Univerzitet Singidunum

Tehnički fakultet

Veb aplikacija za vođenje evidencije o potrošnji goriva

- Projektni rad -

Praktikum - Sistemi elektronskog poslovanja

Profesor:

doc. dr Vladislav Miškovic

Asistent:

Petar Jakić

Student:

Nikola Živanović 2018203857

Beograd, 2021. Godine

Projektna dokumentacija

Sadržaj

1. Uvod

1.1 Cilj razvoja

U ovom projektu, cilj je napraviti web aplikaciju namenjenu korisnicima koji žele da prate potrošnju goriva svog automobila I sredstva koja se troše na dopunjavanje istog.

1.2 Obim sistema

Aplikacija nije velikog obima. U pitanju je troslojna aplikaija radjenja sa MVC arhitekturom.

Tri različite kategorije korisnika preko klijentske strane aplikacije (SPA) komunicira sa serverskom stranom aplikacije dalje slojem perzistencije podataka.

1.3 Prikaz proizvoda

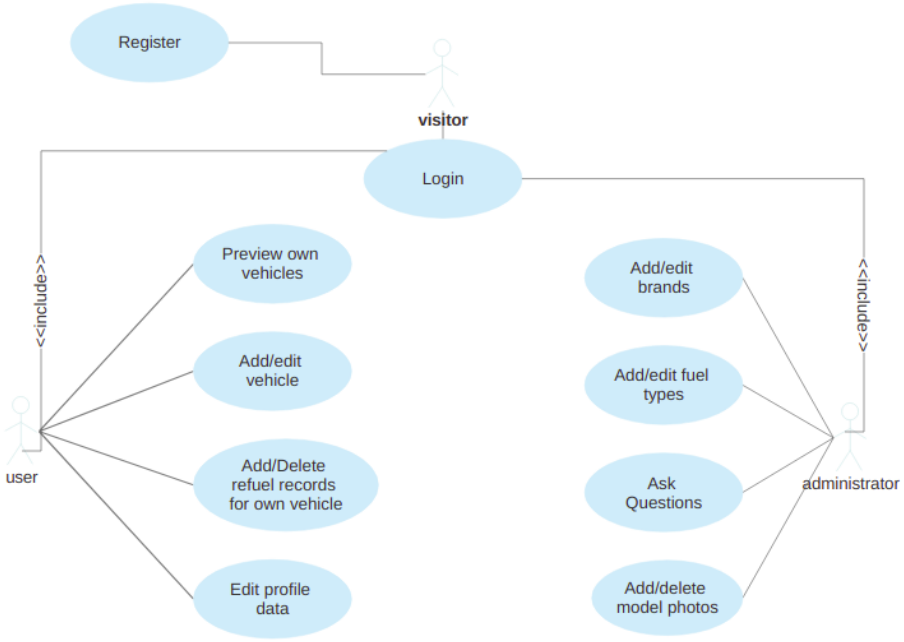
Slika proizvoda

1.3.1 Perspektiva proizvoda

Jednostavan proizvod koji će svako moći da koristi. Nema komercijalne perspektive

1.3.2 Funkcije proizvoda

Osnovna funkcija proizvoda je omogućavanje korisniku da vodi računa o potrošnji goriva. Kada korisnik u sistem doda svoje vozilo, svaki put kada dosipa gorivo, dovoljno je da upiše u aplikaciju I ostaje zabeleženo, kasnije ulazi u račun o potrošnji zajedno sa mesečnom statistikom.



2. Specifikacija zahteva

- Korisnici aplikacije mogu da se registruju unosom podataka o svom imenu, prezimenu, jedinstvenoj adresi elektronske pošte, jedinstvenombroju telefona i željenoj lozinci.
- Kada se prijave, mogu da dodaju u evidenciju aplikacije novo vozilo.
- Jedan korisnik može uneti više vozila.
- Svakom vozilu unosi proizvoljan naziv, proizvođača, model, godinu, boju i trenutnu pređenu kilometražu.
- Prema unetim podacima o proizvođaču, modelu, godini i boji, vozilu će biti pridružena slika konkretnog vozila ako je dostupna, a ako nije, biće pridružena generička slika vozila navedene boje.
- Korisnik može da doda svoju sliku vozila umesto podrazumevane.
- Podrazumevase da je prilikom unosa aplikacije stanje rezervoara bilo puno dovrha.
- Kada se prijavi na aplikaciju i odabere svoje vozilo za koju evidentira sipano gorivo, korisnik unosi podatak o tome koliko je goriva sipao i koja je zabeležena pređena kilometraža tog vozila.
- Aplikacija pita korisnika da li je prilikom konkretne evidencije unosa sipanoggoriva sipao do vrha rezervoara ili ne. Ako nije sipao do vrha, nemože da sračuna potrošnju, već prikuplja te podatke u bazu za budući proračun, kada prvi naredni put korisnik prilikom unosa evidencije naznači da je tada sipao do vrha.
- Korisnik u svakom trenutku može da vidi istoriju svojih sipanja goriva i pređenih kilometara na mesečnom nivou i tabelarno i grafički. Korisnik može da traži prikaz sumiranih potrošnji goriva i pređenog puta za odabrani vremenski period za svoje odabrano vozilo.

