo v elektronskih odpadkih, izpustijo strupene stvari v okolje. To pa, tako kot pri mikroplastiki, vodi do onesnaževanja tal, vode in zraka, kar pa posledično ogroža živali, rastline in človeško zdravje. Številni elektronski odpadki vsebujejo tudi dragocene surovine, ki jih je mogoče reciklirati, vendar to pogosto ne izkoriščamo saj trenutno svet predela le okoli 20 odstotkov(Mrak & Damjanovič, 2019) elektronskih odpadkov.

V Sloveniji se divja odlagališča nahajajo predvsem v bližini naselij, na lokacijah, ki so skrite očem. Najpogosteje so v bližini cest in kolovozov. Razlogi zakaj ljudje kljub nesprejemljivosti in škodljivosti odlaganja odpadkov na divjih odlagališčih so predvsem neozaveščenost občanov, neurejen sistem ravnanja z odpadki, izogibanje stroškom, slab nadzor in neprimerne kazni. Včasih pa odpadki pristanejo na divjih odlagališčih samo zaradi neinformiranosti prebivalstva, saj se ljudje ne zavedajo, da lahko brezplačno odložijo odpadke na urejenih zbirališčih(Mrak & Damjanovič, 2019).

## 2.2 PREPREČEVANJE DIVJIH ODLAGALIŠČ

Trenutno je glavni način preprečevanja nezakonitega odlaganja inšpekcijski nadzor. Inšpekcijski postopek pa se lahko sproži šele, ko je divje odlagališče že nastalo in je škoda že povzročena. Trenutni uradni preventivni ukrepi se osredotočajo le na rezultate in ne na vzroke nelegalnega odlaganja odpadkov. Inšpekcijski postopek dostikrat zmanjša nezakonito aktivnost na območju, ampak le za nekaj časa, potem pa učinki izginejo in divje odlagališče zopet nastane. Večina sredstev, povezanih z divjimi odlagališči, je namenjenih njihovemu odstranjevanju in sanaciji območij. Ti stroški, pa bi lahko bili precej manjši, če bi bilo nekaj sredstev namenjenih za preprečevanje nastanka divjih odlagališč. Da bi se soočili s tem izzivom, je ključno aktivno povezovanje in sodelovanje lokalnih skupnosti, poleg tega je potrebno razviti premišljene politike in instrumente za preprečevanje ter te dosledno in kompetentno izvajati. Le skupen pristop, kjer lokalne skupnosti, vlade in pristojne institucije delujejo združeno, lahko prinese učinkovite rešitve in zmanjša nezakonito odlaganje odpadkov. Sodelovanje, načrtovanje in dosledna izvedba so ključni dejavniki za trajnostno upravljanje odpadkov in ohranjanje okolja za prihodnje generacije.

Običajno nezakonito odlaganje odpadkov zahteva manj truda kot zakonito odlaganje. Vendar pa lokalne skupnosti imajo možnost spremeniti to ravnovesje in olajšati zakonito odlaganje ter otežiti nezakonito odlaganje. Po čiščenju območij se pogosto soočamo s ponovnim odlaganjem odpadkov, kar so opazili tudi v društvu Ekologi brez meja med vseslovenskimi čistilnimi akcijami "Očistimo Slovenijo". S strukturnimi rešitvami lahko lokalne skupnosti zmanjšajo dostopnost do priljubljenih mest za nezakonito odlaganje, povečajo tveganje za odkritje storilcev in jih s tem odvrnejo od nezakonitih dejanj. Očiščenje območja v kombinaciji s postavitvijo opozorilnih znakov, ovir, osvetlitvijo, urejanjem okolice ali povečanjem preglednosti območja lahko prispeva k zmanjšanju ali celo odpravi ponavljajočega se odlaganja(Loznar et al., 2015).

Običajno se posamezniki in podjetja odločajo za odlaganje odpadkov na divja odlagališča, da bi se izognili stroškom legalne odstranitve. Podjetja, ki pri svojem delu, generirajo veliko odpadkov(proizvodnja, gradbeništvo, kemične in farmacevtske industrije, Trgovinske in veleprodajne dejavnosti, gostinstvo) lahko z nelegalnim odlaganjem odpadkov pridobijo oziroma prihranijo veliko denarja. Posamezna gospodinjstva pa se hočejo izogniti večjem trudu in stroškov odvoza odpadkov v zbirne centre. Občine, ki so finančno zmožne, bi morale zagotavljati brezplačne ali subvencionirane možnosti za odvoz in odlaganje odpadkov posameznikov in podjetij. To bi spodbudilo uporabo zbirnih mest in korektnih opcij odvoza odpadkov, kar pomeni manj odpadkov na divjih odlagališčih. Občine pa bi s tem postale tudi bolj privlačne za podjetja in turizem. Globe za odlaganje odpadkov na divjih odlagališčih so zasnovane tako, da se korist nelegalnega odlaganja zmanjša. Običajno mora storilec plačati tako globo, da krije sanacijo nelegalno odloženih odpadkov. Seveda pa je storilce težko odkriti, še težje pa je dokazati katere in koliko odpadkov je odvrgel storilec na določeno divje odlagališče, tako da sanacijo odlagališča običajno plača lokalna skupnost oziroma občina. Sem mnenja, da bi se globe za odlaganje odpadkov na divjih odlagališčih morale povečati, ali pa bi morali poleg globe opravljati še kakšno delo oziroma pomoč(ne denarna) pri sanaciji divjega odlagališča.

# 3 TRENUTNI SISTEM ZA PRIJAVO DIVJIH ODLAGALIŠČ

V tem poglavju bom obrazložil uraden postopek prijave divjega odlagališča, zakonsko podlago in prisojnosti inšpekcij, ki pokrivajo divja odlagališča. Prav tako pa bom omenil postopek neuradne prijave divjega odlagališča nevladni organizaciji.

## 3.1 PRISTOJNI INŠPEKTORATI

Prva stvar s katero se sooča prebivalec Slovenije, ko želi uradno prijaviti najdeno divje odlagališče je komu oziroma kam prijaviti le tega. V Sloveniji imamo trenutno dva inšpektorata, ki delno pokrivata področja odlaganja smeti in onesnaževanja. To sta inšpektorat za okolje in prostor in inšpektorat za naravne vire in prostor. Inšpektorat za okolje in prostor opravlja naloge inšpekcijskega nadzora nad izvajanjem predpisov iz naslednjih področij:

* *ravnanja z odpadki in čezmejnim pošiljanjem odpadkov,*
* industrijskega onesnaževanja voda in tveganja za okolje ter emisij snovi v vode iz naprav, za katere je predpisan obratovalni monitoring,
* emisij snovi v zrak iz nepremičnih virov onesnaževanja,
* dimnikarskih storitev,
* skladiščenja nevarnih snovi v nepremičnih skladiščnih posodah in nepremične opreme, ki vsebuje florirane toplogredne pline ali ozonu nevarne snovi ter gensko spremenjenih organizmov,
* hrupa,
* elektromagnetnega sevanja
* svetlobnega onesnaževanja,
* državne meteorološke, hidrološke, oceanografske in seizmološke službe,
* energetike ter drugih predpisov, ki pooblaščajo energetske inšpektorje za izvajanje inšpekcijskega nadzora ter
* nadzora nad izvajanjem predpisov s področja javnega potniškega prometa potnikov v notranjem cestnem prometu in čezmejnem cestnem prometu do prestopne točke v sosednji državi, ki jo določi organ javnega potniškega prometa, in obvezne gospodarske javne službe prevoza potnikov v notranjem in čezmejnem regijskem železniškem prometu.
* so organ Republike Slovenije, ki je pristojen za sprejemanje predhodnih pisnih prijav in izdaje soglasij za pošiljanje odpadkov čez meje ter vodenje upravnih postopkov po prijavah oziroma vlogah za pošiljanje odpadkov na ozemlje, z ozemlja ali čez ozemlje Republike Slovenije. Inšpektorat vodi tudi evidence o izdanih soglasjih in dejanskih prevozih, opravljenih v okviru teh soglasij.

V inšpektoratu za naravne vire in prostor, pa delujeta dve inšpekciji. To sta Gradbena in geodetska inšpekcija in pa Inšpekcija za naravne vire in rudarstvo, ki se ponovno deli na inšpektorje za naravo in vode in pa na rudarske inšpektorje. Pri področju divjih odlagališč nas zanimajo predvsem inšpektorji za naravo in vode, ki nadzirajo izvajanja predpisov za:

* varstvo in urejanje voda, vodna in priobalna zemljišča, vodne pravice, rabo vode, ter vodna soglasja in dovoljenja,
* delovanje gospodarskih javnih služb na področju oskrbe s pitno vodo ter opremljenost agromelioracij za odvajanje in čiščenje komunalne in padavinske odpadne vode,
* *izvajanja predpisov s področja varstva in ohranjanja narave oziroma varstvo prosto živečih rastlinskih in živalskih vrst, varstvo naravnih vrednot, izpolnjevanje zahtev naravovarstvenih soglasij in drugih dovoljenj ter spoštovanje predpisov glede vožnje v naravnem okolju.*

Zakonodaja določa zelo obsežno delovno področje inšpektorjev za naravo in vode, kar povzroči visoko število zavezancev, to pa pomeni, da pri nadzoru, ki ga morajo opraviti, določijo prioriteto z naslednjimi kriteriji:

* vpliv dejavnosti na vode in naravo
* zaveze za doseganje skladnosti z evropskim pravnim redom, ki jih mora zagotavljati Slovenija,
* glede na cilje nacionalnih strategij, akcijskih načrtov, operativnih programov ipd.,
* zaznan obseg kršitev na posameznih področjih
* ugotovitve monitoringov,
* analize obremenitev in vplivov na vode.

