Tehničko veleučilište u Zagrebu  
Stručni studij računarstva

Erik Mihaljinec Kotarski

**Razvoj web aplikacije za ATV servis**

Završni rad br. 0246096063-53-2022-1

Zagreb, rujan, 2023.

Tehničko veleučilište u Zagrebu  
Stručni studij računarstva

Erik Mihaljinec Kotarski

JMBAG: 0246096063

**Razvoj web aplikacije za ATV servis**

Završni rad br. 0246096063-53-2022-1

Zagreb, rujan, 2023.

**Predgovor**

Zahvaljujem se svojoj mentorici mag.ing.graph.techn Dariji Ćutić na uloženom trudu, strpljenju, vremenu i pomoći kroz izvršavanje ovog završnog rada. Zahvaljujem se svim nastavnicima i djelatnicima Tehničkog veleučilišta u Zagrebu na stečenom znanju.

**Sažetak**

Teorijski dio rada sadrži ideju aplikacije, arhitekturu te objašnjenje pojedinih alata korištenih za razvoj web aplikacije. Uz pojašnjenje trenutne implementacije, navedena su i daljnja moguća proširenja za poboljšanje krajnjeg proizvoda.

Praktična demonstracija web aplikacije je raspoređena na DigitalOcean poslužitelju za prikaz funkcionalnosti aplikacije. DigitalOcean je povezan sa GitLab repozitorijem sa kojega se direktno primjenjuju promjene.

Ključne riječi: Python, Django, Docker, ATV servis, PostgreSQL, pgAdmin, DigitalOcean, React, Next.js 13, GitLab

**Sadržaj**

[1 UVOD 1](#_Toc144558052)

[2 Ideja aplikacije 2](#_Toc144558053)

[3 Arhitektura aplikacije 3](#_Toc144558054)

[4 Alati za razvoj aplikacije 4](#_Toc144558055)

[4.1 Python 5](#_Toc144558056)

[4.2 Django 5](#_Toc144558057)

[4.2.1 Konfiguracija 6](#_Toc144558058)

[4.2.2 Migracije 10](#_Toc144558059)

[4.2.3 Inicijalni podatci 10](#_Toc144558060)

[Pokretanje 11](#_Toc144558061)

[4.2.4 Izrada super admina 11](#_Toc144558062)

[4.2.5 Autorizacija i autentifikacija 12](#_Toc144558063)

[4.3 Baza podataka 13](#_Toc144558064)

[4.3.1 PostgreSQL 13](#_Toc144558065)

[4.3.2 pgAdmin 14](#_Toc144558066)

[4.4 React 15](#_Toc144558067)

[4.4.1 Next.js 13 15](#_Toc144558068)

[4.4.2 Redux Toolkit 16](#_Toc144558069)

[4.4.3 RTK Query 17](#_Toc144558070)

[4.4.4 Tailwind 18](#_Toc144558071)

[4.5 Docker 19](#_Toc144558072)

[5 GitLab 21](#_Toc144558073)

[5.1 Vrsta grananja aplikacije 21](#_Toc144558074)

[5.2 Repozitoriji aplikacije 23](#_Toc144558075)

[5.3 Verzioniranje 24](#_Toc144558076)

[6 DigitalOcean 25](#_Toc144558077)

[7 Proširenja 25](#_Toc144558078)

[8 Prikaz i opis izrađene aplikacije 26](#_Toc144558079)

[8.1 Zajednički elementi 26](#_Toc144558080)

[8.2 Početna stranica 28](#_Toc144558081)

[8.3 Prijava i odjava 29](#_Toc144558082)

[8.4 Funkcionalnosti 31](#_Toc144558083)

[8.5 Sigurnost 40](#_Toc144558084)

[9 Zaključak 42](#_Toc144558085)

**Popis kratica**

AJAX - asynchronous JavaScript and XML

ATV audio-televizija

RAM – radna memorija

DBMS – database management system

CRUD - create, read, update and delete

REST - representational state transfer

DRY - don't repeat yourself

YAML - Yet Another Markup Language)

JSON - JavaScript Object Notation

HTTP *-* Hypertext Transfer Protocol

HTTPS - Hypertext Transfer Protocol Secure

XSS - cross-site-scripting

HTML - HyperText Markup Language

TBD – Trunk-based

CORS - cross-origin resource sharing

CI/CD - continuous integration continuous delivery

URL - Uniform Resource Locator

OIB - osobni identifikacijski broj

**Popis slika**

[Slika 1 prikazuje konfiguracijsku datoteku paketa potrebnih za backend: izvor: autor 6](#_Toc144663236)

[Slika 2 prikazuje komandu za kreiranje virtualnog okruženja: izvor: autor 6](#_Toc144663237)

[Slika 3 prikazuje komande za pokretanje virtualnog okruženja: izvor: autor 7](#_Toc144663238)

[Slika 4 prikazuje komandu za instalaciju backend paketa: izvor: autor 7](#_Toc144663239)

[Slika 5 prikazuje komandu za prvo pokretanje Django projekta: izvor: autor 7](#_Toc144663240)

[Slika 6 prikazuje pakete i aplikacije za Django projekt: izvor: autor 7](#_Toc144663241)

[Slika 7 prikazuje primjer modela: izvor: autor 8](#_Toc144663242)

[Slika 8 prikazuje primjer klase za serijalizaciju: izvor: autor 8](#_Toc144663243)

[Slika 9 prikazuje primjer klase za kreiranje krajnjih točaka: izvor: autor 9](#_Toc144663244)

[Slika 10 prikazuje primjer datoteke za određivanje prikaza: izvor: autor 9](#_Toc144663245)

[Slika 11 prikazuje fixture datoteku za tipove statusa: izvor: autor 10](#_Toc144663246)

[Slika 12 prikazuje komandu i ispis prilikom pokretanja backend servera: izvor: autor 11](#_Toc144663247)

[Slika 13 prikazuje komandu i podatke potrebne za kreiranje super admina: izvor: autor 11](#_Toc144663248)

[Slika 14 prikazuje djoser postavke i postavke za kolačiće: izvor: autor 12](#_Toc144663249)

[Slika 15 prikazuje pgAdmin sučelje sa inicijalnim podacima za tip statusa: izvor: autor 14](#_Toc144663250)

[Slika 16 prikazuje primjer konfiguracije Next.js automatske rute: izvor: autor 15](#_Toc144663251)

[Slika 17 prikazuje primjer Redux React Hookova: izvor: autor 16](#_Toc144663252)

[Slika 18 prikazuje primjer Redux Storea sa reduktorima: izvor: autor 16](#_Toc144663253)

[Slika 19 prikazuje primjer RTK Querya sa oznakama: izvor: autor 17](#_Toc144663254)

[Slika 20 prikazuje Orders stranicu na velikom ekranu u tamnom stilu: izvor: autor 18](#_Toc144663255)

[Slika 21 prikazuje Orders stranicu na manjem ekranu u svjetlom stilu: izvor: autor 18](#_Toc144663256)

[Slika 22 prikazuje konfiguraciju backend Dockerfile datoteke: izvor: autor 19](#_Toc144663257)

[Slika 23 prikazuje konfiguraciju backend docker-compose datoteke: izvor: autor 20](#_Toc144663258)

[Slika 24 prikazuje komandu za kreiranje i ispis Git grana: izvor: autor 21](#_Toc144663259)

[Slika 25 prikazuje primjer Trunk-based grananja: izvor: [6] 22](#_Toc144663260)

[Slika 26 prikazuje primjer GitFlow grananja: izvor: [7] 22](#_Toc144663261)

[Slika 27 prikazuje GitLab grupu i podgrupu aplikacije: izvor: autor 23](#_Toc144663262)

[Slika 28 prikazuje GitLab Backend REST API repozitorij: izvor: autor 23](#_Toc144663263)

[Slika 29 prikazuje GitLab Frontend Next.js repozitorij: izvor: autor 24](#_Toc144663264)

[Slika 30 prikazuje primjer Git komande za vraćanje na određenu verziju: izvor: autor 24](#_Toc144663265)

[Slika 31 prikazuje podnožje: izvor: autor 26](#_Toc144663266)

[Slika 32 prikazuje navigaciju za neprijavljenog korisnika: izvor: autor 26](#_Toc144663267)

[Slika 33 prikazuje navigaciju za prijavljenog korisnika: izvor: autor 26](#_Toc144663268)

[Slika 34 prikazuje korisničku navigaciju admin korisnika: izvor: autor 27](#_Toc144663269)

[Slika 35 prikazuje korisničku navigaciju običnog korisnika: izvor: autor 27](#_Toc144663270)

[Slika 36 prikazuje navigaciju na manjim ekranima: izvor: autor 27](#_Toc144663271)

[Slika 37 prikaz padajući izbornik za mijenjanje teme: izvor: autor 28](#_Toc144663272)

[Slika 38 prikazuje početnu stranicu web aplikacije: izvor: autor 28](#_Toc144663273)

[Slika 39 prikazuje stranicu za prijavu: izvor: autor 29](#_Toc144663274)

[Slika 40 prikazuje primjer validacije na korisničkom sučelju: izvor: autor 29](#_Toc144663275)

[Slika 41 prikazuje primjer notifikacije u slučaju uspješnog izvršenja akcije: izvor: autor 30](#_Toc144663276)

[Slika 42 prikazuje primjer notifikacije u slučaju ne uspješnog izvršenja akcije: izvor: autor 30](#_Toc144663277)

[Slika 43 prikazuje postavljanje autoriziranog stanja: izvor: autor 30](#_Toc144663278)

[Slika 44 prikazuje Orders stranicu bez kreiranih statusa: izvor: autor 31](#_Toc144663279)

