TEHNIČNA DOKUMENTACIJA

za

Julian Alps Virtual Run

Matevž Mak

Koper, 24.05.2021

Kazalo vsebine

1. Definicija naloge 3
2. Opis zahtev 3
2.1 Uporabniške zahteve 3
2.2 Tok podatkov 4
3. Podatkovni model 4
4. Vmesniki 5
4.1 Vmesnik registracije in prijave 5
4.2 Ponastavitev gesla 5
4.3 Nadzorna plošča tekmovanja 5
4.4 Multijezičnost 5
5. Uproabljene knjižnice 6
5.1 ExpressJS 6
5.2 Passport 6
5.3 Mongoose 6
5.4 Ngx translate 6
6. Zunanje povezave 6
6.2 Strava 6
6.2 Mapbox in Leaflet 7
7. Produkcija 7

1. Definicija naloge

Julian alps virtual run je namenjena vsem tekmovalcem, ki se niso mogli udeležiti dogodka v živo, ker je bil preklican zaradi COVID-19. Gre za online izvedbo športnega dogodka, kjer tekmovalce vnašajo svoje tekaške podatke na platformo in tekmujejo z drugimi. V pogled pa imajo tudi zemljevid za posamezno traso, kjer lahko vidijo svoj napredek, če bi dejansko tekli na dogodku.

2. Opis zahtev

Za izdelavo sem uporabil MEAN stack, kjer lahko zelo enostavno:

- Definiramo komponente naše platforme
- Predstavimo dinamiče podatke
- Definiramo storitve
- Uporabljamo NPM knjižnice
- Manipuliramo izgled platforme
- Definiramo REST API
- Modularno implementiramo celoten projekt

MEAN stack je sestavljen iz podatkovne baze MongoDB, express-a, Angularja in Nodejs. Celoten projekt je razdeljen na dva dela. Prvi del je frontend, kjer uporabljamo framework Angular, ki je odličen za manipulacijo videza spletne strani in definiranje storitev, ki kličejo podatke na naš API. Ostale komponente MEAN stacka spadajo pod backend. S pomočjo Node.js smo definirali API, ki pobira podatke iz podatkovne baze MongoDB. Baza je gostovana na oblačni platformi mongoAtlast, aplikacija pa na oblačni platformi digital Ocean, kjer lahko konfiguriramo svoj linux server po svojih željah.

2.1 Uporabniške zahteve

Celoten sistem je namenjen športnim navdušencem, predvsem tekmovalcev, ki uživajo v teku po prečudoviti naravi Julijskih Alp. Vsak uporabnik lahko kreira svoj uporabniški račun ob pridobitvi aktivacijske kode na e-naslov.

Uporabnik nato jasno vidi potek tekmovanja za posamezno traso, ki so razvrščene po zavihkih. Uporabnik mora jasno viditi svoj napredek na zemljevidu in zahtevnost proge. Da lahko zaključi posamezno traso mora opraviti višinsko vzpon trase in pa razdaljo.

Uporabnik ima na volje več načinov vnosa podatkov. Svoje podatke lahko vnaša ročno, lahko naloži GPX datoteke iz svojih sledilnih naprav ali pa se poveže z zunanjo platformo Strava. Strava je odlična mobilna aplikacija, ki beleži teke, ki jih opravimo doma. Uporabnik lahko nato na naši platformi izbere posamezno aktivnost, ki jo je opravil doma in jo je zabeležila Strava.

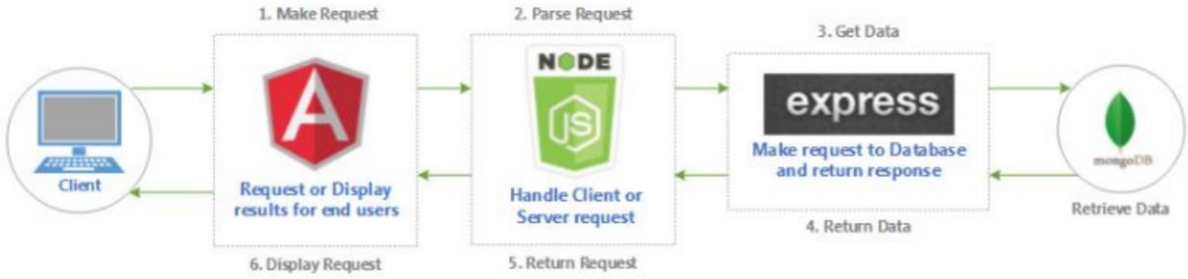
Za vsako traso uporabnik vidi svojo konkurence in njihove čase, opravljeno razdaljo in vzpone.