Za obravnavo primerov odvrženih odpadkov v naravi so odgovorni tako državna inšpekcija kot mestni, občinski in medobčinski inšpektorji, odvisno od naslednjih dejavnikov: ali je povzročitelj odpadkov znan ali neznan, ali so odpadki odloženi na zemljišču v lasti države ali občine, ter ali gre za komunalne odpadke ali druge vrste odpadkov.

V situacijah nezakonitega odlaganja odpadkov, kjer je povzročitelj identificiran in gre za komunalne odpadke, je odgovornost za nadzor v pristojnosti občin. V primerih, kjer je povzročitelj identificiran in gre za druge vrste odpadkov, je za ukrepanje pristojna državna inšpekcija.

V primeru, ko povzročitelj ni znan in je zemljišče v lasti države ali občine, so za izvajanje ukrepov pristojni tako inšpektorji Inšpektorata Republike Slovenije za okolje in energijo kot tudi inšpektorji Medobčinskih inšpektoratov.

Na ministerstvu za okolje, podnebje in energijo pravijo, da se prijava divjega odlagališča opravi na inšpektoratu za okolje in energijo, ki spada pod to ministrstvo. Prijavo lahko opravimo osebno, po pošti ali elektronsko. Za elektronsko prijavo sta potrebna elektronska prijava in elektronski podpis. V obrazcu za prijavo moramo izpolniti podatke o kršitelju, opisati lokacijo in opis kršitve in pa napisati datum kršitve. Nato pa, je potrebno izpolniti tudi podatke o prijavitelju, če prijavo opravljamo po pošti ali osebno.

Če želimo prijavo opraviti po uradnem postopku potem je dobro dobiti, čim več informacij o odlagališču, saj to poveča možnost, da se krivec najde. Koristne informacije pri prijavi so, datum in čas incidentov(dodatno odlaganje smeti na odlagališče), koordinate GPS odlagališča, lastnik zemljišča, podrobnosti o vozilih na lokaciji, število ljudi na lokaciji in ali so odlagali smeti ali ne, čim boljši opis ljudi, katera vrsta odpadkov je na odlagališču(komunalni, gradbeni, nevarni) in količina teh smeti. Na lokaciji se ne dotikajte odpadkov ali odlagališča, saj lahko uničite dokaze ali se dotaknite strupenih snovi, ne komunicirajte z ljudmi, če mislite, da izvajajo nezakonito dejanje in ne bodite preveč očitni pri zbiranju podatkov, če so na lokaciji ljudje, ki kršijo zakon.

## 3.2 DRUGI NAČINI PRIJAVE DIVJIH ODLAGALIŠČ

Divje odlagališče pa lahko prijavimo tudi na spletni strani ocistimo.si, ki pripada nevladni organizaciji Ekologi brez meja, kjer se nahaja največji register divjih odlagališč v Sloveniji. Tam lahko prijavimo divje odlagališče, prijava pa bo povzročila samo to, da bodo odlagališče shranili v svojo bazo podatkov in mogoče, bodo enkrat v prihodnosti prostovoljci počistili to odlagališče. Lahko se jim tudi pridružimo pri čiščenju odlagališč.

Več ljudi pa se odloča za prijavo v register pri Ekologih brez meja, saj je prijava anonima, lažja, potrebuje manj informacij in možnost, da se divje odlagališče počisti je veliko večja. Uporabnik samo odpre njihovo spletno stran in sledi navodilom, ki jih imajo v pet minutnem videoposnetku, ta pa na zelo podrobno in počasi pokaže postopek, ki pa nam lahko vzame manj, kot je dolžina videoposnetka. Če se po uradnem postopku krivec ne najde (možnost, da se storilec ne najde je velika), potem pristojni inšpektor odredi izvajalcu javne službe zbiranja določenih vrst komunalnih odpadkov na območju občine, da mora zagotoviti njihovo odstranitev. Stroške odstranitve odpadkov nosi država, če gre za odpadke na zemljišču v lasti države, ali občina, če gre za odpadke na zemljišču v lasti občine(Bagon, 2023).

## 3.3 ZAKONODAJA

# 4 SPECIFIKACIJA ZAHTEV

To poglavje bo vsebovalo zelo podroben opis delovanja aplikacije. To pomeni vloge pri uporabi aplikacije, funkcionalne zahteve, nefunkcionalne zahteve in opis vmesnikov.

## 4.1 UPORABNIŠKE VLOGE

*Gost* – Lahko se registrira ali pa preprosto uporablja oziroma vstopi v aplikacijo brez prijave. Ob vstopu v aplikacijo mu je na voljo začetna stran na kateri je zemljevid z črnimi odlagališči. Na zemljevidu si za več informacij lahko izbere odlagališče in te informacije bodo prikazane. Na voljo mu je tudi gumb za pomoč uporabnikom sistema, kjer lahko prebere vsa potrebna navodila za uporabo sistema.

*Registriran uporabnik* – Ima možnost prijave v sistem. Ob vstopu v sistem mu je na voljo začetna stran, kjer na zemljevidu vidi označena divja odlagališča. Če izbere neko odlagališče lahko vidi podatke in komentarje drugih uporabnikov, če seveda obstajajo. Na voljo mu je tudi gumb za pomoč, kjer lahko prebere navodila za uporabo sistema. Lahko prijavlja neprimerne komentarje in prijavlja nova odlagališča. Prav tako pa lahko ureja svoj profil, to pomeni, da lahko spreminja uporabniško ime in geslo.

*Administrator* – Ima možnost prijave v sistem. Ob vstopu v sistem mu je na voljo začetna stran, če izbere neko lokacijo na zemljevidu se mu prikažejo podatki in komentarji za tisto odlagališče. Odlagališče lahko označi kot očiščeno oziroma sanirano. Lahko tudi briše komentarje in ureja svoj profil. Kot obvestilo prejme prijave komentarjev uporabnikov in pa nove prijave divjih odlagališč, ki jih lahko potrdi ali pa zavrne, če je prijava lažna.

## 4.2 FUNKCIONALNE ZAHTEVE

*Registracija uporabnika v sistem(F1)* – Gost se lahko registrira v sistem.

Osnovni tok:

1. Gost izbere funkcionalnost Registracija uporabnika v sistem s klikom na gumb Registracija.
2. Sistem prikaže vnosna polja v katera gost vpiše podatke.
3. Gost izpolni vnosna polja: Uporabniško ime, Elektronska pošta, Geslo in Potrditev gesla
4. Gost pritisne gumb Registracija
5. Sistem prikaže sporočilo o uspešni registraciji

Izjemni tok(ovi):

Izjemni tok 1 – Gost se ne more registrirati, ker je naslov elektronske pošte že v uporabi.

Izjemni tok 2 – Gost se ne more registrirati, ker je vnesel neustrezno geslo.

Izjemni tok 3 – Gost se ne more registrirati, ker je ponovljeno geslo nepravilno.

Izjemni tok 4 – Gost se ne more registrirati, ker ni rešil reCAPTCHE.

Posledice – Gostu se ustvari uporabniški račun. ReCAPTCHA pomaga pri zaščiti spletne strani pred spamom in zlorabo.

Prioriteta funkcionalnosti – *Must have*

*Prijava v sistem(F2)* – Administrator ali registriran uporabnik se lahko prijavita v sistem.

Osnovni tok:

1. Administrator ali registriran uporabnik klikne gumb Prijava.
2. Sistem prikaže vnosna polja, ki jih administrator ali registriran uporabnik izpolni.
3. Administrator ali registriran uporabnik izpolni vnosna polja: Elektronska pošta in geslo
4. Administrator ali registriran uporabnik klikne Prijava
5. Sistem administratorja ali registriranega uporabnika preusmeri na domačo stran.

Izjemni tok(ovi):

Izjemni tok 1 – Administrator ali registriran uporabnik se ne more prijaviti, ker je vnesel napačen elektronski naslov.

Izjemni tok 2 – Administrator ali registriran uporabnik se ne more prijaviti, ker je vnesel napačno geslo.

Izjemni tok 3 – Administrator ali registriran uporabnik se ne more prijaviti, ker je dvakrat vnesel napačno geslo ali napačen elektronski naslov, zato se ob naslednji prijavi pojavi reCAPTCHA, ki jo je potrebno rešiti, za ponovno možnost prijave.

Posledice – Administrator ali registriran uporabnik je prijavljen v sistem. V izjemnem toku 3 uporabnik ali administrator mora rešiti reCAPTCHO, to pomaga pri zaščiti spletne strani.

Prioriteta funkcionalnosti – *Must have*

*Ogled podrobnosti divjega odlagališča na zemljevidu(F3)* – Administrator, registriran uporabnik ali gost si izberejo divje odlagališče na zemljevidu in si ogledajo podrobnosti.

Predpostavimo, da sta administrator in registriran uporabnik že prijavljena v sistem.

Osnovni tok:

1. Gost, administrator ali registriran uporabnik, izberejo divje odlagališče na zemljevidu
2. Prikažejo se podatki s podrobnostmi o divjem odlagališču. Gost in registiran uporabnik vidi le podatke o odlagališču, administrator pa vidi tudi gumba, ki omogočata izbris in urejanje divjega odlagališča.
3. Prioriteta funkcionalnosti – *Must have*

*Komentiranje divjih odlagališč(F4)* – registriran uporabnik ali administrator lahko komentira pod podrobnostmi oziroma informacijami o divjem odlagališču.

Osnovni tok:

1. Prijava v sistem
2. Izbira divjega odlagališča s klikom na lokacijo na zemljevidu.
3. Sistem prikaže informacije o odlagališču pod katerimi so tudi komentarji
4. Registriran uporabnik ali administrator vneseta komentar v besedilno polje
5. Registriran uporabnik ali administrator pritisneta gumb Komentiraj.