[Slika 45 prikazuje Orders stranicu bez postavljenog početnog statusa za naloge: izvor: autor 31](#_Toc144663280)

[Slika 46 prikazuje Orders stranicu koja je spremna za upotrebnu: izvor: autor 32](#_Toc144663281)

[Slika 47 prikazuje uvjete za korištenje i polja za pretragu nad nalozima: izvor: autor 32](#_Toc144663282)

[Slika 48 prikazuje primjer kreiranog naloga sa ne proširenim retkom: izvor: autor 32](#_Toc144663283)

[Slika 49 prikazuje dodatne akcije za redak naloga: izvor: autor 33](#_Toc144663284)

[Slika 50 prikazuje dijalog za završavanje naloga: izvor: autor 33](#_Toc144663285)

[Slika 51 prikazuje primjer proširivog retka i filtera za naloge: izvor: autor 34](#_Toc144663286)

[Slika 52 prikazuje prvi korak dijaloga za kreiranje naloga sa validacijom: izvor: autor 34](#_Toc144663287)

[Slika 53 prikazuje drugi korak dijaloga za kreiranje naloga sa validacijom: izvor: autor 35](#_Toc144663288)

[Slika 54 prikazuje polje za uređaj kod kreiranja naloga kada nije kreiran ni jedan uređaj: izvor: autor 35](#_Toc144663289)

[Slika 55 prikazuje Devices stranicu sa prikazom Devices taba: izvor: autor 35](#_Toc144663290)

[Slika 56 prikazuje primjer forme za unos uređaja: izvor: autor 36](#_Toc144663291)

[Slika 57 prikazuje Devices stranicu sa prikazom Brand taba: izvor: autor 36](#_Toc144663292)

[Slika 58 prikazuje Materials stranicu sa prikazom Materials taba: izvor: autor 37](#_Toc144663293)

[Slika 59 slika prikazuje primjer obavijesti u slučaju greške kod provjere jedinstvenosti: izvor: autor 38](#_Toc144663294)

[Slika 60 prikazuje Status stranicu sa prikazom Status Type taba: izvor: autor 38](#_Toc144663295)

[Slika 61 prikazuje profilnu stranicu za promjenu korisničkog imena: izvor: autor 38](#_Toc144663296)

[Slika 62 prikazuje profilnu stranicu za promjenu zaporke: izvor: autor 39](#_Toc144663297)

[Slika 63 prikazuje profilnu stranicu za promjenu osobnih podataka: izvor: autor 39](#_Toc144663298)

[Slika 64 prikazuje tablicu korisnika: izvor: autor 40](#_Toc144663299)

[Slika 65 prikazuje formu za kreiranje korisnika: izvor: autor 40](#_Toc144663300)

[Slika 66 prikazuje postavljanje zadanog statusa za naloge: izvor: autor 41](#_Toc144663301)

[Slika 67 prikazuje raspored koji nasljeđuju ugniježđene Dashboard rute: izvor: autor 41](#_Toc144663302)

[Slika 68 prikazuje provjeru za autorizaciju: izvor: autor 41](#_Toc144663303)

[Slika 69 prikazuje stranicu u slučaju pokušaja pristupa neprijavljenog korisnika na neku od Dashboard stranica: izvor: autor 42](#_Toc144663304)

[Slika 70 prikazuje stranicu u slučaju pokušaja pristupa običnog korisnika upravljačkim stranicama: izvor: autor 42](#_Toc144663305)

# UVOD

Web aplikacije su aplikacijski programi koji su spremljeni na lokalnom ili udaljenom serveru te se usluga pruža preko Interneta u web-pregledniku. Prve web aplikacije bile su vrlo jednostavni programi koji su slali upite prema bazi, te dobivale nazad rezultat. Bile su ograničene ponovnim prikazivanjem u potpunosti novih web stranica, što je usporavalo rad. 1995. godine izašao je JavaScript i pružao je mogućnost dinamičkih elemenata na klijentskoj strani. Sa dolaskom AJAX (asynchronous JavaScript and XML) omogućena je asinkrona razmjena između klijenta i servera bez potrebe osvježavanja stranice [1]. Današnje aplikacije nastoje učitati što više stranica i komponenata na serveru, kako bi se smanjila količina JavaScripta koja se šalje klijentu i time unaprjeđuju performanse. Takav pristup je korišten i prilikom razvoja ove web aplikacije.

Zadatak je izraditi web aplikaciju koja sadrži glavne značajke za poslovanje servisa poput praćenje naloga i stanje materijala. U svrhu pojednostavljenja svakodnevnih radnji, te smanjiti mogućnost ljudskih grešaka. Karakteristike koje će se implementirati su: podjela aplikacije na korisničko sučelje i *backend* za bolju modularnost. Korisničko sučelje biti će responzivno i nuditi mogućnost korištenja na većim i manjim ekranima. Praćenje stanja aplikacije poput prijave korisnika. Implementirati sigurnost i više razina sigurnosti za različite korisnike. U ovu svrhu potrebno je implementirati autorizaciju i autentifikaciju na *backendu,* te razine pristupa na krajnjim točkama. Implementirati bazu podataka za pohranu i manipulaciju podataka. Implementirati mogućnost raspoređivanja preko trećeg poslužitelja ili kontejnerizaciju.

# Ideja aplikacije

Primarni cilj aplikacije je pojednostavljenje i automatizacija svakodnevnih procesa. Neki od tih svakodnevnih procesa su: unos naloga, promjena naloga, praćenje materijala i sl.

Bitno je zadržati učestalost aplikacije. Ukoliko se negdje napravi promjena, ta promjena mora biti provedena na svim povezanim mjestima, na primjer: ukoliko se promijeni naziv uređaja na sučelju za kontrolu uređaja, promjena tog naziva mora biti provedena na svim nalozima koji sadrže taj uređaj.

Osim raznih aktivnosti sa nalozima, implementacija poluautomatskog lagera je od velike važnosti. U ATV servisu svakodnevno se koriste razni materijali i praćenje zaliha vrlo brzo postane složen proces. Jedna od najčešćih pogrešaka je zaboraviti smanjiti količinu materijala na lageru u Excel tablici ili sličnim alatima koji zahtijevaju ručan unos. Stoga se materijal unosi prilikom dolaska, te prilikom završetka naloga nakon što se odaberu korišteni materijali, radi se provjera i automatski oduzima odabrana količina. U slučaju da na stanju nema odabrana količina, sustav će korisniku izbaciti grešku.

Također, aplikacija je razvijena imajući nadogradnju na umu. Za olakšanje daljnjeg razvoja, komponente su modularne i izmjenjive. Korisničko sučelje ili *backend* može se kasnije zamijeniti drugom implementacijom.

# Arhitektura aplikacije

Arhitektura aplikacije općenito se dijeli na korisničko sučelje i *backend*. Korisničko sučelje i backend su modularni, te se mogu izmijeniti sa nekim drugim rješenjima od trenutno implementiranih.

Aplikacija je raspoređena na DigitalOcean poslužitelju unutar kontejnera. Lako je skalabilna, te ima mogućnost promjene konfiguracije, npr. količina RAM-a (radne memorije), vrsta i brzina procesora itd. Ukoliko se to želi, moguće je promijeniti poslužitelja. Važno je napomenuti da aplikacija sadrži Docker datoteke, koje omogućavaju lako raspoređivanje aplikacije i bez stranog poslužitelja.

Postoji nekoliko razina pristupa. Te razine su podijeljene na korisnika, admina i super admina. Korisnik ima pristup svim funkcijama za normalan rad poput kreiranja naloga, uređaja i sl. Uz razinu pristupa korisnika, admin ima dodatan pristup za kreiranje korisnika, brisanje korisnika, te promjenu pojedinih korisničkih podataka. No, nema pristup promjene podataka drugim adminima ili brisanje drugih admina. Super admin ima sve ovlasti, te može dodatno brisati ili mijenjati podatke drugih admina.

# Alati za razvoj aplikacije

U ovoj sekciji su ukratko navedeni alati koji se koriste za razvoj aplikacije. Svaki od alata dalje je detaljnije objašnjen u svojoj sekciji.

Programski jezik koji se koristi za razvoj *backenda* je Python. Python dolazi sa mnoštvom standardnih paketa. Svrha paketa je raznolika, od rada sa datotekama, do web servera koji je spreman za upotrebu [2]. Uz Python, koristi se okvir Django. Django omogućuje laku skalabilnost aplikacije, te je pregledniji kada se radi o većim projektima zbog poticanja kodnih standarda.

Za korisničko sučelje koristi se React. To je JavaScript biblioteka koja nudi dodatne mogućnosti. Koristi se u kombinaciji sa Next.js 13 koji dalje ubrzava razvoj. Redux Toolkit i RTK Query koriste se za praćenje stanja aplikacije, dohvat podataka i mutacije. Za osiguranje responzivnosti i stiliranje komponenata koristi se Tailwind.

Gotovo svaka aplikacija ima potrebe za spremanje nekih podataka, te je potrebna baza podataka. Za DBMS (Database Management System) koristi se PostgreSQL. PostgreSQL omogućava spremanje velikih količina podataka na siguran način. U slučaju drugačijeg načina raspoređivanja aplikacije od DigitalOceana, za pregled i upravljanje tih podataka koristi se pgAdmin.

Kontejnerizacija uz pomoć Dockera koristi se ukoliko raspoređivanje nije preko DigitalOcean poslužitelja. Docker služi kao dodatno osiguranje za stabilan rad aplikacije u raznim okolinama, te jednostavno raspoređivanje, bilo to na Windows, Linux ili nekom drugom okruženju.

Aplikacija prolazi kroz više verzija. Potreban je jednostavan način za verzioniranje, nadogradnju i popravljanje aplikacije. Za to se koristiti GitLab. GitLab omogućuje laku nadogradnju aplikacije uz mnoštvo analitičkih i integracijskih alata.

