UNIVERZA V MARIBORU
FAKULTETA ZA ELEKTROTEHNIKO,
RAČUNALNIŠTVO IN INFORMATIKO

Jernej Ofič, Domen Perko, Gregor Sulcer, Matevž Semprimožnik

DOKUMENTACIJA
ODLOČITEV

Tehnična dokumentacija

TechBuild
Maribor, maj 2023

KAZALO VSEBINE

1 ARHITEKTURNE ODLOČITVE

1. 1 Arhitektura rešitve

Aplikacijo sestavlja tehnološki sklad:

- Next.js
- tRPC
- TypeScript
- PlanetScale
- Vercel
- Tailwind CSS
- Prisma
- Clerk

1. 1. 1 Next.js

Next.js je močno orodje za razvoj spletnih aplikacij, ki temeljijo na Reactu. Ponuja izboljšano, enostavnejše in zmoglivejše usmerjanje na večstranskih aplikacijah. Vgrajena je podpora za dinamično usmerjanje in prerisovanje komponent, ki omogoča boljšo uporabniško izkušnjo. Omogočeno je tako SSR in SSG. S tem izboljšamo hitrost nalaganja strani in SEO. Vključenih je več funkcionalnosti za optimizacijo spletnih strani (npr. optimizacija prenosa in predponjenje strani). Next podpira tudi API usmerjanje, ki omogoča enostavno definiranje API-jev in poenostavi komunikacijo med odjemalcem in strežnikom. Omogoča ustvarjanje lastnih API-jev ali uporabo obstoječih za vzpostavitev hitrega in zanesljivega podatkovnega toka. Vključen je sistem za zajemanje napak, ki pripomore pri odrivanju in odpravljanju napak v aplikacijah. Prav tako je ogrodje priljubljeno in podporto s strani široke skupnosti razvijalcev. Obstaja obsežna dokumentacija, vodiči in vzorčni projekti, ki nam pomagajo hitro začeti in reševati morebitne težave.

1. 1. 2 TypeScript

TypeScript je programski jezik, ki predstavlja razširitev JavaScripta z možnostjo statične tipizacije. To pomeni, da lahko pred izvajanjem programa preverimo, ali so spremenljivke, funkcije in objekti pravilno uporabljeni glede na njihove tipe, kar lahko zmanjša število napak in pohitri razvoj. Izbrali smo ga zaradi dolgoročne vzdržljivosti projekta. TypeScript zagotavlja večjo varnost pri razvoju z manjšim številom napak, saj preprečuje nepričakovane vrednosti, ki bi lahko sprožile napake pri izvajanju. Poleg tega olajša razumevanje kode, saj je zaradi tipizacije lažje prepoznati namen posameznih delov kode. Podpira dobre prakse razvoja sodobnih aplikacij. TypeScript ima tudi veliko podporo razvijalcev in skupnosti, zato obstaja veliko orodij in knjižnic, ki olajšajo razvoj v jeziku. Čas razvoja se z uporabo jezika zmanjša, saj

je lažje prepoznati in odpraviti napake. Prav tako omogoča lažjo refaktorizacijo kode, saj zaradi tipizacije lažje prepoznamo, kje so potrebne spremembe.

1. 1. 3 tRPC

tRPC (TypeScript RPC) je odprtokodna knjižnica za izgradnjo odjemalcev in strežnikov za oddaljene klice (RPC) med programskimi jeziki. Prinaša številne prednosti. Je zelo lahek in enostaven za uporabo. Z uporabo tRPC-ja enostavno ustvimo RPC storitev, ki se lahko uporablja na več platformah, saj podpira različne protokole prenosa podatkov. Poleg tega je zelo učinkovit, kar pomeni, da lahko aplikacija hitro in učinkovito odgovori na zahteve. Vgrajene so funkcije za ročno upravljanje z napakami in podatkovnimi pretvorbami, kar poenostavlja razvoj. Poleg tega tRPC omogoča enostavno uporabo avtentikacije in avtorizacije. Podpira različne modele podatkov, vključno s težkimi objektno-relacijskimi modeli (ORM) in grafičnimi modeli, kar pomeni, da se lahko prilagodi skoraj vsem potrebam našega projekta. Omogoča tipsko varno komunikacijo s strežnikom na osnovi TypeScripta. Zasnovan je z uporabo sodobnih razvojnih praks, kot so funkcionalne komponente in generiki. To omogoča enostavno in pregledno razvojno izkušnjo, ki omogoča hitro in učinkovito odzivanje na spremembe. tRPC je aktivno vzdrževan in ima veliko skupnost podpornikov in razvijalcev.