Če uporabnik pozabi geslo, si ga lahko ponastavi s pomočjo ponastavitvene strani.

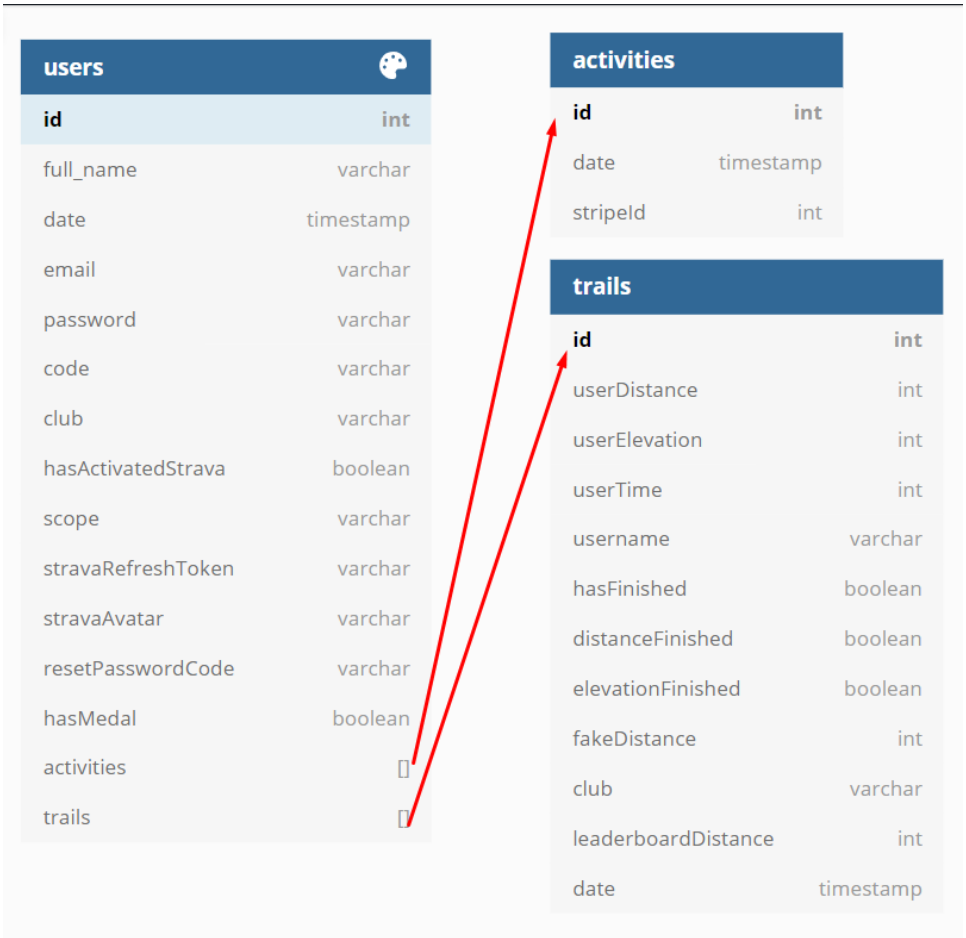
Ko uporabnik zaključi posamezno traso, se mu mora prikazati grafični vmesnik, kjer se mu čestita in ponudi možnost, da prične znova. Hkrati pa mora aplikacija tudi obvestiti organizatorje preko e-maila, da je tekmovalec zaključil s traso in da mu lahko pošljejo nagrado.

2.2 Tok podatkov

Celotna platforma je sestavljena iz statičnih in dinamičnih komponent. Statične komponente so komponente, ki ostanejo vidne ne glede na operacijo uporabnika. Celotna aplikacija posluša na evente oz. Dogodke uporabnika. Ko se zgodi nek dogodek v aplikaciji, se potem sproži storitev, ki pošlje zahtevo na server. Te zahteve so lahko različnega tipa GET, POST, PUT, DELETE. Express nato procesira podatke in poišče podatke v podatkovni bazi in jih vrne clientu. Ti podatki pa se nato tudi dinamično predstavijo uporabniku.



3. Podatkovni model



Baza je bila deljena na tri zbirke, in sicer na users, activities in trails. Zbirka user vsebuje activites in trails, kjer lahko shranjujemo dokumente drugih zbirk, kot svoje otroke. Torej, če želimo dobiti posamezne aktivnosti in trase za posameznega uporabnika moramo populatati te dve vrednosti.