Posledice – Komentar je objavljen.

Prioriteta funkcionalnosti – *Wont't have*

*Brisanje komentarjev divjih odlagališč(F5)* – Administrator lahko briše vse komentarje, registriran uporabnik pa le svoje komentarje.

Osnovni tok:

1. Prijava v sistem
2. Izbira divjega odlagališča s klikom na lokacijo na zemljevidu
3. Sistem prikaže informacije o odlagališču pod katerimi so tudi komentarji
4. Administrator klikne gumb za izbris pri katerem koli komentarju, registriran uporabnik pa le pri svojih komentarjih.
5. Sistem vpraša administratorja ali registriranega uporabnika, če res želi izbrisati komentar.
6. Administrator ali registriran uporabnik izbereta DA gumb.

Alternativni tok(ovi):

1. Prijava v sistem
2. Izbira divjega odlagališča s klikom na lokacijo na zemljevidu
3. Sistem prikaže informacije o odlagališču pod katerimi so tudi komentarji
4. Administrator klikne gumb za izbris pri katerem koli komentarju, registriran uporabnik pa le pri svojih komentarjih.
5. Sistem vpraša administratorja ali registriranega uporabnika, če res želi izbrisati komentar.
6. Administrator ali registriran uporabnik izbereta NE gumb.

Posledice – komentar je izbrisan pri osnovnem toku, pri alternativnem toku pa ne.

Prioriteta funkcionalnosti – *Won't have*

*Prijava neprimernega komentarja(F6)* – Registriran uporabnik lahko prijavi komentar drugih uporabnikov.

Osnovni tok:

1. Prijava v sistem
2. Izbira divjega odlagališča s klikom na lokacijo na zemljevidu
3. Sistem prikaže informacije o odlagališču pod katerimi so tudi komentarji
4. Registriran uporabnik na gumb za prijavo neželenega komentarja pri katerem koli komentarju razen pri svojih komentarjih.
5. Sistem vpraša registriranega uporabnika, če res želi prijaviti komentar.
6. Registriran uporabnik izbere DA gumb.

Alternativni tok(ovi):

1. Prijava v sistem
2. Izbira divjega odlagališča s klikom na lokacijo na zemljevidu
3. Sistem prikaže informacije o odlagališču pod katerimi so tudi komentarji
4. Registriran uporabnik na gumb za prijavo neželenega komentarja pri katerem koli komentarju razen pri svojih komentarjih.
5. Sistem vpraša registriranega uporabnika, če res želi prijaviti komentar.
6. Registriran uporabnik izbere Ne gumb.

Posledice – Administrator dobi obvestilo o neprimernem komentarju pri osnovnem toku.

Prioriteta funkcionalnosti – *Won't have*

*Ogled prijavljenega komentarja(F7)* – Administrator pogleda prijavljen komentar in se odloči ali je neprimeren.

Osnovni tok:

1. Prijava v sistem.
2. Klik na gumb obvestila.
3. Sistem prikaže obvestila.
4. Izbira obvestila za neprimeren komentar.
5. Sistem prikaže Informacije o divjem odlagališču in prijavljen komentar.
6. Komentar je primeren, klik na gumb ignoriraj.

Alternativni tok(ovi):

1. Prijava v sistem.
2. Klik na gumb obvestila.
3. Sistem prikaže obvestila.
4. Izbira obvestila za neprimeren komentar.
5. Sistem prikaže Informacije o divjem odlagališču in prijavljen komentar.
6. Komentar je neprimeren, klik na gumb zbriši.

Posledice – Pri osnovnem toku komentar ni izbrisan in prijava je odstranjena iz obvestil administratorja. Pri alternativnem toku pa je komentar izbrisan in prijava je odstranjena iz obvestil.

Prioriteta funkcionalnosti –*Won't have*

*Pomoč uporabnikom pri uporabi sistema(F8)* – Gost, registriran uporabnik ali administrator lahko preberejo navodila, ki opisujejo kako aplikacija deluje oziroma kako uporabljati aplikacijo.

Osnovni tok:

1. Gost, registriran uporabnik ali administrator vstopijo na spleto stran
2. Gost, registriran uporabnik ali administrator kliknejo na gumb pomoč
3. Sistem jim prikaže navodila za uporabo aplikacije

Prioriteta funkcionalnosti – *Could have*

*Spreminjanje podatkov svojega profila(F9)* – Registriran uporabnik ali administrator lahko spreminjata, geslo ali elektronski naslov svojega profila.

Osnovni tok:

1. Prijava v sistem
2. Registriran uporabnik ali administrator klikneta gumb uredi profil
3. Sistem prikaže polja za vnos podatkov, tako kot pri registraciji samo, da so že zapolnjena s podatki računa.
4. Registriran uporabnik ali administrator klikneta gumb shrani podatke.

Posledice – podatki registriranega uporabnika ali administratorja so posodobljeni.

Prioriteta funkcionalnosti – *Won't have*

*Dodajanje divjega odlagališča(F10)* – Registriran uporabnik ali administrator lahko dodata oziroma prijavita divje odlagališče v bazo podatkov.

Osnovni tok:

1. Prijava v sistem.
2. Klik na gumb dodaj odlagališče.
3. Izbira lokacije odlagališča na zemljevidu ali vpis GPS koordinat v obrazec za dodajanje.
4. Vpis podatkov o odlagališču v obrazec, opcijsko lahko naložimo tudi slike
5. Klik na gumb dodaj odlagališče.
6. Sistem nas vrne na domačo stran.

Alternativni tok(ovi):

1. Prijava v sistem.
2. Klik na gumb dodaj odlagališče.
3. Izbira lokacije odlagališča na zemljevidu ali vpis GPS koordinat v obrazec za dodajanje.
4. Vpis podatkov o odlagališču v obrazec, opcijsko lahko naložimo tudi slike
5. Klik na gumb prekliči.
6. Sistem nas vrne na domačo stran.

Posledice – Pri osnovnem toku se v bazo podatkov doda novo divje odlagališče, pri alternativnem toku pa je proces preklican, tako da nas sistem samo vrne nazaj na domačo stran.

Prioriteta funkcionalnosti – *Won't have*

*Sprememba podatkov o divjem odlagališču (F11)* – Administrator lahko spremeni podatke o divjem odlagališču ali njegov status v bazi podatkov(potrjen, nepotrjen, očiščen).

Osnovni tok:

1. Prijava v sistem.
2. Administrator izbere lokacijo na zemljevidu s klikom.
3. Sistem prikaže se podatke o divjem odlagališču
4. Administrator klikne gumb uredi.
5. Administrator lahko sedaj ureja podatke o odlagališču po želji.
6. Administrator klikne gumb posodobi podatke.
7. Sistem vrne administratorja nazaj na domačo stran.

Alternativni tok(ovi):

1. Prijava v sistem.
2. Administrator izbere lokacijo na zemljevidu s klikom.
3. Sistem prikaže se podatke o divjem odlagališču
4. Administrator klikne gumb uredi.
5. Administrator lahko sedaj ureja podatke o odlagališču po želji.
6. Administrator spremeni status odlagališča na potrjen
7. Administrator klikne gumb posodobi podatke.
8. Sistem pošlje anonimno prijavo odlagališča na inšpektorat, administratorja pa vrne nazaj na domačo stran.

Posledice – Podatki v bazi podatkov se posodobijo za tisto določeno lokacijo divjega odlagališča. V alternativnem toku se pošlje še anonimno prijavo divjega odlagališča na inšpektorat.

Prioriteta funkcionalnosti – *Must have*

*Izbris divjega odlagališča(F12)* – Administrator lahko izbriše lokacijo divjega odlagališča iz baze podatkov

Osnovni tok:

1. Prijava v sistem
2. Administrator izbere lokacijo na zemljevidu s klikom.
3. Sistem prikaže podatke o divjem odlagališču.
4. Administrator klikne gumb izbriši.
5. Sistem vpraša uporabnika, če res želi izbrisati divje odlagališče
6. Administrator klikne gumb DA.
7. Sistem vrne Administratorja na domačo stran.

Alternativni tok(ovi):

1. Prijava v sistem
2. Administrator izbere lokacijo na zemljevidu s klikom.
3. Sistem prikaže podatke o divjem odlagališču.
4. Administrator klikne gumb izbriši.
5. Sistem vpraša uporabnika, če res želi izbrisati divje odlagališče.
6. Administrator klikne gumb Ne.
7. Sistem administratorja vrne nazaj na podatke o divjem odlagališču.

Posledice – Pri osnovnem toku administrator izbriše divje odlagališče iz baze podatkov, pri alternativnem toku pa prekliče postopek izbrisa.

Prioriteta funkcionalnosti – *Should have*

*Potrditev prijavljenega divjega odlagališča(F13*) – Administrator spremeni status odlagališča iz nepotrjenega v potrjenega, ko se to zgodi aplikacija obvesti pristojne službe.

Osnovni tok:

1. Administrator klikne gumb Prijava.
2. Sistem prikaže vnosna polja, ki jih administrator ali registriran uporabnik izpolni.
3. Administrator izbere nepotrjeno odlagališče na zemljevidu.
4. Sistem prikaže podatke o odlagališču
5. Administrator pritisne gumb uredi.
6. Sistem prikaže obrazec za urejanje podatkov o divjih odlagališčih.
7. Administrator spremeni podatek iz nepotrjenega v potrjeno odlagališče.
8. Administrator klikne gumb Posodobi podatke.
9. Sistem posodobi podatke in osveži odlagališča.

Posledice – Status odlagališča se spremeni in služba, ki je pristojna za divja odlagališča dobi elektronsko sporočilo z obvestilom o novem divjem odlagališču in z linkom do spletne strani.