## Python

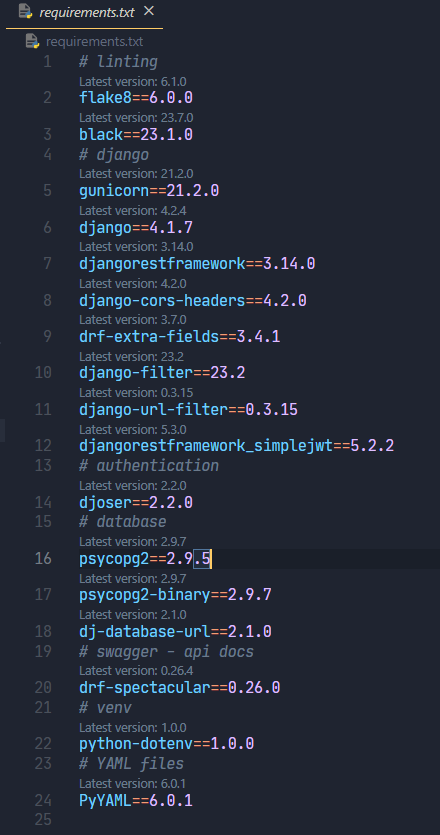
Python je interpretirajući programski jezik, koji podržava različite paradigme programiranja. Neke od njih su: objektno-orijentirano, proceduralno, funkcionalno itd. To je jezik sa dinamičkom sintaksom, što znači da se tip podataka ne treba navoditi, već se prilikom pokretanja određuje tip podataka. Koristi se u raznim područjima i ne samo za razvoj web aplikacija. Neka od njih su: analiza i manipulacija podataka u područjima znanosti i financija, strojno učenje, te primjena različitih algoritama.

## Django

Kao okvir za razvoj web aplikacije korišten je Django. Pisan je u Python programskom jeziku. Pruža razne pripremljene alate koji znatno ubrzavaju razvoj krajnjih točaka za CRUD (create, read, update and delete) operacije. Django REST (representational state transfer) Framework je alat korišten za razvoj krajnjih točaka. Zajedno čine jedan od najpopularnijih načina izrade prilagodljivog aplikacijskog programskog sučelja. Django je skalabilan i jako dobro se nosi sa visokim opterećenjima. [3]. Bitno je napomenuti da se Django konstantno unapređuje. Time je osigurano da će podržavati najnovije standarde, što čini aplikaciju boljim krajnjim proizvodom. Ima podršku za razne baze podataka. Potiče praćenje dobrih programskih standarda i čitljivost. Neki od njih su „*KISS'* (keep it short and simple) i *„DRY“* (don't repeat yourself). „*KISS*“ predstavlja kod koji je kratak i lako razumljiv. Prema „*DRY*“ principu rada, programski uzorci koji se ponavljaju, zamjenjuju se apstrakcijama. To omogućuje korištenjem sličnog koda na različitim mjestima, čime se potiče ponovno korištenje koda.

### Konfiguracija

Unutar početnog direktorija gdje se nalazi *backend*, kreirana je datoteka „requirements.txt“ u kojoj su navedeni paketi potrebni prilikom izvođenja. Radi stabilnosti, paketi su navedeni sa eksplicitnim verzijama.



Slika prikazuje konfiguracijsku datoteku paketa potrebnih za backend: izvor: autor

Prije instalacije, kreirano je virtualno okruženje. Virtualno okruženje kreira se radi odvajanja paketa za različite projekte (umjesto da su instalirani globalno i koriste se nepotrebni paketi).



Slika prikazuje komandu za kreiranje virtualnog okruženja: izvor: autor

Nakon kreiranja, potrebno je aktivirati virtualno okruženje.



Slika prikazuje komande za pokretanje virtualnog okruženja: izvor: autor



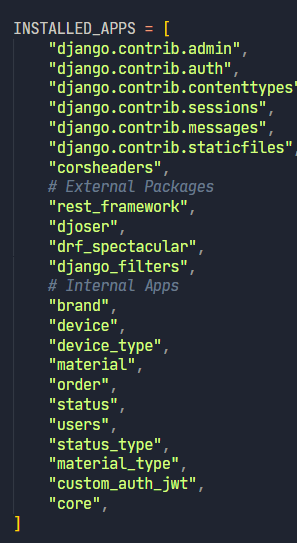
Slika prikazuje komandu za instalaciju backend paketa: izvor: autor

Nakon instalacije paketa, pokretanjem komande za izrade Django projekta, generirana je inicijalna konfiguracija.



Slika prikazuje komandu za prvo pokretanje Django projekta: izvor: autor

Unutar „settings.py“ datoteke postavljena je sva konfiguracija za Django projekt. U njoj su navedene aplikacije koje koristi projekt, podatci za spajanje na bazu, dozvole, postavke za kolačiće itd. Osim internih aplikacija koje su samostalno kreirane, navedene su i vanjske aplikacije (aplikacije instalirane kao paket) poput „djoser“ paketa za konfiguriranje dozvola.



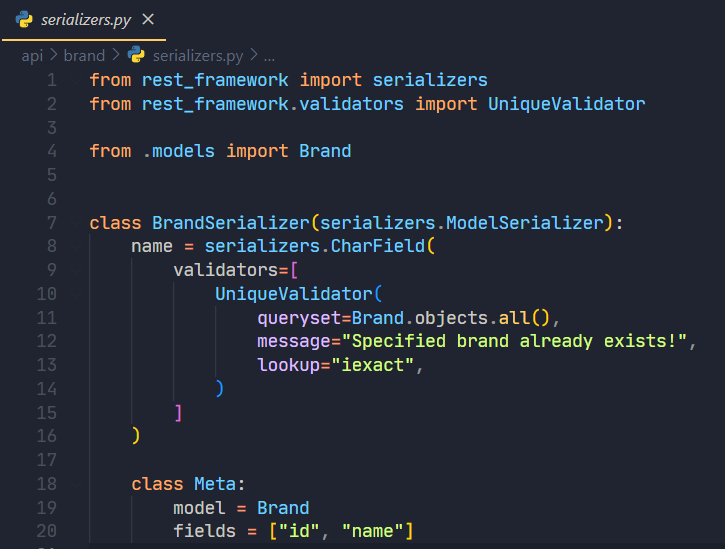
Slika prikazuje pakete i aplikacije za Django projekt: izvor: autor

Nakon pokretanja komande za kreiranje aplikacije unutar Django projekta, kreiran je direktorij sa nazivom aplikacije i potrebnim datotekama za postavljanje modela, krajnjih točaka, serijalizaciju i sl. Modeli opisuju tablicu u bazi podataka. Definirana su imena polja, tipovi pojedinih polja, duljina i sl.



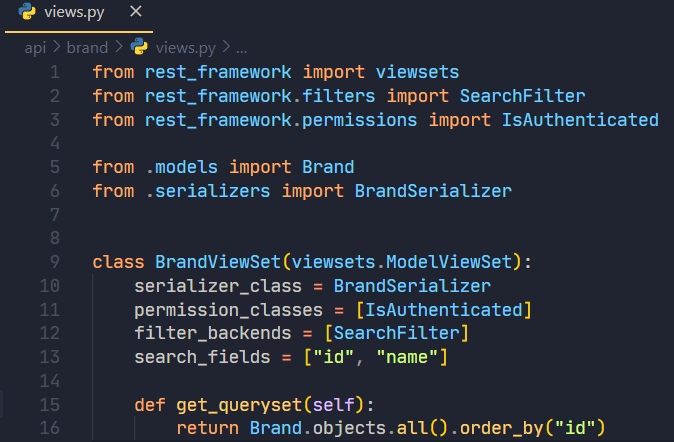
Slika prikazuje primjer modela: izvor: autor

U datoteci za serijalizaciju opisan je prikaz modela prema korisničkom sučelju. Prema zadanim postavkama, postavljanjem „unique“ postavke na „True“, provjera jedinstvenosti je osjetljiva na mala i velika slova. U serijalizacijskoj datoteci, na modelima kojima je potrebno, postavljena je provjera koju pruža „rest\_framework“ paket te je neosjetljiva na mala i velika slova.



Slika prikazuje primjer klase za serijalizaciju: izvor: autor

Za kreiranje krajnjih točaka, koristi se „ModelViewSet“ koji pruža „rest\_framework“ paket. Ovime se automatski generiraju sve CRUD metode. Unutar ove datoteke također su postavljena polja za pretragu i korišteni filteri. Funkcijom „get\_queryset“ naveden je model koji se dohvaća i način poretka.



Slika prikazuje primjer klase za kreiranje krajnjih točaka: izvor: autor

Unutar „urls.py“ datoteke definiran je prefiks i naveden je prikaz koji se koristi.

A screen shot of a computer program

Description automatically generated

Slika prikazuje primjer datoteke za određivanje prikaza: izvor: autor

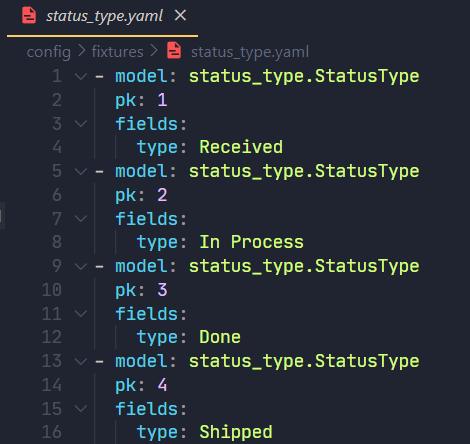
Na ovaj način definirani su i ostali modeli tablica, sa svojim validatorima i prikazima.