2.1 Ograničenja

- Korisnik mora imati pristupu webu preko web browseru ili sl.
- Korisnik mora da se registruje

2.2 Projektna/Tehnička ograničenja

- Aplikacija mora da bude realizovana na Node.js platformi korišćenjem Express biblioteke. Aplikacija mora da bude podeljena u dve nezavisne celine: back-end veb servis (API) i front-end (GUI aplikacija). Sav kôd aplikacije treba da bude organizovan u jednom Git spremištu u okviru korisničkog naloga za ovaj projekat, sa podelom kao u primeru zadatka sa vežbi.

- Baza podataka mora da bude relaciona i treba koristiti MySQL ili MariaDB sistem za upravljanje bazama podataka (RDBMS) i u spremištu back-end dela aplikacije mora da bude dostupan SQL dump strukture baze podataka, eventualno sa inicijalnim podacima, potrebnim za demonstraciju rada projekta.

- Back-end i front-end delovi projekta moraju da budi pisani na TypeScript jeziku, prevedeni TypeScript prevodiocem na adekvatan JavaScript. Back-end deo aplikacije, preveden na JavaScript iz izvornog TypeScript koda se pokreće kao Node.js aplikacija, a front-end deo se statički servira sa rute statičkih resursa back-end dela aplikacije i izvršava se na strani klijenta. Za postupak provere identiteta korisnika koji upućuje zahteve back-end delu aplikacije može da se koristi mehanizam sesija ili JWT (JSON Web Tokena), po slobodnom izboru.

- Sav generisani HTML kôd koji proizvodi front-end deo aplikacije mora da bude 100% validan, tj. da prođe proveru W3C Validatorom (dopuštena su upozorenja - Warning, ali ne i greške - Error). Grafički korisnički interfejs se generiše na strani klijenta (client side rendering), korišćenjem React biblioteke, dok podatke doprema asinhrono iz back-end dela aplikacije (iz API-ja). Nije neophodno baviti se izradom posebnog dizajna grafičkog interfejsa aplikacije, već je moguće koristiti CSS biblioteke kao što je Bootstrap CSS biblioteka. Front-end deo aplikacije treba da bude realizovan tako da se prilagođava različitim veličinama ekrana (responsive design).

- Potrebno je obezbediti proveru podataka koji se od korisnika iz front-end dela upućuju back-end delu aplikacije. Moguća su tri sloja zaštite i to: (1) JavaScript validacija vrednosti na front-end-u; (2) Provera korišćenjem adekvatnih testova ili regularnih izraza na strani servera u back-end-u (moguće je i korišćenjem izričitih šema - Schema za validaciju ili drugim pristupima) i (3) provera na nivou baze podataka korišćenjem okidača nad samim tabelama baze podataka.

- Neophodno je napisati prateću projektnu dokumentaciju o izradi aplikacije koja sadrži (1) model baze podataka sa detaljnim opisom svih tabela, njihovih polja i relacija; (2) dijagram baze podataka; (3) dijagram organizacije delova sistema, gde se vidi veza između baze, back-end, front-end i korisnika sa opisom smera kretanja informacija; (4) popis svih aktivnosti koje su podržane kroz aplikaciju za sve uloge korisnika aplikacije prikazane u obliku Use-Case dijagrama; kao i (5) sve ostale elemente dokumentacije predviđene uputstvom za izradu dokumentacije po ISO standardu.

- Izrada oba dela aplikacije (projekata) i promene kodova datoteka tih projekata moraju da bude praćene korišćenjem alata za verziranje koda Git, a kompletan kôd aplikacije bude dostupan na javnom Git spremištu, npr. na besplatnim GitHub ili Bitbucket servisima, jedno spremište za back-end projekat i jedno za front-end projekat. Ne može ceo projekat da bude otpremljen u samo nekoliko masovnih Git commit-a, već mora da bude pokazano da je projekat realizovan u kontinuitetu, da su korišćene grane (branching), da je bilo paralelnog rada u više grana koje su spojene (merging) sa ili bez konflikata (conflict resolution).