1. 1. 4 Tailwind

Tailwind CSS je odprtokodna knjižnica CSS, ki omogoča, da hitro in enostavno oblikujemo spletna mesta z uporabo predhodno definiranih razredov CSS. Tailwind omogoča izredno prilagodljivost in hitrost razvoja. Tailwind je zasnovan tako, da je enostaven za prilagajanje in razširitev. Omogoča nam kreiranje svojih razredov in definiranje globalnih spremenljivk. Knjižnjica je lahka in ne upočasnjuje naše aplikacije. Olajšan je tudi razvoj odzivnih aplikacij za vse širine naprav.

1. 1. 5 Vercel

Vercel je platforma za gostovanje in razvoj aplikacij, prednosti so enostavnost, hitrost in zmogljivost. Zasnovan je tako, da je enostaven za uporabo. Nudi intuitivno uporabniško izkušnjo, ki omogoča hitro in enostavno nalaganje aplikacij in datotek ter enostavno upravljanje z njimi. Je zelo hitra platforma, ki omogoča hitro nalaganje spletnih strani in aplikacij, to pripomore k boljši uporabniški izkušnji. Omogoča prilagajanje okolja in nastavitev, ki so potrebne za našo aplikacijo.

1. 1. 6 PlanetScale

PlanetScale je platforma za upravljanje podatkovnih baz, ki je preprostosta, zmogljiva in skalabilna. Omogoča enostavno skaliranje podatkovne baze. To je zelo pomembno, če bo aplikacija rastla in se bo število uporabnikov povečevalo. PlanetScale je zasnovan tako, da zagotavlja hitro in učinkovito delovanje podatkovne baze. To je pomembno za zagotavljanje hitrega delovanja aplikacije in za izboljšanje uporabniške izkušnje. Vgrajene so funkcije za varnost in zaščito podatkovne baze. Uporablja MySql podatkovno bazo.

1. 1. 7 Prisma

Prisma je ORM (Object-Relational Mapping) in platforma za dostop do podatkov. Ponuja preprost vmesnik, ki olajša uporabo in integracijo s katero koli tehnologijo. Sintaksa je jasno definirana in je intuitivna za uporabo in razumevanje. Omogoča hitrejše in učinkovitejše dostopanje do podatkovne baze in ustvarjanja poizvedb brez potrebe po ročnem pisanju SQL poizvedb. To zmanjšuje čas razvoja in zagotavlja hitrejšo dostavo aplikacije. Prisma uporablja enotni model podatkov, ki olajša prenos podatkov med različnimi platformami in storitvami. To omogoča boljšo modularnost in skalabilnost aplikacije. Zagotavlja varnost podatkov, saj uporablja pripravljene poizvedbe in pripravljene ukaze, ki preprečujejo SQL vbrizgavanje. Ponuja tudi možnosti zaščite podatkov in dovoljenj, kar pomaga pri zagotavljanju, da se podatki hranijo na varnem. Skaliranje aplikacije je enostavno.

1. 1. 8 Clerk

Clerk je platforma za avtentikacijo in upravljanje uporabniških računov. Zagotavlja enostavno integracijo avtentikacije v aplikacijo. Z uporabo lahko enostavno dodamo funkcionalnosti za registracijo, prijavo in ponastavitev gesla. Ponuja sodobno uporabniško izkušnjo za avtentikacijo in upravljanje uporabniških računov, ki jo lahko enostavno prilagodimo. Zagotavlja visoko raven varnosti pri upravljanju uporabniških podatkov. Ponuja vse varnostne funkcije, kot so varnostno kopiranje podatkov, šifriranje in tokenizacija.