### Migracije

Django dolazi sa sustavom za izradu i provođenje migracija. Nakon kreiranja modela, pozivanjem komande za kreiranje migracije i komande za propagiranja migracije, u bazi se kreiraju tablice. Prilikom migriranja, kreiraju se revizije, tj. verzije. Time je omogućen odabir drugačije revizije (naprijed ili nazad) u bilo kojem trenutku.

### Inicijalni podatci

Django omogućava kreiranje kolekciju zapisa u bazi tzv. *fixtures*. Podržani su različiti tipovi datoteka za definiranje *fixturesa* poput YAML (Yet Another Markup Language), JSON (JavaScript Object Notation). Pred konfigurirana je kolekcija zapisa za tipove statusa. Time, korisnik odabire tip statusa i željeno ime za vlastiti status. Može imati više statusa istog tipa, no različitog imena za različite scenarije.



Slika prikazuje fixture datoteku za tipove statusa: izvor: autor

### Pokretanje

Server se pokreće jednom komandom dodavanjem 0.0.0.0 adrese i proizvoljnog broja porta. Ukoliko se ne navede 0.0.0.0 adresa, server će biti dostupan samo na lokalnom poslužitelju.

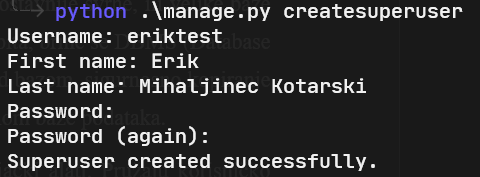
A screen shot of a computer error

Description automatically generated

Slika prikazuje komandu i ispis prilikom pokretanja backend servera: izvor: autor

### Izrada super admina

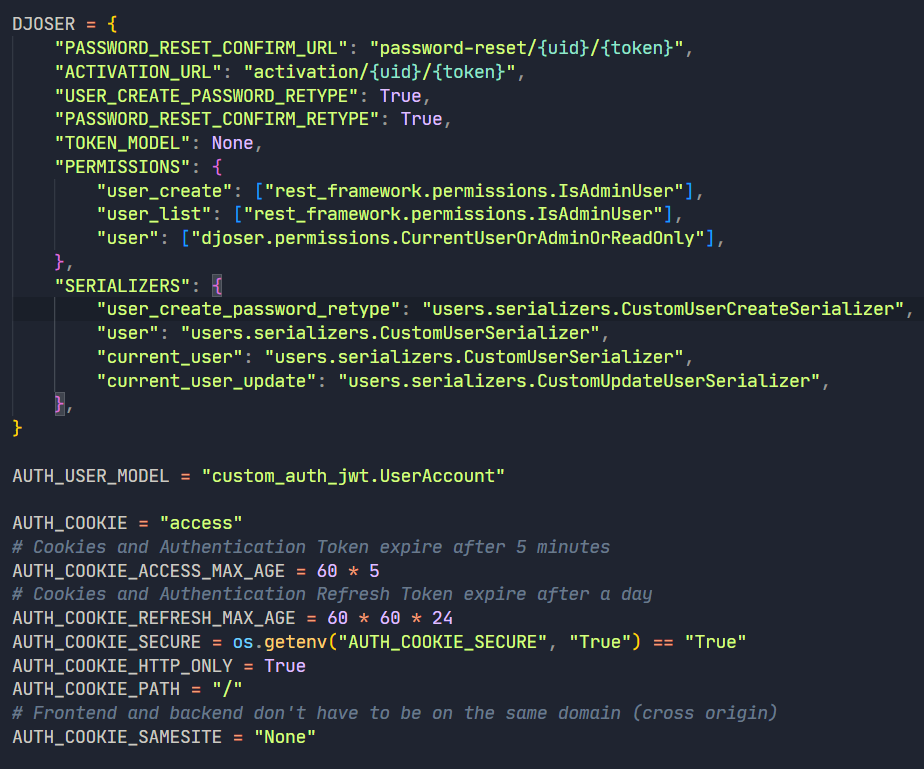
Spomenuto je da se aplikacija sastoji od više razina pristupa. Jedna od njih je super admin. Kreiranje super admina izvršava se preko ljuske. Nakon pokretanja komande za kreiranje, ispune se korisnički podatci i vjerodajnice za pristup. Super admin dalje kreira admine i korisnike.



Slika prikazuje komandu i podatke potrebne za kreiranje super admina: izvor: autor

### Autorizacija i autentifikacija

U svakoj aplikaciji sigurnost je jedan od bitnijih faktora. Django već dolazi sa zadanim postavkama, no nadjačane su vlastitom implementacijom sa *HttpOnly (*Hypertext Transfer Protocol Only*)* kolačićima za bolju sigurnost. Mogu se koristiti samo preko HTTPS (Hypertext Transfer Protocol Secure) zahtjeva. Korištenje takve vrste kolačića smanjuje rizik od napada preko XSS (cross-site-scripting). U datoteci za konfiguraciju, podešeno je trajanje pristupnog tokena, te trajanje tokena za osvježavanje. Implementirane su krajnje točke za kreiranje, validaciju i osvježavanje tokena te je prilagođen bazni korisnički model. Na način na koji je kreiran model iz prijašnjeg primjera, kreiran je prilagođen korisnički model i krajnje točke. Cjelokupan proces pojednostavljen je „djoser“ paketom. U datoteci za konfiguraciju, postavljene su dodatne postavke za dozvole, serijalizaciju podataka na različitim krajnjim točkama, te postavke za kolačiće.



Slika prikazuje djoser postavke i postavke za kolačiće: izvor: autor

## Baza podataka

Kao što je spomenuto u uvodnom dijelu, velikom broju aplikacija današnjice potreban je način pohrane podataka. To može biti tekstualna datoteka za najjednostavnije svrhe ili velike baze podataka sa više servera. Za spremanje, dohvat ili pokretanje upita, brine se DBMS. Također, pruža autorizacijske ovlasti nad bazom, sigurnosno kopiranje podataka, povrat podataka i sl. To čini DMBS-om glavnim dijelom baze podataka.

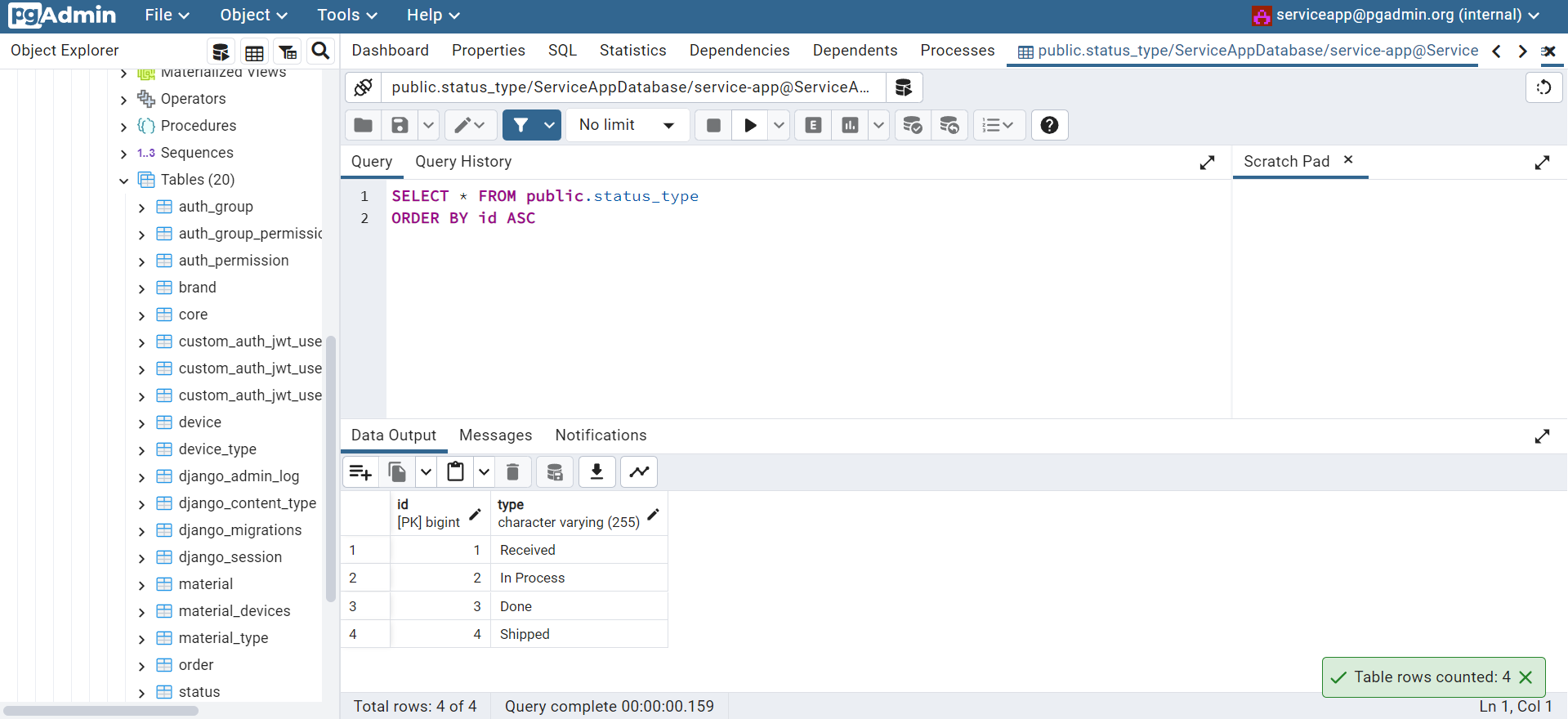
Za dodatno pojednostavljenje koriste se razni dodatni upravljački alati. Pružaju korisničko sučelje sa kojim se izrađuju upiti i omogućavaju jednostavan pregled tablica, korisnika i svih ostalih postavki vezanih za bazu.