Prioriteta funkcionalnosti – *Won't have*

## 4.3 NEFUNKCIONALNE ZAHTEVE

Zahteve izdelka:

* Sistem mora biti dosegljiv na javno dostopnem spletnem naslovu.
* Sistem mora na vsako poizvedo odgovoriti v največ 2s.
* Sistem mora biti na voljo najmanj 99,9 odstotkov časa.
* Sistem mora biti zmožen streči najmanj 1000 hkratnim uporabnikom.
* 95% odzivnih časov za določen vnos za iskanje v zbirki podatkov mora biti ≤5s za najvišjo delovno obremenitev med normalnim delovanjem.

Organizacijske zahteve:

* Transakcijska podatkovna baza naj uporablja jezik SQL.
* Spletna aplikacija mora do podatkov iz baze dostopati preko REST Api-ja.

Zunanje zahteve:

* Sistem mora delati na brskalnikih: Google chrome, Opera, Firefox.

## 4.4 SLOVAR POJMOV

Uporabniška vloga - uporabnik lahko v sistemu zaseda različne vloge, ki mu omogočajo različne funkcionalnosti oziroma početje v sistemu.

Prijava - prijava v sistem s elektronsko pošto in geslom, ki ga je uporabnik izbral.

Registracija - registracija v sistem kjer uporabnik poda nekaj svojih osebnih podatkov svojo elektronsko pošto in vnese izbrano geslo.

reCAPTCHA - Googlova brezplačna storitev, ki pomaga zaščititi spletna meta pred neželeno pošto in zlorabo.

GPS - Global Positioning System(sistem globalnega pozicioniranja) je globalni satelitski navigacijski sistem, ki zagotavlja sinhronizacijo lokacije, hitrosti in časa. GPS sisteme lahko najdemo povsod, npr. v avtu, pametnem telefonu ali uri.

Transakcijska podatkovna baza – podatkovna baza za hitro iskanje podatkov.

Administrator - skrbnik sistema, ki lahko dodeluje moderatorske pravice in odvzame dostop uporabniku.

Odzivni čas - čas od začetka akcije do prikaza nekih rezultatov.

Spam – Pošiljanje velikega števila ukazov na spletni strani s pomočjo avtomatskih sistemov.

## 4.5 DIAGRAM PRIMEROV UPORABE

**Slika 1: UML diagram za aplikacijo**

Slika, ki vsebuje besede besedilo, posnetek zaslona, diagram, vrstica

Opis je samodejno ustvarjen

Diagram primerov uporabe nam prikazuje vse prej naštete funkcionalnosti, in kdo jih lahko uporablja. Na levi strani vidimo vloge uporabnikov, na desni strani pa so zunanji sistemi, s katerimi komunicira aplikacija. UML diagram je bil izdelan s pomočjo programske opreme PowerDesigner.

PowerDesigner je celovito orodje za modeliranje in upravljanje z informacijskimi sistemi ter poslovnimi procesi. Gre za napredno programsko rešitev, ki omogoča organizacijam načrtovanje, oblikovanje in upravljanje različnih vidikov njihove informacijske infrastrukture. Z njegovo pomočjo lahko organizacije ustvarjajo grafične modele podatkovnih baz, aplikacijskih sistemov, poslovnih procesov in še več. Omogoča vizualizacijo različnih komponent in njihovih odnosov ter s tem olajšuje analizo, načrtovanje in razvoj projektov. Prednost PowerDesignerja je njegova sposobnost za modeliranje na več ravneh, kar omogoča boljše razumevanje kompleksnih sistemov in zagotavlja usklajenost med različnimi deležniki. Poleg tega orodje omogoča sledenje spremembam, upravljanje verzij in izmenjavo informacij med različnimi ekipami. PowerDesigner je ključno orodje za podporo strategijam upravljanja z informacijskimi viri ter omogoča boljše načrtovanje, usklajevanje in izvajanje projektov v organizacijah.

## 4.6 OPIS VMESNIKOV

Zaslonske maske so bile izdelane v programu Adobe XD. Ta pa je profesionalno oblikovalsko orodje, ki je namenjeno ustvarjanju interaktivnih in privlačnih uporabniških vmesnikov ter izkušenj. Gre za programsko rešitev, ki je posebej zasnovana za oblikovalce spletnih strani, mobilnih aplikacij in drugih digitalnih vmesnikov. Z Adobe XD lahko oblikovalci ustvarjajo vizualne načrte, interaktivne prototipe in animacije, ki omogočajo realističen vpogled v to, kako se bo uporabnik premikal po aplikaciji ali spletni strani. Orodje vključuje številne funkcionalnosti, kot so možnosti za sodelovalno delo med člani ekipe, ustvarjanje ponovne uporabe elementov, združevanje oblikovalskih in razvojnih procesov ter testiranje uporabniških izkušenj pred dejansko implementacijo. S poudarkom na intuitivnosti in preprostosti uporabe omogoča Adobe XD oblikovalcem, da hitro preizkušajo ideje, optimizirajo uporabniške izkušnje ter prispevajo k hitrejšemu in učinkovitejšemu razvoju digitalnih rešitev.

**Slika 2: Zaslonska maska-prikaz divjih odlagališč**

Slika, ki vsebuje besede zemljevid, besedilo, atlas, posnetek zaslona

Opis je samodejno ustvarjen

Prva zaslonska maska nam prikazuje stran, ki je uporabniku prikazana ob vstopu na spletno stran. Tu lahko uporablja filtre(desno spodaj), da prikaže različne vrste divjih odlagališč, lahko se prijavi ali registrira, če si želi dodati oziroma prijaviti novo divje odlagališče. Zemljevid seveda lahko premika, približa in oddalji po želji. Prav tako lahko klikne na eno od lokacij na zemljevidu, za prikaz podrobnosti o izbranem divjem odlagališču. Ob registraciji in prijavo je na voljo tudi gumb za pomoč, ki pa prikaže podrobna navodila, kako prijaviti novo divje odlagališče.

**Slika 3: Zaslonska maska-prikaz podatkov o izbranem divjem odlagališču**

Slika, ki vsebuje besede besedilo, zemljevid, posnetek zaslona, atlas

Opis je samodejno ustvarjen

Na tej zaslonski maski pa je prikazan prikaz podatkov o izbranem odlagališču, uporabnik ima poleg prejšnjih možnosti tudi možnost povečave slik s klikom na le te. Prav tako lahko uporabnik poljubno izbira druga divja odlagališča na zemljevidu, brez zapiranja prikaza s podatki.

**Slika 4: Zaslonska maska-prijava uporabnika v sistem**

Slika, ki vsebuje besede besedilo, posnetek zaslona, diagram, zemljevid

Opis je samodejno ustvarjen

Na tej zaslonski maski pa se je uporabnik po ogledu informacij o izbranih odlagališčih odločil za prijavo v sistem. Po kliku na gumb prijava se mu je prikazalo pojavno okno, v katerega mora vnesti podatke za prijavo. Prijavo lahko prekliče ali pa pritisne gumb potrdi, z vnesenimi podatki, če so podatki pravilni, ga bo sistem prijavil v sistem, v nasprotnem primeru pa bo uporabnik obveščen o napaki.

# 5 ZASNOVA INFORMACIJSKE REŠITVE ZA PROCES PRIJAVE DIVJEGA ODLAGALIŠČA

V tem poglavju bo procesni model, ki nam prikazuje celotni potek prijave divjega odlagališča v novem sistemu in obrazložitev izbire orodij, jezikov in sistemov za izdelavo informacijske rešitve

## 5.1 PROCESNI MODEL PRIJAVE DIVJEGA ODLAGALIŠČA V NOVEM SISTEMU

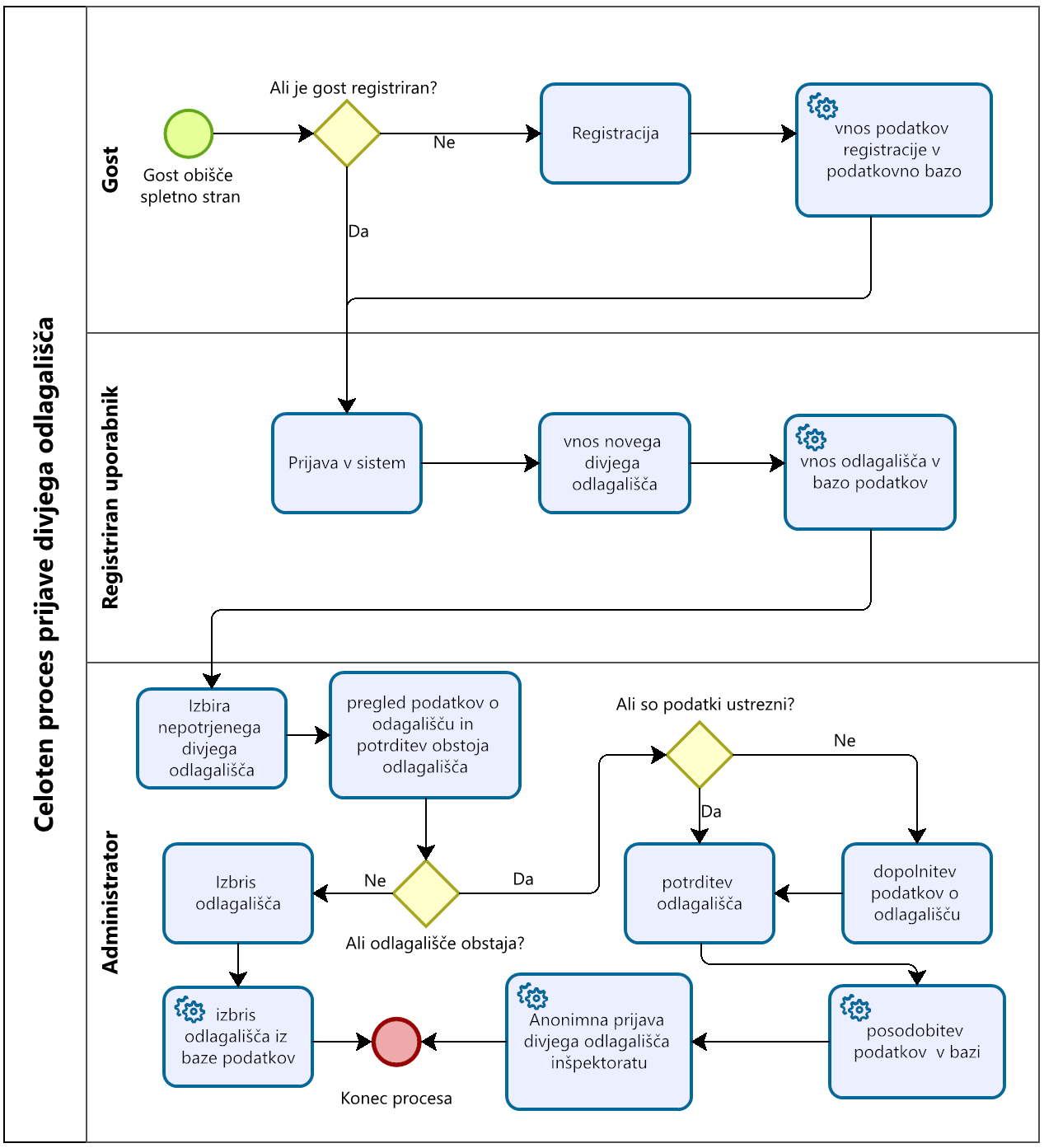
Procesni model je ustvarjen z uporabo programa Bizagi Modeler, ki uporablja standardizirano notacijo BPMN 2.0 in je splošno razumljiv med strokovnjaki. Model ima nekaj posebnosti in sicer:

1. Organizacijske enote predstavljajo vloge uporabnika v sistemu(gost in registriran uporabnik sta v procesu prijave odlagališča enaka oseba, imata samo drugo vlogo).
2. Splošne aktivnosti(tiste, ki niso označene) so tiste, ki jih uporabnik izvaja ročno

**Tabela 1: legenda simbolov procesnega modela**

|  |  |
| --- | --- |
| Začetni dogodek |  |
| Končni dogodek |  |
| Odločitev ali razvejišče |  |
| Splošna aktivnost |  |
| Atomizirana aktivnost s pomočjo informacijskega sistema |  |
| Potek izvajanja |  |

**Slika 5: Celoten proces prijave divjega odlagališča**



## 5.2 PODATKOVNA BAZA

Za podatkovno bazo sem izbral PostgreSQL iz več razlogov. PostgreSQL je zmogljiva, odprtokodna relacijska podatkovna baza, ki ponuja širok nabor funkcionalnosti za upravljanje in poizvedovanje podatkov. Podpira kompleksne poizvedbe, indeksiranje ter transakcijsko in večuporabniško delovanje, kar je ključno pri obdelavi in analizi geografskih podatkov.

Ima vgrajeno podporo za geografske podatke prek paketa PostGIS. Ta pa je razširitev za PostgreSQL, ki omogoča upravljanje in analizo geografskih podatkov, vključno z vektorji in rastri. PostGIS dodaja podporo za geografske objekte in omogoča izvajanje prostorskih poizvedb ter analiz nad temi podatki. Z njem lahko izvajamo operacije, kot so presečišča, združevanja, ločitven območja, analiza bližine in veliko več.

PostGIS ponuja tudi različne funkcije in operacije za obdelavo in analizo prostorskih podatkov, kot so izračun razdalj, iskanje najbližjih objektov, določanje površine in obsega, preoblikovanje koordinatnih sistemov ter interakcija z drugimi geografskimi podatki. Prav tako podpira uvoz in izvoz standardnih geografskih formatov, kar olajšuje izmenjavo podatkov med različnimi sistemskimi orodji.

PostgreSQL s PostGIS-om je dobra izbira za upravljanje, analizo in obdelavo geografskih podatkov. Razvijalcem in analitikom omogoča, da izkoristijo moč relacijske podatkovne baze in prostorskih operacij za reševanje kompleksnih izzivov v geografskem okolju.

## 5.3 ANGULAR(FRONTEND)

Angular je ogrodje s katerim razvijamo frontend spletne aplikacije. Uporablja komponento arhitekturo, kar pomeni, da je aplikacija razdeljena na manjše in samostojne komponente. To nam omogoči boljšo organizacijo in ponovno uporabo kode. Komponente imajo svojo logiko in uporabniški vmesnik, kar olajša vzdrževanje in nadgrajevanje aplikacije.

Ponuja nam tudi veliko orodij in funkcionalnosti, ki pomagajo pri razvoju aplikacije(Angular Material, Anuglar forms). Ima močno podporo za testiranje, kar mi bo olajšalo testiranje aplikacije in pa veliko uporabnikov in razvijalec, kar olajša reševanje težav in iskanje rešitev med razvojem.

Podpira tudi razvoj večplatformnih aplikacij, kar pomeni možnost izvajanja aplikacije na strani strežnika in na strani odjemalca, to pa posledično izboljša hitrost nalaganja aplikacije.

## 5.4 SPRING BOOT(BACKEND)

Spring Boot je ogrodje za razvoj backend-a. Izbral sem ga zaradi svoje enostavne uporabe in hitrosti razvoja. Zagotavlja priročne privzete vrednosti in samodejno konfiguracijo, kar olajša začetek razvoja projekta. Sledi modularen pristop, kar omogoča razvijalcem, da izberejo samo tiste module in knjižnice, ki jih potrebujejo, kar poenostavlja konfiguracijo projekta in zmanjšuje nepotrebno kompleksnost.

Prav tako avtomatsko konfigurira veliko aspektov projekta, kar zmanjšuje potrebo po ročnem konfiguriranju. Poleg tega ima obsežno skupnost razvijalcev, ki ponuja podporo, dokumentacijo in odgovore na vprašanja, kar olajšuje reševanje težav med razvojem.

Spring Boot omogoča enostavno razvojno okolje za REST API-je. REST API-ji so priljubljen način za komunikacijo med backend-om in frontend-om prek standardnih HTTP metod, kot so GET, POST, PUT in DELETE. Ponuja tudi integracijo za ustvarjanje RESTful storitev, kar olajša razvoj API-jev za komunikacijo s spletnimi aplikacijami in pa integracijo s podatkovnimi bazami prek modula Spring Data JPA, kar omogoča enostavno upravljanje in izvajanje poizvedb nad podatkovno bazo.

# 6 RAZVOJ SPLETNE APLIKACIJE

V tem poglavju bom predstavil potek razvoja spletne aplikacije in ključne elemente v kodi in sestavi spletne aplikacije.

## 6.1 PRIDOBITEV PODATKOV IN VZPOSTAVITEV PODATKOVNE BAZE

V pogovoru s neprofitno organizacijo Ekologi brez meja, ki trenutno upravljajo z največjo zbirko divjih odlagališč in podatkov o le teh v Sloveniji, sem izvedel da lahko od njih dobim izvožene podatke o iz njihovega registra in jih uporabim v svoji diplomski nalogi. Podatke sem pridobil brez slik zaradi lažjega izvoza in sicer kot GPKG datoteko.

Datoteka GPKG(GeoPackage) je odprt standardiziran format podatkov za geografske informacijske sisteme, zgrajen na osnovi konvencij nad podatkovno bazo SQLite. Določen s strani organizacije Open Geospatial Consortium (OGC)in z vzpodbudo ameriške vojske je leta 2014, GeoPackage prejel obsežno podporo različnih vladnih, komercialnih in odprtokodnih organizacij. Nastal je saj kljub desetinam različnih formatov datotek in storitev za izmenjavo geoprostorskih podatkov, ni obstajal odprt format, ki bi lahko podpiral vektorske in rasterske podatke ter bil učinkovito dekodiran s programsko opremo. To potrebo so leta 2012 izrazili pri organizaciji OGC, standard pa je bil odobren februarja 2014.

To GPKG datoteko sem moral prebrati in uvoziti v mojo PostgreSQL podatkovno bazo, da sem to dosegel pa sem potreboval kar nekaj pripomočkov in knjižnic. To so GDAL (Geospatial Data Abstraction Library), ki omogoča obdelavo in manipulacijo z geoprostorskimi podatki, python in pa knjižnica 'psycopg2', ki sta uporabljena za povezavo z PostgreSQL podatkovno bazo. Vse te knjižnice sem pridobil z namestitvijo programa OSGeo4W(Open Source Geospatial Foundation for Windows). Ta program je zbirka odprtokodnih programov in knjižnic, ki uporabniku omogočijo enostaven dostop do številnih orodij za obdelavo, analizo in vizualizacijo geoprostrskih podatkov. Po namestitvi programa sem odprl svoj ukazni poziv in z naslednjim ukazom uvozil podatke o divjih odlagališčih v svojo bazo podatkov:

ogr2ogr -f PostgreSQL PG:"dbname=postgres user=postgres password=2131" pot do datoteke GPKG -lco SCHEMA=dataOdlagališča -lco OVERWRITE=YES

* ogr2ogr: To je ukazna vrstica za orodje, ki se uporablja za konverzijo med različnimi geoprostorskimi formati.
* -f PostgreSQL: Ta del ukaza določa izhodni format, v tem primeru je to podatkovna baza PostgreSQL.
* PG:"dbname=postgres user=postgres password=2131": Določa povezavo s PostgreSQL podatkovno bazo, vključno z imenom baze (dbname), uporabniškim imenom (user) in geslom (password).
* pot do datoteke GPKG: To je pot do vhodne Geopackage datoteke, ki sem jo želel uvoziti.
* -lco SCHEMA=dataOdlagališča: Ta del ukaza ustvari novo shemo (dataOdlagališča) v PostgreSQL bazi, kamor bodo uvoženi podatki.
* -lco OVERWRITE=YES: Ta del ukaza določa, da se obstoječe tabele v PostgreSQL bazi, če že obstajajo, prepišejo.