2.3 Spoljašnji interfejsi

slika front enda

2.4 Funkcije

Posetilac:

Registracija I autentifikacija korisnika

Servisne informacije na klijentskoj strani

Korisnik

Ograničenje pristupa I autorizacija bazirani na JWT tehnologiji

Dodavanje vozila

Izmena i/ili brisanje vozila

Unos pojedinačnih rekorda o svakom točenju goriva

Izmena informacija o korisniku

Administrator

Ograničenje pristupa I autorizacija bazirani na JWT tehnologiji

Dodavanje I ažuriranje brendova automobila I njihovih modela

Dodavanje novih tipova goriva

Ažuriranje fotografija određenih modela automobila

2.5 Zahtevane performanse

Aplikacija nije zahtevna po pitanju performansi I može se koristiti čak I na slabijim platformama, pod uslovom da mogu parsirati nove verzije Javascript-a

I klijentska I serverska strana su razvijene korišćenjem programskog jezika typescript koji se pre kompajliranja prevodi u Javascript, pa je tako jedini uslov mogućnost platfome da parsira I kompajlira Javascript kod.

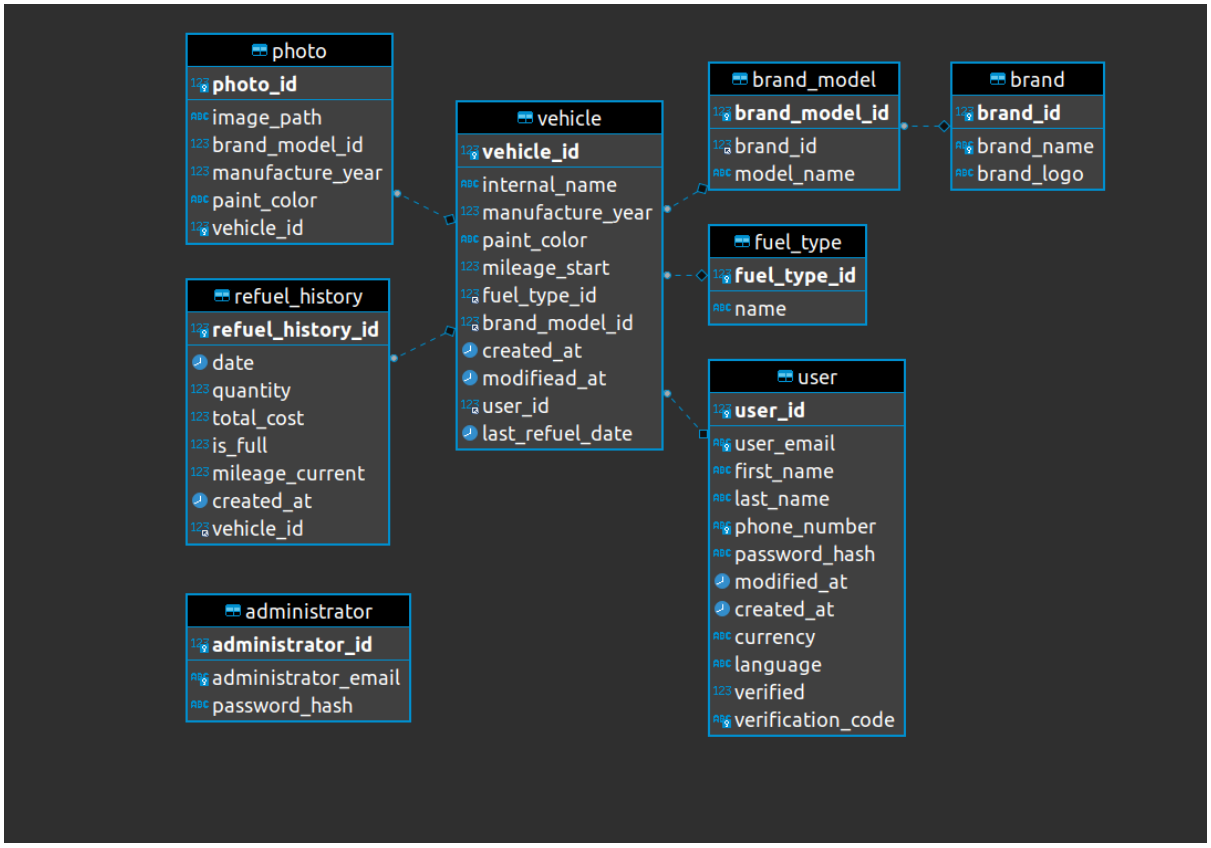
Tehnologija serverske strane je Node.js sa bibliotekom express

Tehnologija klijentske strane je Typescipt sa bibliotekom React.js

2.6 Zahtevi baze podataka

Baza podataka takodje nije zahtevna. Korišćenjem relacione baze podataka MySQL tehnologije postiže se relativno brz, a opet nezahtevan pristup podacima.

Tehnologija baze podataka po specifikaciji je MySQL



Slika: Dijagram relacija izmedju entiteta