### PostgreSQL

Korišten DBMS za ovu aplikaciju je PostgreSQL. Django podržava više baza podataka, no preporučena je PostgreSQL. Glavni razlozi su više pruženih funkcionalnosti, te jednostavnije rješavanje problema u slučaju kompleksnijih baza podataka. Neke stvari za koje se koristi baza su: spremanje, pretraga tj. filtriranje, sortiranje, brisanje podataka itd. Omogućava brz dohvat podataka i dovoljno brzu pretragu u većini slučajeva. Za dodatnu optimizaciju brzine koriste se indeksi. Indeksi su postavljeni na stupcima u tablicama unutar baze. Postavljaju se nad stupcima nad kojima se najčešće radi pretraga. Nedostatak indeksiranja je u tome što zauzima dodatni prostor u memoriji. No, s obzirom na prednosti taj nedostatak je zanemariv.

### pgAdmin

PgAdmin je korišteni alat za upravljanje PostgreSQLom. U potpunosti je besplatan i otvorenog koda. S obzirom na to da je otvorenog koda, razni korisnici mogu pridonijeti razvoju aplikacije, čime se ubrzava dodavanje novih mogućnosti i rješavanje problema. Može se pokretati kao web ili desktop aplikacija. Najpoznatiji je i najkorišteniji alat za upravljanje PostgreSQLa. Ima jednostavno grafičko sučelje sa mnogim mogućnostima. Neke od njih su kreiranje novih tablica, pregled tablica, postavljanje korisničkih ovlasti itd. Kod posluživanja aplikacije na DigitalOcean pgAdmin nije implementiran. DigitalOcean nudi mogućnost vođenja administrativnih zadataka poput izrade sigurnosnih kopija, nadograđivanje i slično, te je u tom slučaju pgAdmin dodatan resurs. PgAdmin korišten je u sklopu Docker datoteke kod lokalnog raspoređivanja ili sličnih scenarija gdje je potrebno samostalno rukovođenje baze podataka.



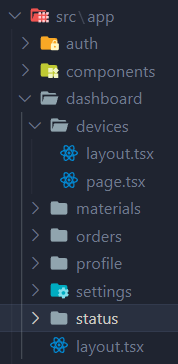
Slika prikazuje pgAdmin sučelje sa inicijalnim podacima za tip statusa: izvor: autor

## React

React je JavaScript biblioteka za izgradnju korisničkih sučelja na efektivan način. Kreiraju se komponente koje se mogu koristiti na više mjesta sa različitim podacima [4]. Konvencija je da se komponente kreiraju unutar „components“ direktorija. S obzirom da se radi o kompleksnijoj aplikaciji, sa praćenjem korisničkog stanja, propagiranjem promjena na klijentu, puno bolja je alternativa od samog JavaScripta ili HTMLa (HyperText Markup Language).

### Next.js 13

Next.js je dodatan *framework* za React, te dolazi sa mnoštvom korisnih mogućnosti. Neke od najznačajnih su automatsko kreiranje ruta, tj. usmjeravanje i server komponenti. Rute se kreiraju dodavanjem direktorija sa imenom željene rute.



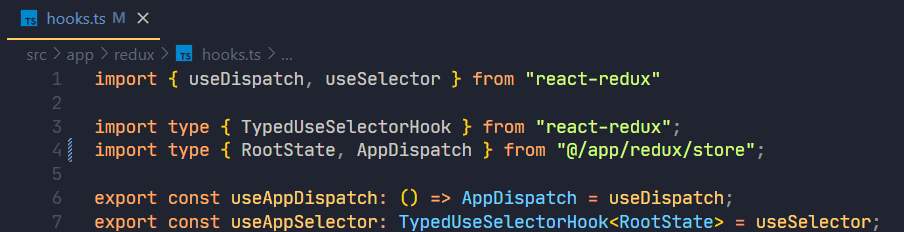
Slika prikazuje primjer konfiguracije Next.js automatske rute: izvor: autor

Direktorij „components“ je direktorij gdje se nalaze komponente korištene kroz aplikaciju, npr. navigacija, tablice za prikaz, gumb i sl. Kreirani direktoriji „auth“ i „dashboard“, te daljnji ugniježdeni direktoriji predstavljaju rute npr. „/dashboard/devices“. Datoteke sa nazivom „page“ predstavljaju sadržaj same stranice na toj ruti, dok datoteke „layout“ predstavljaju općenit raspored. Raspored se također nasljeđuje od roditeljskog direktorija („devices“ direktorij nasljeđuje od „dashboard“, a on nasljeđuje od početnog direktorija“).

Druga značajnija mogućnost su server komponente. Po zadanim postavkama, u Next.js 13, sve komponente i stranice su server komponente. To znači da se te komponente i stranice ne šalju klijentu. Time se znatno dobiva na brzini jer se manje podataka šalje, naročito JavaScripta.

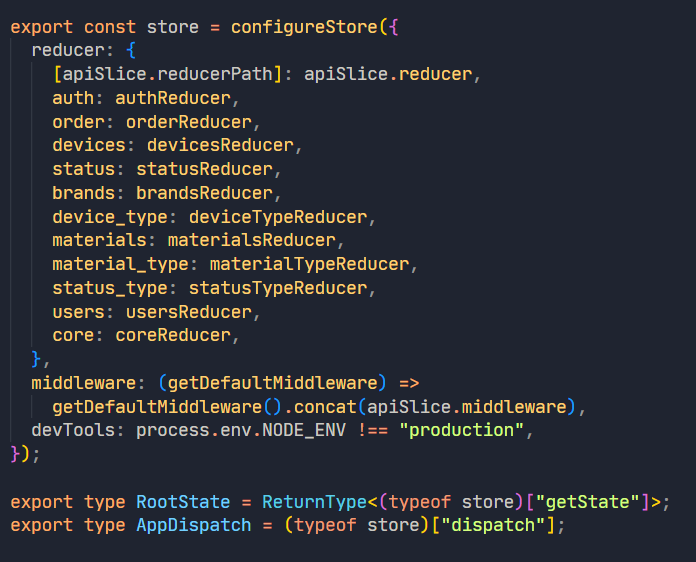
### Redux Toolkit

Redux Toolkit pojednostavljuje i pruža standardizirani način za razvoj Reduxa. Redux je alat za praćenje i postavljanje stanja aplikacije, npr. da je korisnik prijavljen. Koristi akcije i reduktore. Reduktori primaju stanje aplikacije i akciju, te vraćaju novo stanje s obzirom na akciju. Za dohvat stanja koriste se React Hooks koji se automatski generiraju nakon postavljanja Redux Storea.



Slika prikazuje primjer Redux React Hookova: izvor: autor

Redux Store služi za praćenje globalnog stanja aplikacije, te poboljšava brzinu izvođenja na način da resetira prikaz komponenata samo kad je potrebno. Na Redux Store postavljaju se željeni reduktori.



Slika prikazuje primjer Redux Storea sa reduktorima: izvor: autor

### RTK Query

RTK Query je alat za dohvat podataka i slanja mutacija. Koristi se u kombinaciji sa Redux Toolkitom. Pojednostavljuje kreiranje logike za dohvat podataka i za predmemoriranje. Predmemoriranje je konfigurirano predavanjem oznaka i poništavanjem oznaka. Oznake se predaju prilikom dohvata (GET), te se poništavaju prilikom mutacija (POST, PUT, itd.). Dodatno su konfigurirane za pružanje i poništavanje oznaka po logici djelomičnih lista i identifikacijskih oznaka. Prilikom dohvata podataka koristi se paginacija, te se podatci dobivaju kao djelomične liste. RTK Query zna da mora poništiti oznake na samo određenoj djelomičnoj listi, umjesto na svim djelomičnim listama koje se nalaze u pred memoriji. Za građenje upita, predaju se parametri sa definiranim nazivom koji imaju validaciju tipa.



Slika prikazuje primjer RTK Querya sa oznakama: izvor: autor

### Tailwind

Tailwind je framework sa naglaskom na korisnost-prvo. Omogućuje brzo oblikovanje elemenata i komponenata bez dodatnih CSS (Cascading Style Sheets) datoteka. Uz stiliranje nudi mogućnost postavljanja prijelomnih točaka za responzivnost. Uz Tailwind koriste se shadcn komponente koje su unaprijed stilirane i spremne za korištenje. Aplikacija također ima mogućnost za tamni i svijetli stil.

A screenshot of a computer

Description automatically generated

Slika prikazuje Orders stranicu na velikom ekranu u tamnom stilu: izvor: autor

A screenshot of a computer

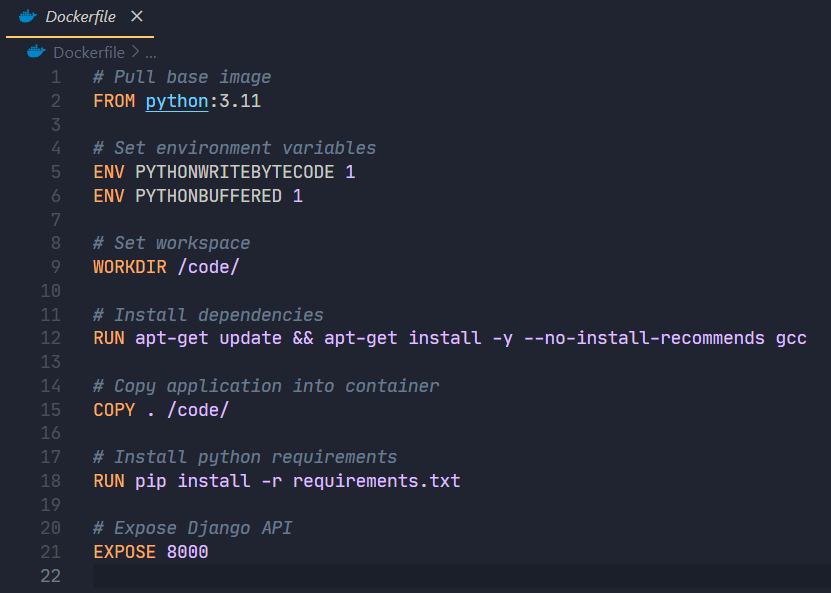
Description automatically generated

Slika prikazuje Orders stranicu na manjem ekranu u svjetlom stilu: izvor: autor

## Docker

Docker se na tržištu pojavio 2013. godine. Odmah nakon njegove pojave postao je vrlo popularan alat. Služi za automatizaciju raspoređivanja aplikacije u kontejnerima. Dodaje mehanizam za pokretanje aplikacija u virtualnom okruženju [5]. Docker virtualno okruženje osigurava stabilnost rada aplikacije na različitim platformama.