2 ORGANIZACIJA DELA

Za organizacijo dela bomo uporabili metodo Scrum.

Scrum je agilna metoda upravljanja projektov, ki smo jo že uporabljali v preteklih projektih in se je izkazala za učinkovito. S Scrumom lahko učinkovito razdelimo delo med ekipo, določimo prioritete nalog in dosežemo cilje projekta v časovnih okvirih. Poleg tega Scrum spodbuja tesno sodelovanje med razvojno ekipo, kar zagotavlja, da se izdelek razvija z jasnimi zahtevami in pričakovanji.

Po Scrumu bomo delali v iteracijah, ki jih imenujemo sprinti. Sprinti bodo trajali teden dni. Na začetku vsakega sprinta bomo imeli sestanek, imenovan Sprint planning, na katerem se bomo dogovorili o ciljih sprinta, določili naloge, ki jih je treba izvesti, in določili časovni okvir za dosego teh ciljev.

Med sprintom se bomo vsakodnevno srečevali na sestankih, imenovanih Daily scrum, na katerih bomo pregledali, kaj smo že opravili, kaj moramo še opraviti in kje smo morda naleteli na ovire. To nam bo omogočilo, da se hitro odzovemo na morebitne težave in zagotovimo, da ostajamo na pravi poti za dosego ciljev sprinta.

Na koncu sprinta bomo imeli sestanek, imenovan Sprint review, na katerem bomo pregledali dosežke sprinta, preverili, ali smo dosegli cilje, in se pogovorili o morebitnih izboljšavah, ki jih lahko uvedemo v prihodnjih sprintih. Poleg tega bomo imeli tudi sestanek, imenovan Sprint retrospective, na katerem se bomo pogovorili o tem, kaj smo se naučili iz prejšnjega sprinta in kaj bi lahko izboljšali v prihodnosti.

Za komunikacijske namene bomo uporabili platformo Discord. Naloge bomo vodili s pomočjo orodja Trello.

3 ZAGOTAVLJANJE KAKOVOSTI

Za zagotavljanje kakovosti rešitve bomo uporabljali več pristopov. Naša ekipa bo redno preverjala kodo. V primeru, da bi se pojavili kakršni koli problemi ali napake, bomo hitro ukrepali in jih odpravili. Naša ekipa bo tudi redno pregledovala in izboljševala postopke razvoja in testiranja, da bomo zagotovili, da je naša metoda razvoja čim bolj učinkovita in zanesljiva.

3. 1 Unit testiranje

Za komponente naše aplikacije bomo napisali unit teste s pomočjo Jest-a. Uporaba Jesta za testiranje Next.js aplikacije je dobra izbira, saj je Jest enostaven za namestitev in uporabo, ima dobro integracijo z Next.js, je hiter in učinkovit, ponuja lepo oblikovana poročila o testiranju, podpira različne vrste testov in omogoča preprosto konfiguriranje.

3. 2 Spremljanje napak

Napake v kodi bomo spremljali s pomočjo Axiom-a. Axiom je orodje za spremljanje napak in beleženje dogodkov v aplikacijah. S pomočjo Axioma lahko spremljamo in analiziramo različne vrste napak, vključno z napakami v kodi, težavami s konfiguracijo in drugimi težavami, ki lahko vplivajo na delovanje aplikacije. Poleg tega lahko Axiom zazna tudi druge dogodke, kot so zahteve API-jev, obremenitve strežnikov, operacije v bazi podatkov in podobno.

3. 3 Tehnični dolg

Za odpravljanje tehničnega dolga bomo uporabili SonarCloud. SonarCloud je orodje za upravljanje kakovosti kode in vodenje tehničnega dolga. S pomočjo SonarClouda lahko odkrijemo težave v kodi, kot so podvajanje kode, slaba berljivost kode, težave z varnostjo, kritična tveganja in druge težave.