S tem je bil uvoz podatkov v podatkovno bazo uspešno zaključen. V podatkovni bazi sem ustvaril še tabelo uporabniki, v kateri se hranijo vrednosti id, elektronski naslov, geslo in administratorski status uporabnika. Status uporabnika nam pove ali ima uporabnik administratorske pravice.

## 6.2 PRIKAZ LOKACIJ DIVJIH ODLAGALIŠČ NA ZEMLJEVIDU

Podatke sem iz podatkovne baze moral spraviti na frontend(Angular) moje spletne aplikacije. To pa sem dosegel s pomočjo backend-a(Spring Boot), saj backend spletne aplikacija poskrbi za komunikacijo med podatkovno bazo in frontend-om. Prvi korak je bila kreacija novega razreda entitete, ker ta omogoča povezavo med Java objekti in podatkovno bazo. Razred entitete predstavlja tabelo v podatkovni bazi, kjer vsaka lastnost ustreza stolpcu v tabeli. Na sliki pod odstavkom se lahko vidi, da pred razredom definiram ime tabele v podatkovni bazi nato pa za vsak atribut v tabeli, povem ime in mu določim vrednost v java razredu.

**Slika 6: Začetek razreda Odlagališča**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Nato pa sem ustvaril še prazen konstruktor in pa konstruktor s parametri. Prazen konstruktor sem ustvaril, ker tako zahteva Hibernate(ogrodje za komunikacijo z podatkovnimi bazami). Hibernate med pridobivanjem podatkov iz podatkovne baze najprej ustvari prazen objekt prek praznega konstruktorja nato pa s pomočjo setterjev nastavi vrednosti iz rezultatov poizvedb. Konstruktor s parametri pa je potreben, da lahko inicializiramo objekt z vrednostmi, ki jih prejmemo. Oba konstruktorja sta potrebna za pravilno delovanje Hibernate-a pri preslikavi objektov iz in v podatkovno bazo. Nato sem za vse elemente ustvaril še getterje in setterje za lastno uporabo.

Drugi korak je bila kreacija razreda repozitorija, ki je pomemben za komunikacijo med backend-om in podatkovno bazo. Namen razreda je omogočiti enostavno izvajanje operacij za branje, ustvarjanje, posodabljanje in brisanje podatkov v podatkovni bazi. Razred deluje kot vmesnik med Spring Boot aplikacijo in JPA(Java Persistence API). V razred lahko dodamo tudi lastne poizvedbe, če standardne metode, ki so samodejno ustvarjene, ne zadostujejo pri komunikaciji z podatkovno bazo.

Tretji korak je bila kreacija razreda storitve, ki je namenjen izvajanju poslovne logike in upravljanju z obdelavo podatkov med krmilniki in repozitoriji. Glavna vloga razreda storitve je omogočiti ločitev poslovne logike od sloja za dostop do podatkov in st tem doseči boljšo modularnost, vzdrževanje in prenosljivost kode(koda je zaradi tega razreda bolj pregledna).

V tem razredu sem izvedel pretvorbo niza WKB(Well-known Binary) v niz WKT(Well-known text). Najprej sem pridobil seznam odlagališč iz repozitorija, nato sem za vsako odlagališče pridobil niz WKB iz geometrijskega atributa. S pomočjo pomožnih metod sem niz WKB pretvoril v bajtno obliko, potem pa sem uporabil knjižnico JTS(Java Topology Suite) za pretvorbo bajtov v geometrijski objekt. Zatem sem ta objekt z knjižnico pretvoril v niz WKT in ga nastavil nazaj kot vrednost za atribut geometrije. S to pretvorbo sem omogočil, da se geometrijski podatki iz baze prevedejo v berljivo obliko, kar olajša nadaljnjo uporabo in prikaz teh podatkov.

Četrti in zadnji korak na backend-u pa je bila kreacija krmilnika, ki ima ključno vlogo pri vzpostavljanju povezave med uporabniškim vmesnikom(frontend) in poslovnim slojem aplikacije. Namenjen je sprejemanju zahtev HTTP od uporabnikov ali drugih sistemov ter usmerjanju teh zahtev v ustrezne obdelovalne enote znotraj aplikacije. Ko krmilnik prejme zahtevo, lahko izvede potrebne ukrepe, sproži poslovno logiko prek storitev in pripravi podatke za pripravi podatke za prikazovanje uporabniku. Po prejeti zahtevi vrne odgovor v obliki http odgovora, ki vključuje podatke, odgovore ali napake.

V krmilnik za pridobitev seznama vseh odlagališč samo kliče funkcijo 'getOdlagalisca()' iz storitvenega razreda. Ob prejemu zahteve GET na pravilni poti od krmilnika, ta preprosto pokliče metodo 'getOdlagalisca()' in vrne seznam vseh odlagališč. S tem krmilnik omogoča pridobivanje podatkov o odlagališčih prek določene API poti, ki je lahko uporabljena za izgradnjo uporabniškega vmesnika.

Peti korak pa je bila izdelava frontend-a in seveda strani na kateri se bodo prikazale lokacije divjih odlagališč. Najprej sem želel, da se mi na strani prikaže zemljevid, to sem dosegel tako, da sem uvozil in uporabil Google Maps API in inicializiral Google zemljevid na komponenti.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, vrstica

Opis je samodejno ustvarjen

Na zgornji sliki se vidi kako se inicializira Google zemljevid. S pomočjo objekta 'mapOptions' se nastavijo začetne vrednosti zemljevida, kot so središče(center), začetna povečava in pa tip zemljevida, ki je v tem primeru 'ROADMAP'. Nato se s pomočjo teh nastavitev in kontejnerja za zemljevid ustvari nova instanca Google zemljevida.

Potem je potrebno ta kontejner dodati še v datoteko html in poklicati funkcijo 'loadMap()' ob inicializaciji spletne strani. Zemljevid se potem prikaže na spletni strani in s pomočjo css-a lahko oblikujem in določamo velikost kontejnerja.

Na zemljevidu pa je seveda potrebno prikazati vsa divja odlagališča, tako da je potrebno pridobiti podatke o divjih odlagališčih iz baze, te podatke obdelati in nato s pomočjo pridobljenih koordinat prikazati na zemljevidu. Najprej sem ustvaril nov TypeScript razred 'DataService', ki je storitveni razred uporabljen za komunikacijo s strežnikom prek HTTP zahtev.

**Slika 7: Koda za HTTP zahtevo**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, vrstica

Opis je samodejno ustvarjen

Na zgornji sliki vidimo kako komuniciramo z strežnikom v storitvenem razredu, 'baseUrl' predstavlja osnovni naslov strežnika(backenda), metoda 'getOdlagalisca()' pa uporablja 'HttpClient' za izvajanje GET zahteve na osnovnem naslovu, ki mu dodamo '/odlagalisca'.

Za prikaz odlagališč sem napisal štiri funkcije. Prva 'fetchOdlagalisca' uporabi zgornjo storitev, da pridobi podatke o odlagališčih preko asinhronega klica. Ko so podatki pridobljeni se v funkciji kliče funkcija 'displayOdlagalisca' s pridobljenimi podatki kot argument.

Funkcija 'displayOdlagalisca' poskrbi za prikaz odlagališč na zemljevidu. Najprej prečisti že obstoječe označevalce(markerje), tako da za vsak označevalec pokliče metodo 'setMap(null)', kar povzroči izbris označevalca iz zemljevida. Ponastavi tudi seznam označevalcev, ki so shranjeni v spremenljivki. Nato gre funkcija skozi seznam podanih odlagališč, ki je bil prejet kot argument in za vsako odlagališče prebere geometrijski zapis in ga pretvori v koordinate z metodo 'parseGeometryString'. S temi koordinatami in z objektom Odlagalisce se nato kliče funkcijo 'addMarker', ki kot argumente prejeme objekt Odlagalisce in pretvorjene koordinate. To kar funkcija vrne se nato shrani v seznam označevalcev, ki je bil na začetku spraznjen. Funkcija počisti označevalce in jih ponovno shrani v seznam, da se lahko kliče tudi kot osvežitev zemljevida po spremembah.

Funkcija 'parseGeometryString' sprejme geometrijski zapis v obliki niza in ga analizira ter pretvori v koordinate zemljepisne širine (latitude) in dolžine (longitude). Uporablja regularni izraz, da išče ujemanje med vzorcem 'MULTIPOINT (([širina] [dolžina]))'. Če najde ujemanje funkcija pridobi vrednosti širine in dolžine ter jih pretvori v števila.

Funkcija 'addMarker' sprejme podatke o odlagališču in koordinatah ter na podlagi teh podatkov dodaja označevalce na zemljevidu. Glede na lastnosti odlagališča, se določi, kakšna ikona bo uporabljena za označevalec. Poleg tega se doda poslušalec dogodkov za klik na označevalcu, ki sproži funkcijo 'onMarkerClick', funkcija na koncu vrne ustvarjeni označevalec. Poslušalec dogodkov pa je dodan na označevalce, da se lahko ob kliku na marker prikažejo podatki o odlagališču.