Za korištenje potrebne su dvije datoteke: Dockerfile i docker-compose. Dockerfile određuje što će se pokretati i baznu sliku, a docker-compose kako.



Slika prikazuje konfiguraciju backend Dockerfile datoteke: izvor: autor

Svaki segment (Django, PostgreSQL, pgAdmin) komponente je posebno definiran u docker-compose datoteci. Za svaki segment je definirana posebna slika koja se koristi unutar Docker kontejnera. Na primjer, Django segment koristi baznu Python sliku (jer nije definiran dodatan parametar „*image*“), dok DBMS koristi sliku za PostgreSQL.



Slika prikazuje konfiguraciju backend docker-compose datoteke: izvor: autor

Backend ima 3 glavna segmenta: DBMS (PostgreSQL), alat za pregled baze (pgAdmin) i Django server. Svaki segment ima svoj kontejner. Postoje i različite vrste kontejnera. Najčešći, i onaj koji je korišten baziran na Linux-u.

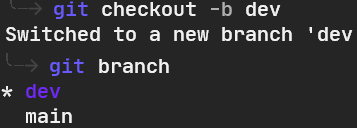
Docker kontejneri su odvojeni jedan od drugog. No, dijele istu privatnu mrežu. S obzirom da se kreira privatna mreža, za pristup alatima, izloženi su samo nužni portovi. Docker rješenje koristi se ukoliko nije željeno posluživanje preko DigitalOceana.

# GitLab

GitLab je kolaboracijska platforma za razvoj programskih aplikacija. Temelji se na Gitu i potiče agilan način rada za brži razvoj aplikacija. Dolazi sa mnogim sigurnosnim i analitičkim mogućnostima za bolji uvid u svim fazama razvoja aplikacija. Neke od njih su skeniranje ovisnih paketa aplikacije za provjeru sigurnosti i praćenje statistika repozitorija (pokrivenost testova, statistika cjevovoda itd.). Pruža jednostavnu konfiguraciju cjevovoda za razne faze provjere koda prije dodavanja na repozitorij. Naročito je ubrzalo razvoj mogućnost i jednostavnost postavljanja projekta na drugom uređaju. Nakon postavljanja osnovnih konfiguracijskih postavka za pristup repozitoriju, korištenjem komande „git pull“, sve datoteke u repozitoriju se skidaju na uređaj.

## Vrsta grananja aplikacije

Prije pojašnjenja vrste grananja korištene prilikom razvoja, potrebno je razumjeti što je koncept grananja. Git podržava tzv. 'grane'. Kreiranjem nove grane 'granamo' aplikaciju i omogućuje nam rad na implementaciji nove značajke, rješavanje nekog problema i sl. bez da utječemo na glavnu, tj. „master“ ili „main“ granu. Naročito je korisno za testiranje nove značajke prije puštanja u produkciju prema krajnjem korisniku.



Slika prikazuje komandu za kreiranje i ispis Git grana: izvor: autor

Postoji više vrsta razvoja, neke od njih su: „Trunk-based“ (TBD), „GitHub flow“, „GitFlow“ itd. Prilikom razvoja koristila se *Trunk-based*  vrsta grananja gdje se potiče dodavanje što manjih promjena na master granu. Ovime je razvoj i uočavanje grešaka bio znatno brži naspram *GitFlow* vrsti grananja.

A diagram of a network

Description automatically generated

Slika prikazuje primjer Trunk-based grananja: izvor: [6]

Kod *GitFlow* vrste grananja greške se kasnije uočavaju. Svaka značajka, kao i kod dobiva svoju granu. Razlika je u vijeku trajanja grana. Vijek trajanja grana može biti znatno veći, sa više promjena. Ukoliko se radi o više većih značajki, time se povećavaju šanse za konflikte između značajki.

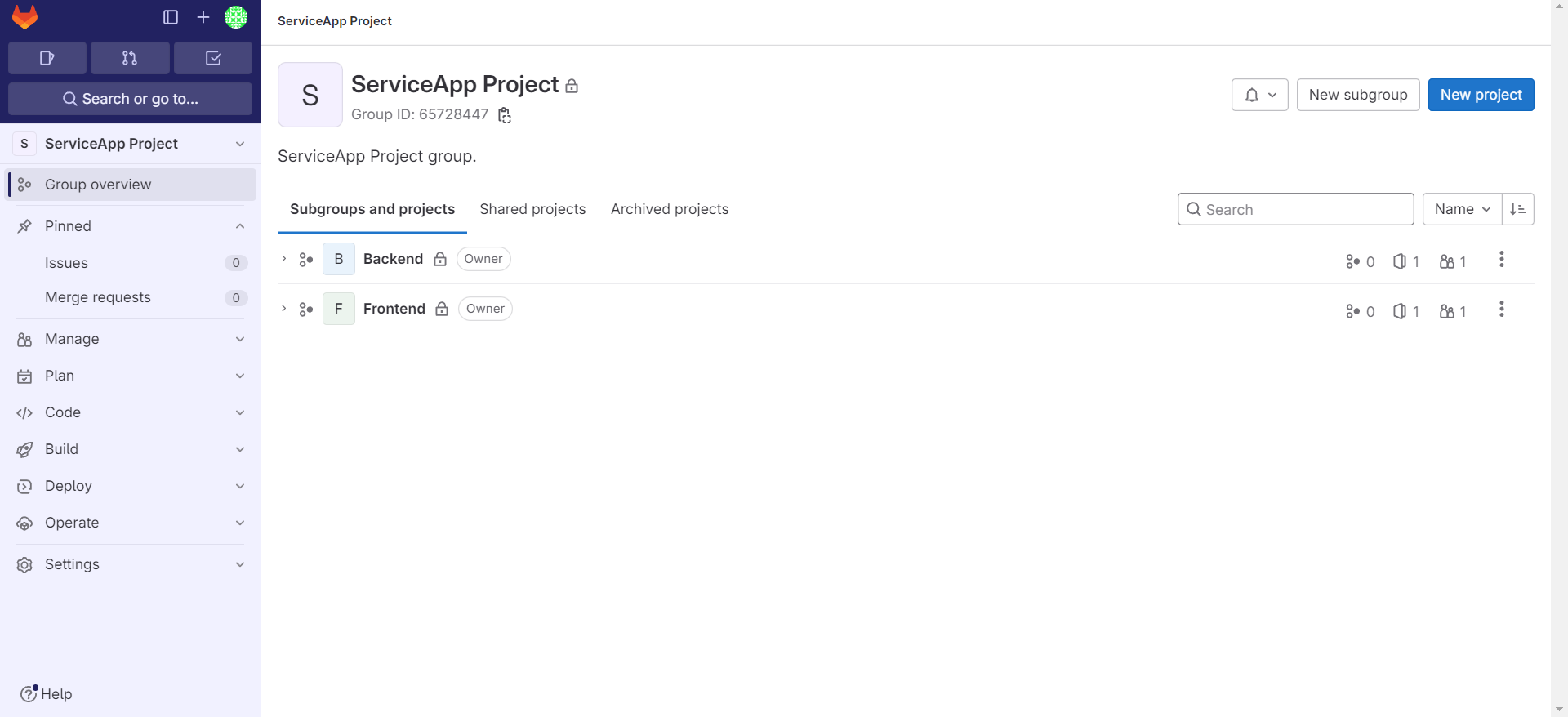
Icon

Description automatically generated

Slika prikazuje primjer GitFlow grananja: izvor: [7]

## Repozitoriji aplikacije

Kreirana je grupa pod nazivom „ServiceApp Project“ . Unutar grupe definirane su podgrupe, tj. glavne komponente aplikacije: „Backend“ i „Frontend“.



Slika prikazuje GitLab grupu i podgrupu aplikacije: izvor: autor

Unutar svake podgrupe nalaze se repozitoriji sa trenutnim implementacijama komponenata kojoj pripadaju. Uz svaki direktorij i datoteku, vidi se prije koliko je napravljena zadnja promjena vezana uz direktorij i/ili datoteku, te poruka prilikom dodavanja promjena.

A screenshot of a computer

Description automatically generated

Slika prikazuje GitLab Backend REST API repozitorij: izvor: autor

A screenshot of a computer

Description automatically generated

Slika prikazuje GitLab Frontend Next.js repozitorij: izvor: autor

## Verzioniranje

Upravljanje verzijama, ili verzioniranje je raspodjela i praćenje softvera kroz više njegovih inačica, tj. iteracija. Uz komande „git add“ koja dodaje sve datoteke na kojima je napravljena promjena, i „git commit“ koja spremi te datoteke, napravi se nova verzija. Svaka verzija dobiva jedinstvenu oznaku. Pomoću tih oznaka, moguće je prebacivanje na različite verzije. Pokretanjem komande „git checkout <commit-id>“ prebacuje se na verziju sa specificiranom oznakom. Prilikom razvoja, u više situacija, bilo je potrebno vratiti se na određenu verziju. Najčešće kako bi se testiralo alternativno rješenje.