Zbirka »users« vsebuje vse informacije o uporabniku, ki so potrebni za avtentikacijo. Poleg pa vsebuje še spremenljivke za komunikacijo z zunanjo aplikacijo strava (strava refresh token). Prav tako vsebuje vrednsot »hasMedal«, s tem beležimo ali je uporabnik že opravil kakšno traso, da ne dobi dvojne nagrade s strani organizatorjev.

Zbirka »activities« vsebuje id posameznega dokumenta in pa ID strave aktivnosti. Pomembno je, da beležimo izbrane aktivnosti od Strava, kajti ne želimo, da bi uporabnik lahko ponovno izbral isto aktivnost (svoj najboljši tek, ki je pogojen z dnevno formo tekača).

4. Vmesniki

4.1 Vmesnik registracije in prijave

Uporabnik mora pri prijavi ali registraciji izbrati vsa polja, v nasprotnem primeru dobi obvestilo, da je prijava/registracija neveljavna. Na strežniku se nato preverijo vsi robni pogoji npr. Če se geslo ujemo v bazi ali če že uporabnik s tem email naslovom že obstaja v primeru registracije. V primeru, da ne zadostimo robnih pogojev, server vrne odgovor, da registracija/prijava ni bila uspešna in to prikažemo uporabniku.

4.2 Ponastavitev gesla

Ponastavitev gesla je sestavljena iz dveh komponent. V prvi komponenti uporabnik vstavi svoj veljaven email naslov, ki ga je uporabil pri registraciji. V primeru, da je email veljaven, uporabnik prejme ponastavitveni link na ta email. Ta link vsebuje v url naslovu unikaten zapis, ki ga povezuje z računom. V tej komponenti nato izbere novo geslo in ga potrdi s potrditvenim geslo. Tukaj pogledamo, če se gesli ujemata in pošljemo zahtevo na server o spremembi gesla. Ob uspešni ponastavitvi nato uporabnik prejme novi email, kjer dobi potrjeno sporočilo o spremembi gesla.

4.3 Nadzorna plošča tekmovanja

Ob vstopu v aplikacijo uporabnik vidi številne trase, ki so porazdeljene po zavihkih. »Rendiranje« celotnih zemljevidu opravi client. Ob vstopu se nato tudi prenesejo vse informacije o trenutnih dosežkih posameznega tekmovalca, torej za vse trase kolikšno razdaljo je že pretekel, koliko vzpona je že opravil in trenuten čas. Na podlagi pridobljenih podatkov in izbrani trasi, nato prikažemo napredek na zemljevidu in v sekciji, kjer je statistika. V zažetni fazi uporabnik še nima povezan Strava račun z našo aplikacijo. V primeru da se želi povezati, mu je na voljo gumb, ki ga preusmeri na prijavo za strava račun in potem potrdi kakšne podatke želi deliti z našo aplikacijo. V primeru, da uporabnik želi deliti vse podatke z nami, zapišemo refresh token v bazo za prihajajoče vnose. Uporabnika se nato preusmeri znova na nadzorno ploščo, kjer pa že lahko izbira med svojimi aktivnosti, ki jih je opravil s Strava aplikacijo. Uporabnik lahko tudi naloži GPX datoteko, ki jo nato preveri GPX parser in iz datoteke pridobi čas, razdaljo in višinske vzpone in jih posodobi v podatkovni bazi. Zadnja opcija pa je ročni vnos, ki je najbolj preprost, kjer uporabnik vnese razdaljo, višinski vzpon in čas. Uporabnik lahko nato vidi tudi tabelo vseh tekmovalcev in vidi kje se nahaja.

4.4 Multijezičnost

Uporabnik lahko tudi v kateremkoli trenutku zamenja jezik v platformi (Slovenščina ali Angleščina). Vsi prevodi so zapisani v JSON datotekah in glede na izbran jezik, se prebere temu primerna datoteka. V htmlju so zaradi tega vpisane spremenljivke namesto texta in s pomočjo pip-ov spremenimo spremenljivko v text, glede na izbran jezik.

5. Uproabljene knjižnice

Skozi projekt sem uporabil številne knjižnice, ki so bile uporabljene za Angular framework ali nodeJS. Vse knjižnice se shranaje v datoteko package.json, tako lahko sistem prenesemo na katerokoli napravi in izvedemo ukaz npm install, ki naloži knjižnice na napravo.