**Slika 8: Prikaz divjih odlagališč na zemljevidu**

Slika, ki vsebuje besede posnetek zaslona

Opis je samodejno ustvarjen

Na zgornji sliki se lahko vidijo različne vrste označb divjih odlagališč, glede na njihove podatke. Z drevesi so označena počiščena divja odlagališča, z kupom smeti navadna divja odlagališča, z kupom smeti in rdečim trikotnikom pa so označena divja odlagališča, ki vsebujejo nevarne odpadke.

Funkcija 'onMarkerClick' ob kliku na izbrano divje odlagališče na zemljevidu shrani podatke o odlagališču in njegove koordinate v objekt. Ta objekt je poslan v drugo komponento, katere naloga je prikaz podatkov o divjem odlagališču. Prav tako spremeni spremenljivko 'isDataDIsplayVisible', na 'true', kar omogoči prikaz druge komponente.

## 6.3 PRIKAZ PODROBNOSTI IZBRANEGA DIVJEGA ODLAGALIŠČA

Podatki poslani iz komponente kjer je prikazana mapa so prikazani na komponenti za prikaz podrobnosti divjih odlagališč. Ta komponenta se prikaže na levi strani spletne aplikacije, tako da se pojave čez mapo. Med prikazom podrobnosti lahko uporabnik še vedno uporablja mapo in klika na druga divja odlagališča, za prikaz drugih podatkov. Logike za prikaz je dokaj malo, saj je potrebno samo prebrati vsak element objekta, pogledati ali je vrednost ni 'null', in nato prikazati podatke, ki obstajajo. Poleg elementov objekta, pa je potrebno še izpisati koordinate. V primeru, da ima divje odlagališče v podatkih tudi poti to slik, ki so v odlagališču je potrebno prikazati predoglede slik. Ker pa so v objektu shranjene le poti, je potrebno na krmilniku backend-a na teh poteh zagotoviti pravo sliko.

Slike v podatkovni bazi hranim, kot seznam nizov, kjer je vsak niz pot do slike. Slike pa hranim na svoji lokalni napravi, saj imam za trenutne potrebe več kot dovolj prostora na trdem disku svoje naprave. Torej v krmilniku sem naredil novo funkcijo, ki obravnava zahteve za prikaz slik. Ko uporabnik posreduje zahtevo za sliko s podanim imenom datoteke, funkcija konstruira pot do direktorija, kjer so shranjene slike. Tej poti pa se nato doda še ime datoteke. S pomočjo te poti se ustvari nov objekt iz sistema datotek. Funkcija nato preveri, ali datoteka obstaja in ali je berljiva. V primeru, je se vrne odgovor 200(OK), v katerem je vključena slika z ustrezno vsebino, za prikazovanje slike v brskalniku. V nasprotnem primeru pa funkcija odgovor 404(Not Found), kar pomeni, da zahtevane slike ni bilo mogoče najti.

S tem je možen prikaz slik. Pri prikazu podrobnosti se tako pojavijo predogledi slik, ki pa jih lahko kliknemo, ob kliku se kliče funkcija 'openImagePopup', ki prikaže celotno sliko kot pojavno okno. To pojavno okno lahko enostavno zapremo in nadaljujemo z ogledovanjem zemljevida in podrobnosti o izbranem odlagališču.

Če je uporabnik prijavljen v sistem in ima status administratorja se na koncu podatkov o odlagališču pojavita še gumba za izbris in urejanje divjega odlagališča.

**Slika 9: Prikaz podrobnosti o divjih odlagališčih**

Slika, ki vsebuje besede besedilo, posnetek zaslona, rastlina

Opis je samodejno ustvarjen

Na zgornjih slikah lahko vidimo prikaz podrobnih podatkov s slikami in prikaz podatkov brez slik, če je podatkov preveč za izpis je omogočeno drsenje samo v komponenti,ki prikazuje podrobnosti, kar se tudi vidi na sliki, kjer so prikazani podatki divjega odlagališča s slikami.

## 6.4 REGISTRACIJA, PRIJAVA IN POMOČ UPORABNIKU

Dostop do vseh treh funkcionalnosti je možen preko gumba v desnem zgornjem kotu spletne strani in vse tri se prikažejo kot pojavno okno na spletni strani. Gumb pomoč je vedno na spletni strani in njegova edina funkcionalnsot je, da prikaže podrobne navodila za ogled in dodajanje divjih odlagališč.

Za registracijo in prijavo mora uporabnikški vmesnik komunicirati z strežnikom, to pa pomeni, da sem na strežniku moral ponoviti vse korake za povezavo s tabelo v podatkovni bazi, tako kot sem to storil pri priakzu divjih odlagališč. Po kreaciji razredov entitete, repozitorija in storitve sem še v kontroler dodal metode, ki skrbijo za komunikacijo med njima.

Metoda 'checkEmailExists' kot parameter prejme elektronski naslov in preveri ali uporabnik s tem elektronskom naslovom že obstaja. Rezulat pa se vrne kot logična vrednost(true ali false).

Metoda 'registerUser' omogoča registracijo novega uporabnika v aplikaciji. Kot vhodni parameter prejme podakte o uporabniku. Pred shranjevanjem v podatkovno bazo uporabi še funkcijo, ki zašifrira geslo uporabnika za varnostne namene. Nato prejetemu uporabniku dodeli zašifrirano geslo in ga prek storitve shrani v podatkovno bazo.

Metoda 'loginUser' omogoča proces prijave uporabnika v aplikacijo. Kot parameter prejme uporabnika, kar vklučuje elketronski naslov in geslo. Napjprej prever, ali uporabnik z podanim elektronskim naslovom obstaja v podatkovni bazi. Če obstaja funkcija preveri, če sep odano geslo ujema z gelsom v podatkovni bazi, ker pa je geslo v bazi podatkov zašifrirano, moramo pred primerjavo zašifrirati na enak način tudi podano geslo. Ob ujemanju gesla metoda vrne odgovor s podatki o uporabniku(brez gesla) s statusom 200(OK). Če pa je avtentokacija kjerkoli neuspešna pa je metoda vrne odgovor s statusom 401(Unauthorized) in sporočilom, da avtentikacija ni usepla.

Za komunikacijo s temi metodami pa je bilo potrebno ustvariti tudi nov servisni razred na frontendu. V razredu so funkcije za komunikacijo z kotrolerjem backend-a, ki so narejene podoben način kot razred 'DataService'. Vsebuje pa tudi funkcije, ki shranijo prejete podatke o uprabniku in spremenijo status uporabnika, ki uporablja uporabniški vmesnik. Če se uporabnik uspešno prijavi se spremeljikva 'loggedIn' in spremeni na vrednost 'true', če pa se uporabnik odjavi pa se spremeni na 'false'. Razred torej hrani in servisira podatke o uporabniku in o tem ali je uporabnik trenutno prijavljen v sistem.

Za uspešno registracijo mora uporabnik na pojavnem oknu pravilno izpolniti vsa tri besedilna polja in rešiti reCAPTCHO, s čimer se preverja, da je človek. Za uporabo reCAPTCHE se je potrebno pri Googlu prijaviti za par ključev. To sta ključ spletnega mesta(site key) in skrivni ključ(secret key). Ključ spletnega mesta se uporablja za prikilc soritve reCAPTCHA na spletnem mestu, skrivni ključ pa avtorizira komunikacijo med spletnim mestom in reCAPTCHA strežnikom, ki preveri uporabnika. Po pridobitvi teh ključov od Googla je potrebno samo še uvoziti pravo knjižnico in uporabiti reCAPTCHA element ter mu podati pravilni ključ.

**Slika 10: Obrazec za registracijo**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, oblikovanje

Opis je samodejno ustvarjen

Na sliki se vidi pojavno okno registracije, v tem primeru je uporabnik vnesel nepravilen elektronski naslov. Ko so besedilna polja prazna v njih piše kaj mora uporabnik vnesti v le tega.

Ob kliku na gumb registracija se izvede funkcija 'onRegister()', ki najprej preveri ali je vneseni elektronski naslov v veljavnem formatu, v premeru, da ni veljaven se prikaže sporočilo o napaki v elektronskem naslov. Zatem preveri ujemanje vnesenih gesel, čese gesla ne ujemata se prikaže sporočilo o neujemanju gesel. Nato preveri ali je reCAPTCHa rešena in v primeru, da ni vrne sporočilo o napaki. Elektronski naslov nato preveri še z klicom na strežnik prek storitvenega razreda, ki preveri ali je elektronski naslov že v bazi podatkov in glede na odgovor strežnika vrne ustrezen rezultat. V primeru, da je s zahtevkom za registracijo vse vredu, potem se kliče še metoda 'registerUser' iz storitvenega razreda. Po uspešni registracije se pojavno okno zapre.

Prijava je narejena na podoben način, le da ima za vnos le dva besedilna polja. To sta elektronski naslov in geslo. ReCAPTHCO pa je potebno rešiti samo, če je uporabnik že več kot dvakrat neuspešno prijavil.

Ob kliku na gumb prijava, se izvede funkcija 'onLogin()', ki najprej preveri, če je elektronski naslov veljaven, tako kot registracija. Zatem pa se preko storitvenega razreda kliče funkcija za prijavo uporabnika, ki strežniku pošlje elektronski naslov in geslo. Če je prijava uspešna se uporabnik označi kot prijavljen, spremenljivka, ki šteje neuspele prijave se nastavi na nič in pojavno okno se zapre. V primeru neuspele prijave se uporabniku izpiše napaka in spremenljivka ki šteje število neuspelih poskusov priajve si doda en poskus.

Po uspešni prijavi, dobi uporabnik nove možnosti na spletni strani kot so odjava in pa dodajane divjih odlagališč. Ti gumbi se pojavijo tam, kjer sta bila prej gumba za prijavo in registracijo.