Slika prikazuje primjer Git komande za vraćanje na određenu verziju: izvor: autor

# DigitalOcean

Primaran način posluživanja aplikacije je preko usluga DigitalOceana. Nude rukovođenje i skaliranje baze, aplikacije, te povezivanje sa poslužiteljima za serviranje statičkih datoteka. DigitalOcean također koristi kontejnerizaciju aplikacija prilikom raspoređivanja. Osim za raspoređivanje korisničkog sučelja i *backenda*, DigitalOcean korišten je kao poslužitelj imena za domenu. Direktno je povezan sa GitLab repozitorijima, te promijene na „main“ granama uzrokuju novu izgradnju i raspoređivanje. Integriran je uvod u analitiku za praćenje statistike aplikacija. Na svakoj aplikaciji podešene su potrebne varijable okoline, poput konekcije za bazu, dozvoljeni domaćin za *backend*, CORS (Cross-origin resource sharing) dozvoljeno podrijetlo i sl. Osim varijabli okoline, podešene su potrebne domene i pod domene za korisničko sučelje i *backend*. Na *backend* komponenti, nakon uspješnog raspoređivanja, pokrenute su komande za kreiranje super admin i učitavanje inicijalnih podataka u bazu.

# Proširenja

U aplikaciju su integrirane osnovne funkcionalnosti za korištenje i raspoređivanje. Daljnji koraci su integracija GitLab CI/CD (continuous integration continuous delivery) i pisanje testova. Moguća je integracija Elasticsearch alata na *backendu* za optimiziranije indeksiranje podataka za pretragu. Za bolji uvod u poslovanje moguća je integracija analize i statistike podataka. Integracija sustava za plaćanje unutar aplikacije te povezati sustav za plaćanje sa analitičkim sustavom.

# Prikaz i opis izrađene aplikacije

Za izradu korisničkog sučelja, Next.js 13 omogućuje nasljeđivanje elemenata od roditeljske rute, tj. stranice. S obzirom da aplikacija ima više razina pristupa, administrativni korisnik može pristupiti dodatnoj stranici za postavke. Pokušaj pristupa običnog korisnika je onemogućen provjerom administrativnih dozvola. Također, svaki pokušaj pristupa bilo kojoj stranici osim početne je onemogućen za neprijavljenog korisnika provjerom stanja prijave.

Na *backendu* sve krajnje točke su zaštićene. Postavljena je minimalna razina potrebnih dozvola. Za optimiziranu pretragu, umjesto vraćanja svih podataka odjednom, svaka krajnja točka vraća podatke u paginiranom prikazu uz mogućnost promjene količine podataka (10-50). Svaka krajnja točka ima svoja polja za pretragu i filtriranje. Prilikom pozivanja krajnje točke, za kontrolu paginacije, vrijednosti filtera i pretragu, dodani su parametri na bazni URL (Uniform Resource Locator).

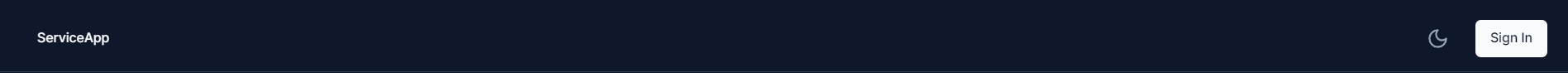
## Zajednički elementi

Sve stranice dijele navigacijsku komponentu i podnožje.



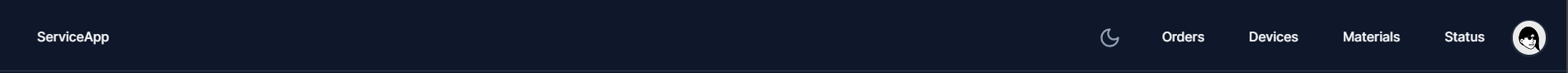
Slika prikazuje podnožje: izvor: autor

Navigacijska komponenta je drugačija za prijavljenog i neprijavljenog korisnika. Neprijavljeni korisnik ima samo mogućnost prijave i promjene teme.



Slika prikazuje navigaciju za neprijavljenog korisnika: izvor: autor

Prijavljeni korisnik uz mogućnost prijave teme, može pristupiti svim stranicama namijenjenim za svakodnevni rad i stranicama za korisnički profil.



Slika prikazuje navigaciju za prijavljenog korisnika: izvor: autor

Klikom na korisničku ikonu, otvara se dodatna navigacija za profil. Kad se radi o administrativnom korisniku, u padajućem izborniku dodatno je prikazana mogućnost navigacije na stranicu za postavke.

A screenshot of a phone

Description automatically generated

Slika prikazuje korisničku navigaciju admin korisnika: izvor: autor

Ukoliko korisnik nije administrator, mogućnost navigacije na stranicu za postavke nije vidljiva.

A screenshot of a phone

Description automatically generated

Slika prikazuje korisničku navigaciju običnog korisnika: izvor: autor

Ukoliko se aplikacija koristi na manjem ekranu, navigacijska komponenta je drugačija. Lista stranica zamijenjena je padajućim izbornikom.

A blue background with a black rectangle and a crescent moon

Description automatically generated

Slika prikazuje navigaciju na manjim ekranima: izvor: autor

U sklopu navigacije moguće je mijenjati temu aplikacije na svjetlo, tamno ili prema postavkama odabranim na korisnikovom uređaju.

A screenshot of a computer

Description automatically generated

Slika prikaz padajući izbornik za mijenjanje teme: izvor: autor

## Početna stranica

Pristupanjem početne stranice aplikacije, korisnik se susreće s porukom dobrodošlice, te gumbom za prijavu.

A screenshot of a computer

Description automatically generated

Slika prikazuje početnu stranicu web aplikacije: izvor: autor

Kad je korisnik prijavljen, prikazana je samo poruka dobrodošlice, bez tipke za prijavu.

## Prijava i odjava

Navigacijom na stranicu za prijavu, korisnik se susreće sa formom unutar kartice. Forma sadrži polja sa podacima potrebnim za prijavu.

A screen shot of a login form

Description automatically generated

Slika prikazuje stranicu za prijavu: izvor: autor

Sve forme aplikacije imaju provjeru na korisničkom sučelju, sa porukama u slučaju greške.

A screenshot of a login screen

Description automatically generated

Slika prikazuje primjer validacije na korisničkom sučelju: izvor: autor

Nakon unosa podataka i klika gumba za slanje, provjera se odrađuje i na *backendu*. U slučaju uspješne prijave, korisnik dobiva notifikaciju u donjem desnom kutu, te je proslijeđen na početnu stranicu.

A blue background with white text

Description automatically generated

Slika prikazuje primjer notifikacije u slučaju uspješnog izvršenja akcije: izvor: autor

U slučaju greške kod provjere na *backendu*, korisnik također dobiva notifikaciju drugačijeg izgleda, sa porukom o grešci.

A screenshot of a login screen

Description automatically generated

Slika prikazuje primjer notifikacije u slučaju ne uspješnog izvršenja akcije: izvor: autor

Ovaj način provjera formi i obavještavanja korisnika o uspješnosti, primijenjen je na svim stranicama. Uspješnom prijavom postavlja se autorizirano stanje aplikacije.

A computer screen with text

Description automatically generated

Slika prikazuje postavljanje autoriziranog stanja: izvor: autor

## Funkcionalnosti

Prije kreiranje naloga, prvo se kreira i postavlja zadani status koji se postavlja prilikom kreiranja naloga. Ukoliko ni jedan status nije kreiran, prikazana je poruka koja o tome obavještava korisnika. Uz poruku, nalazi se gumb koji vodi na stranicu za kreiranje statusa.

A screenshot of a computer

Description automatically generated

Slika prikazuje Orders stranicu bez kreiranih statusa: izvor: autor

Kada postoji status, ali nije postavljen zadani status, prikazana je drugačija poruka. Uz poruku, nalazi se gumb koji vodi na stranicu za postavljanje zadanog statusa. Stranici za postavljanje zadanog statusa mogu pristupiti samo administratori.

A black background with white text

Description automatically generated

Slika prikazuje Orders stranicu bez postavljenog početnog statusa za naloge: izvor: autor

Kada je postavljen zadani status, moguće je kreirati naloge, filtrirati naloge i pretraživati naloge.

A screenshot of a computer

Description automatically generated

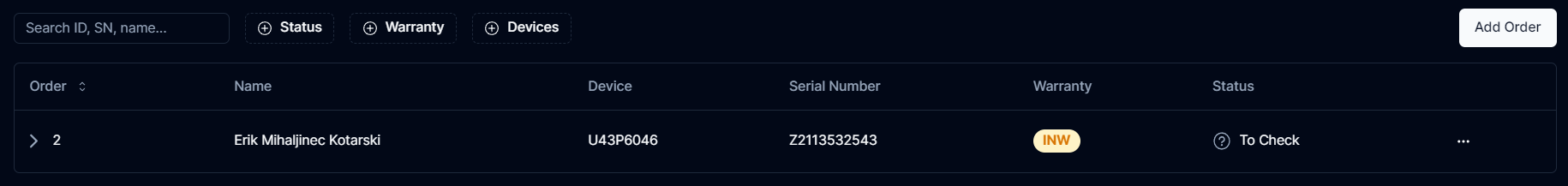
Slika prikazuje Orders stranicu koja je spremna za upotrebnu: izvor: autor

Gumb sa nazivom „Set Default Order Status“ vide korisnici sa administratorskim ovlastima, te vodi na stranicu za postavljanje zadanog statusa. Kada se miš drži iznad gumba sa oznakom za informacije, navedeni su uvjeti za upotrebu i polja nad kojima je moguće raditi pretragu naloga.