5.1 ExpressJS

Express je ena od najbolj uporabljenih knjižnic pri razvoju Node.js aplikacij. V le nekaj vrsticah lahko generiramo lokalni server aplikacije. Je zelo prilagodljiva in minimalistična knjižnica. Uporaba nam omogoča implementacijo robustnih poti oz. URL naslovov, na katere »posluša aplikacija«. Prav tako nam omogoča posredovanje podatkov preko http klicev. Podatke lahko dobimo preko url naslova ali preko forme v HTML dokumentu. Ko podatki pridejo na definirano pot, lahko v tej poti izvedemo številne funkcije in operacije. Nato lahko posodobljene podatke dinamično prikažemo s funkcijo res.render(), v katero lahko vstavimo objekt, ki vsebuje posredovane spremenljivke, ki nosijo poljubne vrednosti.

5.2 Passport

Passport knjižnica je ena najbolj uporabljenih knjižnic za avtentikacijo uporabnika v sistem. Je fleksibilna knjižnica, kjer lahko definiramo mnogo strategij, preko katerih lahko opravljamo proces registracije in prijave. Za registracijo in prijavo lahko uporabljamo socialne platforme, kot so Google plus, Facebook, Twitter, ki jih definiramo v svojih strategijah. Ko se uporabnik prijavi v sistem, shranimo njegov unikaten ID v sejo. Ko uporabnik izvaja operacije v sistemu, pride do številnih obiskov poti na strežnikovi strani, kjer uporabimo njegov ID tako, da v vsakem trenutku vemo, kateri uporabnik izvaja specifične operacije.

5.3 Mongoose

Uporaba knjižnice Mongoose nam omogoča, da se povežemo na podatkovno bazo. Gre za orodje, ki nam pomaga pri modeliranju MongoDB objektov, ki so v JSON obliki. Ukazi se izvajajo asinhrono. Pri projektu smo definirali številne modele oz. sheme, in sicer, kakšne atribute mora vsebovati specifičen objekt. Pri tem je pomagala Mongoose knjižnica. Mongoose omogoča eleganten način poizvedovanja podatkov iz podatkovne baze. Vsak klic poizvedovanja vsebuje povratno funkcijo, kjer preverimo, ali je prišlo do napake.

5.4 Ngx translate

Knjižnica je bila uporabljena v Angular frameworku, kjer lahko definiramo json datoteke in v njih definiramo prevode v spremenljivke. S pomočjo pipe operatorja lahko te spremenljivke v HTML datotekah spremenimo v text.

6. Zunanje povezave

6.2 Strava

Potrebno je bilo prebrati dokumentacijo, kako lahko povežemo implementiramo aplikacijo z zunanjo platformo strava. Z veljavnim klicom prejemo refresh token, ki se skozi čas spreminja. Ob prijavi uporabnika preverimo, če je ta refresh token še veljaven, v primeru, da ni dobimo novi refresh token. Z refresh tokenu lahko nato opravimo klic na Strava API, kjer dobimo access token s katerim lahko pridobimo aktivnosti uporabnika.

6.2 Mapbox in Leaflet

Za zemljevid sem uporabil knjižnico Leaflet, ki je brezplačna. Knjižnica je zelo modularno zgrajena in nam ponuja številne dodatke npr, markerji, oblački. Ker pa je bila podlaga za mapo precej staromodna smo se nato odločili za povezavo z Mapbox platformo, kjer smo pridobili njihov ključ in s tem uporabili njihove teme za vizualno izboljšavo zemljevida.

7. Produkcija

Za development mode, smo uporabljali github + oblačno platformo Heroku. V primeru, da smo posodobili master vejo, se je projekt že avtomatsko gradil na oblačni platformi Heroku. Heroku je brezplačna platforma, ampak v primeru MEAN stacka, je potrebno imeti backend in frontend v isti mapi, kar pa ni ravno pravilno. Dodaten minus pri uporabi Heroku platforme je bila tudi prvotna hitrost. Ko je uporabnik dostopal do aplikacije so se morali vzpostaviti Heroku dynoti, da ne tratijo CPU ciklov 24/7. Za test smo tudi poizkusili premium verzija, vendar še vedno slaba inicialna hitrost (izguba 4-6 sekund). Nato smo iskali novo rešitev, kjer smo izbrali digital ocean. Digital ocean ponuja prazne linux serverje, ki si jih lahko konfiguriramo po svoji željah. Dodali smo domeno in poddomeno za backend REST API. Kar je najbolj pomembno ločili smo backend in frontend, hkrati pa tudi izboljšali začetno hitrost.