## 6.5 DODAJANJE, POSODABLJANJE IN BRISANJE DIVJIH ODLAGALIŠČ

Za implementacijo teh funkcionalnosti sem v kontroler na strežniku moral dodati tri metode, ki skrbijo za komunkiacijo med podatkovno bazo in frontend-om. Poleg tega pa sem v storitveni razred odlagališč dodal še tri nove funkcije, ki skrbijo za dodajanje, izbris in posodabljanje slik(ob posodobitvi odlagališča primerjam imena novih in starih slik ter izbrišem tiste, ki jih v posodobljeni verziji ni več, nove slike pa se shranijo) shranjenih na strežniku.

Metoda 'addNewOdlagalisce' omogoča dodajanje novega divjega odlagališča. Kot argument prejme objekt in pa slike. Najprej se preveri ali je parameter slik prazen ali ne, če ni prazen, se slike shranijo in pridobijo se poti to le teh na strežniku. Te poti slik se nato nastavijo entiteti objekta, če slik ni bil naloženih, se nastavi vredonst poti slik na 'null'. Po pripravah se objekt shrani v bazo podatkov. Za shranjevanje v podatkovno bazo sem moral napisati svojo poizvedbo, saj so koordinate v podatkovni bazi shranjene kot 'geometry'(WKB), kar pa je pomenilo, da poizvedba generirana z JPA ni delovala. Koordinate v WKT formatu sem moral spremeniti v tip, ki ga podatkovna baza spejme. V poizvedbi za vnašanje in posodabanje podatkov sem zato uporabil naslednjo funkcijo 'public.ST\_GeomFromText(:geometry)'(iz paketa postGIS), ki spremeni WKT niz v pravilno obliko za shranjevanje v podatkovni bazi.

Metoda 'deleteOdlagalisce' omogoča izbris odlagališča iz podatkovne baze. Kot parameter prejme id in z njim preveri ali odlagališče obstaja. Če odlagališče obstaja se izvede izbris iz podatkovne baze in vrne status 200(OK) s sporočilom, da je bilo odlagališče uspešno izbrisano. V nasportnem primeru pa vrne status 404(Not Found).

Metoda 'updateOdlagalisce' pa skrbi za posodobitev podatkov o odlagališčh v podatkovni bazi. Kot prarametre prejme enake stvari kot metoda za dodajane odlagališč z dodanim id-jom(Če odlagališle s tem id-jom ne obstaja se vrne status 404) prek katerega pridobim stare podake odlagališče, ki jih lahko primerjam. V primeru, da so prejete nove slike se shranijo in nato se poti slik združijo z že obstoječimi potmi. Te poti se nato primerjajo med sabo, da se prek poti starih slik izbriše slike, ki niso več potrebne. Na koncu se kliče funkcija za posodobitev odlagališča iz stortivenega razreda, ob uspešni posodobitvi metoda vrne status 200(OK).

Na frontendu aplikacije sem v servisni razred dodal še metode, ki komunicirajo z opisanimi metodami. Prijavljen uporabnik za dodajanje novega odlagališča pritisne gumb dodaj odlagališče in ob pritisku se na enakem mestu kot je prikaz podrobnosti pojavi obrazec za dodajanje novega odlagališča.

**Slika 11: Obrazec za dodajanje novega divjega odlagališča**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, številka

Opis je samodejno ustvarjen

Na zgornji sliki je se vidi začetek in konec obrazca za dodajanje novega odlagališča. Po prikazu obrazca za dodajanje divjega odlagališča lahko uporabnik klikne kjerkoli na mapo in tam se bo prikazal marker, na obrazec pa se bodo dodale koordinate. Premikanje markerja je mogoče, in koordinate se bodo posodabljaje s premikanjem. Edino obvezno besedilno polje za dodajanje novega odlagališča so koordinate, to pomeni, da uporabnik mora klikniti na zemljevid in izbrati lokacijo novega odlagališča. Uporabnik nato izpolni podatke, ki jih ima o odlagališču in prtisne gumb shrani. Ob pritisku na gumb se oblikuje objekt 'Odlagališče', ki se nato skupaj s slikami preko storitvenega razreda pošlje na strežnik.

Prijavljen uporabnik, ki ima status administratorja ima še poleg dodajanja še možnost posodoabljanja in izbrisa divjih odlagališč. Administratorju se pri prikazu podrobnosti pojavita še dva gumba za posodobitev in izbris divjega odlagališča. Pri posodobitvi se odpre skoraj enak obrazec kot pri dodajanju novega odlagališča. Samo da ima še opicjo spremeniti status odlagališča na potrjeno divje odlagališče. Potem se prav tako oblikuje objket, ki se s slikami pošlje kot zahteva na strežnik. Pri pritisku na gumb za izbris odlagališča pa se odpre pojavno okno, ki uporabnika vpraša, če želi izbristai to odlagališče. Tam ima opcijo izbirati med da ali ne. Pojavno okno se pojavi zato, da ne pride do nenamernih izbrisov divjih odlagališč iz baze podatkov.

# 7 TESTIRANJE SPLETNE APLIKACIJE

Testiranje aplikacije sem razdelil na testiranje uporabniškega vmesnika in na testiranje zahtev na strežnik. Strežnik sem testiral s pomočjo testov. Uporabniški vmesnik pa sem testiral tako, da sem manjšemu številu ljudi omogočil uporabo oziroma testiranje spletne aplikacije.

Prvi korak pri testiranju je bila ustrezna priprava okolja za testiranje. To je bilo doseženo z uporabo anotacij '@SpringBootTest', '@AutoConfigureMockMvc' in '@Import(TestDatabaseConfiguration.class)', zadnja anotacija je potrebna, ker uporabljam podatkovne baze. V razred za testiranje sem nato uvozil vse potrebne razrede za izvajanje testov. Pred začetkom testiranja sem še pripravil testno podatkovno bazo. Na spodnji sliki je primer priprave testne baze podatkov za testiranje prijave v aplikacijo.

**Slika 12: Priprava testne baze podatkov**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Anotacija pred začetkom funkcije povzroči, da se baza podatkov vzpostavi pred vsakim testom. Po pripravljeni bazi za primer prijave v aplikacijo pa lahko napišemo teste za prijavo.

**Slika 13: Test prijave uporabnika v sistem**

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Na zgornji sliki je test, ki testira zahtevo poslano strežniku za prijavo v spletno aplikacijo. V testu simuliramo POST zahtevek na poti '/odlagalisca/user/login', zahtevek vsebuje 'requestUser' objekt v katerem sta elektronski naslov in geslo. Test nato preveri pričakovane rezultate, ko strežnik odgovori. V tem primeru preverja statusno kodo in objekt v odgovoru. Za testiranje prijave je bilo potrebno napisati še en test, ki pa testira prijavo z napačnim elektronskim naslovom ali geslom.

Na tak način so bile testirane tudi vse druge metode v krmilniku strežnika. Testi olajšajo testiranje spletne aplikacije po spremembah, saj ni potrebno ročno testirati vsakega zahtevka vsakič znova, ko se naredijo spremembe. Aplikacijo sem testiral tekom razvoja tudi ročno, saj sem teste za aplikacijo napisal na koncu razvoja aplikacije. Če bi bila aplikacija bolj obsežna in bi na razvoju aplikacije delalo več ljudi, bi bilo teste potrebno pisati sproti, saj je s tem testiranje pri spremembah veliko lažje.

Spletna aplikacija večini testnih uporabnikov ni delala težav in nihče ni potreboval pomoči. Z funkcionalnostmi aplikacije so bili večinoma zadovoljni, so pa komentirali, da bi želeli več možnosti za filtriranje divjih odlagališč, na primer po občinah ali regijah. Komentirali so tudi, da bi si želeli, kot uporabnik brez statusa administratorja imeti možnost urejanja divjega odlagališča, ki so ga oni dodali. Saj se lahko velikokrat zgodi, da pri se pri vnašanju informacij lahko zmotimo, ali pa pridobimo več informacij o divjem odlagališču, ki jih potem ne morejo dodati. To bi tudi olajšalo delo administratorjev, saj bi bilo potrebno manj popravkov pred potrditvijo divjega odlagališča. Pri testiranju funkcionalnosti administratorja pa so si uporabniki želeli, da bi obstajal gumb za spremembo statusa odlagališča, kar bi olajšalo delo administratorjev pri potrjevanju divjih odlagališč, saj trenutno morajo odpreti urejevalnik odlagališča, tam najti potrditveno polje, ga spremeniti, nato pa se premakniti do konca obrazca in tam pritisniti shrani. Postopek se jim je zdel nepotrebno dolg, če ne želiš spremeniti podatkov o odlagališču, ampak le status odlagališča.

LITRETURA IN VIRI

Bagon, Š. (2023, 02). *Na inšpektoratu dobijo več prijav, kot jih lahko sproti obdelajo*. https://obalaplus.si/na-inspektoratu-dobijo-vec-prijav-kot-jih-lahko-sproti-obdelajo/

Loznar, A., Bućan, V., Lipuš, I., Kavčič, A., Mohar, G., & Odar, A. (2015). *PRIJAVI DIVJE ODLAGALIŠČE*. http://zagovorniki-okolja.si/wp-content/uploads/2018/11/Prijavi-divje-odlagali%C5%A1%C4%8De-A.-Lozar-V.-Bu%C4%87an-I.-Lipu%C5%A1-A.-Kav%C4%8Di%C4%8D-G.-Mohar-A.-Odar.pdf

Mrak, D. I., & Damjanovič, Z. (2019). *DIVJA ODLAGALIŠČA ODPADKOV V NAŠEM OKOLJU*. https://www.pzs.si/javno/kvgn\_dokumenti/Seminarske%20naloge%20VGN/Damjanovi%C4%8D%20Zdravko\_Divja%20odlagali%C5%A1%C4%8Da%20odpadkov%20v%20na%C5%A1em%20okolju.pdf