A screenshot of a phone

Description automatically generated

Slika prikazuje uvjete za korištenje i polja za pretragu nad nalozima: izvor: autor

Tablica za naloge ima proširive retke. Kada je redak ne proširen, prikazane su najbitnije informacije.

Slika prikazuje primjer kreiranog naloga sa ne proširenim retkom: izvor: autor

Pritiskom na ikonu u zadnjem stupcu retka, otvara se padajući izbornik sa dodatnim akcijama.

A screenshot of a computer

Description automatically generated

Slika prikazuje dodatne akcije za redak naloga: izvor: autor

Klikom na „Finish“ gumb otvara se dijalog za završavanje naloga. Unutar dijaloga postavlja se status, potrošeni materijal i njegova količina. Materijal je filtriran po uređaju u nalogu. Količina je automatski oduzeta sa stanja. U slučaju pogreške, prikazana je notifikacija korisniku.

A screenshot of a computer

Description automatically generated

Slika prikazuje dijalog za završavanje naloga: izvor: autor

Ostale informacije nalaze se u proširenom retku. Naloge je moguće filtrirati po statusu, jamstvu i uređajima. Svaki element za filtriranje ima paginaciju i polje za pretragu. Pretraživanjem u filteru ili promjenom paginacije u filteru, šalje se upit na *backend* za taj filter.

A screenshot of a computer

Description automatically generated

Slika prikazuje primjer proširivog retka i filtera za naloge: izvor: autor

Klikom na gumb za kreiranje naloga, otvara se dijalog koji ima više koraka. Polja koja su obavezna imaju validaciju. Neka polja imaju dodatnu validaciju, poput polja za OIB (osobni identifikacijski broj).

A screenshot of a phone

Description automatically generated

Slika prikazuje prvi korak dijaloga za kreiranje naloga sa validacijom: izvor: autor

A screenshot of a device

Description automatically generated

Slika prikazuje drugi korak dijaloga za kreiranje naloga sa validacijom: izvor: autor

Nakon kreiranja naloga, automatski je ažuriran prikaz naloga na stranici. U slučaju da nema kreiranog uređaja, prikazano je drugačije polje kod odabira uređaja.

A screenshot of a phone

Description automatically generated

Slika prikazuje polje za uređaj kod kreiranja naloga kada nije kreiran ni jedan uređaj: izvor: autor

Stranice koje nisu za profil ili postavke imaju sličan raspored. Sadrže tablicu sa pretragom, filterima (ukoliko je potrebno) i gumbom za kreiranje.

A screenshot of a computer

Description automatically generated

Slika prikazuje Devices stranicu sa prikazom Devices taba: izvor: autor

Sadrže *tabove* koji su potrebni za kreiranje nekog objekta, npr. za kreiranje uređaja potrebno je izabrati brand i tip uređaja.

A screenshot of a device

Description automatically generated

Slika prikazuje primjer forme za unos uređaja: izvor: autor

Stoga, *tabovi* su „Brand“ i „Device Types“ u kojima se kreiraju objekti potrebni za kreiranje uređaja.

A screenshot of a computer

Description automatically generated

Slika prikazuje Devices stranicu sa prikazom Brand taba: izvor: autor

„Materials“ i „Status“ stranice prate istu shemu.

A screenshot of a computer

Description automatically generated

Slika prikazuje Materials stranicu sa prikazom Materials taba: izvor: autor

Na *backendu* je implementirana vlastita provjera jedinstvenosti za unos materijala i uređaja. Sa slike 57 vidi se da je kreiran materijal pod nazivom „MAKITLB43“ čiji je tip „LED“. Ukoliko se pokuša kreirati uređaj sa istim nazivom (neosjetljivo na velika i mala slova) i tipom prikazana je greška.

A red rectangular sign with white text

Description automatically generated

Slika slika prikazuje primjer obavijesti u slučaju greške kod provjere jedinstvenosti: izvor: autor

*Tab* za tip statusa nema gumb za kreiranje novog tipa. Mogućnost kreiranja novog tipa ima samo super admin.

A screenshot of a computer

Description automatically generated

Slika prikazuje Status stranicu sa prikazom Status Type taba: izvor: autor

Na profilnoj stranici moguće je mijenjati vlastito korisničko ime, zaporku i osobne informacije.

A screenshot of a computer

Description automatically generated

Slika prikazuje profilnu stranicu za promjenu korisničkog imena: izvor: autor

A screenshot of a computer

Description automatically generated

Slika prikazuje profilnu stranicu za promjenu zaporke: izvor: autor

Za promjenu osobnih podatak nužno je ispuniti barem jedno od ponuđenih polja.

A screenshot of a computer

Description automatically generated

Slika prikazuje profilnu stranicu za promjenu osobnih podataka: izvor: autor

Na stranici za postavke upravlja se korisnicima i postavlja zadani status naloga. Administratori imaju mogućnost pregleda, kreiranja, brisanja i naknadne promjene podataka korisnika. Pravo na promjenu podataka administratora i sl. posjeduje samo super admin.

A screenshot of a computer

Description automatically generated

Slika prikazuje tablicu korisnika: izvor: autor

A screenshot of a computer

Description automatically generated

Slika prikazuje formu za kreiranje korisnika: izvor: autor

A screenshot of a computer

Description automatically generated

Slika prikazuje postavljanje zadanog statusa za naloge: izvor: autor

## Sigurnost

Zaštita stranica od neovlaštenog pristupa provedena je u rasporedu „Dashboard“ rute. Ostale stranice poput „Devices“ i „Orders“ nasljeđuju raspored.

A screen shot of a computer code

Description automatically generated

Slika prikazuje raspored koji nasljeđuju ugniježđene Dashboard rute: izvor: autor

Komponenta „RequireAuth“ sadrži logiku za provjeru autorizacijskog stanja aplikacije uz pomoć Redux Toolkita.

A screen shot of a computer program

Description automatically generated

Slika prikazuje provjeru za autorizaciju: izvor: autor

U slučaju neovlaštenog pristupa, prikazana je posebna stranica.

A screenshot of a computer

Description automatically generated

Slika prikazuje stranicu u slučaju pokušaja pristupa neprijavljenog korisnika na neku od Dashboard stranica: izvor: autor

Ista logika zaštite implementirana je za stranice sa određenom razinom dozvole (npr. stranicu za postavke mogu pristupiti samo administratori).

A screenshot of a computer

Description automatically generated

Slika prikazuje stranicu u slučaju pokušaja pristupa običnog korisnika upravljačkim stranicama: izvor: autor

# Zaključak

Razvoj novih tehnologija izuzetno je brz. Naspram nekoliko godina unazad već postoje puno bolja rješenja. Koristeći najnovije tehnologije aplikacija posjeduje znatno bolje performanse, te bržu i jednostavniju dostavu proizvoda korisniku. Današnji alati pružaju vrlo brz razvoj aplikacija sa mnogim već unaprijed konfiguriranim postavkama. Potiču dobre kodne standarde za lakšu čitljivost i iskoristivost. S obzirom da svi koriste mobitele, responzivnost je također uzeta u obzir prilikom razvoja.

Sigurnost je jedna od najbitnijih značajaka svake aplikacije. Današnji alati unaprijed podese većinu postavki, te je potrebna samo vlastita prilagodba za početak korištenja. Time se smanjuje mogućnost ljudske pogreške prilikom razvoja. Neadekvatna sigurnost može prouzročiti curenje osobnih podataka korisnika i klijenata. U krajnjem slučaju i ugroziti samo poslovanje.

U današnjem svijetu prepunom raznim informacijama, baza podataka ima jednu od ključnih uloga. Potrebno je osigurati adekvatnu sigurnost pristupa i spremanje pričuvnih podataka u slučaju izvanrednih situacija.

Aplikacija je modularna, nudi više načina raspoređivanja, te ima ugrađene sigurnosne značajke i mogućnost prilagodbe za krajnjeg korisnika. Integracija sa DevOps alatima poput CI/CD pruža dodatnu sigurnost za dostavu kvalitetnijeg proizvoda korisniku.

**Literatura**

|  |  |
| --- | --- |
| [1] | S. Dauzon, A. Bendoraitis i A. Ravindran, »Django: Web Development with Python,« Birmingham, UK, Packt Publishing, 2016, p. 4. |
| [2] | P. Barry, »Head First Python,« *A Brain-Friendly Guide*, Sebastopol, CA, O’Reilly Media, 2016, p. 10. |
| [3] | W. S. Vincent, »Django for APIs,« *Build web APIs with Python & Django*, WelcomeToCode, 2020, p. 1. |
| [4] | A. Banks i E. Porcello, »Learning React, 2nd Edition,« *Modern Patterns for Developing React Apps*, Sebastopol, CA, O'Reilly Media, 2020, p. 66. |
| [5] | J. Turnbull, »The Docker Book,« *Containerization is the new virtualization*, Turnbull Press, 2014, p. 7. |
| [6] | A. Martins, »Git(Hub) Flow, Trunk Based Development, and Code reviews,« reviewpad, 17.6.2021. [Mrežno]. Dostupno: https://reviewpad.com/blog/github-flow-trunk-based-development-and-code-reviews. [Pokušaj pristupa 29 Kolovoz 2023]. |
| [7] | M. Wanyoke, »How to Use Git Branches & Buddy to Organize Project Code,« sitepoint, 9.7.2019. [Mrežno]. Dostupno: https://www.sitepoint.com/use-git-branches-buddy/. [Pokušaj pristupa 29 Kolovoz 2023]. |