Web aplikacija za konfiguriranje automobila

Grizelj, Zvonimir

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:536584

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-21**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Stručni studij računarstva

WEB PORTAL ZA KONFIGURACIJU AUTOMOBILA

Završni rad

Zvonimir Grizelj

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac Z2S: Zapisnik sa završnog ispita

Osijek, 18.07.2023.

Studentskoj službi

Zapisnik sa završnog ispita

Ime i prezime Pristupnika:	Zvonimir Grizelj
Studij, smjer:	Preddiplomski stručni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	AR 4726, 19.07.2019.
OIB Pristupnika:	19406095047
Mentor:	prof. dr. sc. Krešimir Nenadić
Sumentor:	,
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	doc. dr. sc. Krešimir Romić
Član Povjerenstva 1:	prof. dr. sc. Krešimir Nenadić
Član Povjerenstva 2:	Robert Šojo, mag. ing. comp.
Naslov završnog rada:	Web aplikacija za konfiguriranje automobila
Kratko obrazloženje prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina

Završni ispit održan je na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek dana 18.07.2023. u 9:00 sati

Pitanja članova Povjerenstva za završni ispit:	Nenadić: 1. Koliko ja vaša web aplikacija prilagodljiva različitim zahtjevima prodavača? 2. Koji portal vam je bio glavni izvor funkcionalnosti? Šojo: 1. Postoje li nekakve zaštite pri kreiranju novog korisnika? 2. Što ako korisnik zaboravi zaporku? Romić: 1. Kako bi se rješio problem manjih promjena u konfiguraciji bez povećavanja broja slika - asseta? 2. Na koji način bi se mogla dijeliti konfiguracija s drugim osobama?
Mišljenje mentora o pismenom dijelu rada i Povjerenstva o tijeku završnog ispita:	Pristupnik je samostalno obradio zadanu temu i izradi web aplikaciju pri čemu se koristio znanjima i vještinama stečenim tijekom studija kao i onima stečenim samostalno. Pristupnik je jasno izložio sažetak svog rada i odgovorio na sva postavljena pitanja.

Ocjena pismenog dijela ispita (završni rad):	Izvrstan (5)
Ocjena usmenog dijela ispita (završni ispit):	Izvrstan (5)
Ukupna ocjena na završnom ispitu:	**Izvrstan (5)**

Predsjednik Povjerenstva:	
Član 1:	
Član 2:	
Zapisničar:	
Pristupnik:	

Uspješno položenim završnim ispitom Pristupnik stječe stručni naziv:

Stručni prvostupnik (baccalaureus) inženjer računarstva

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 01.08.2023.

Ime i prezime studenta:	Zvonimir Grizelj
Studij:	Preddiplomski stručni studij Računarstvo
Mat. br. studenta, godina upisa:	AR 4726, 19.07.2019.
Turnitin podudaranje [%]:	15

Ovom izjavom izjavljujem da je rad pod nazivom: **Web aplikacija za konfiguriranje automobila**

izrađen pod vodstvom mentora prof. dr. sc. Krešimir Nenadić

i sumentora ,

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Grizelj

SADRŽAJ

1. UVOD 1

1.1 Zadatak završnog rada 1

1.2 Idejno rješenje 2

2. POSTOJEĆI PORTALI 3

2.1 Ferarri configurator 3

2.2 Lamborghini configurator 3

2.3 Porsche configurator 4

2.4 Lexus configurator 5

2.5 Bmw configurator 6

3. TEHNOLOGIJE 7

3.1 Visual Studio Code razvojno okruženje 7

3.2 JavaScript 7

3.3 Yarn 8

3.4 React 8

3.5 Figma 9

3.6 HTML 9

3.7 CSS 9

3.7.1 Emotion 10

3.8 Cloud Firestore 11

4. FUNKCIONALNOSTI WEB APLIKACIJE 12

4.1. Baza podataka 12

4.2. Prijava i registracija 14

4.3. Naslovna stranica 15

4.4. Izbor vozila 19

4.5. Eksterijer vozila 19

4.5.1 Boja vozila 20

4.5.2 Stil vozila 21

4.6. Interijer vozila 21

4.6.1 Boja unutrašnjosti vozila 22

4.7 Sažetak 23

5. ZAKLJUČAK 24

LITERATURA 25

SAŽETAK 26

ABSTRACT 27

ŽIVOTOPIS 28

1. UVOD

U ovom završnom radu objašnjene su i opisane funkcionalnosti te metode izrade web aplikacije *konfigurator automobila*. Pristup samoj web aplikaciji je putem prijave na stranicu, a uz nju stranica za registraciju novih korisnika. Glavne funkcionalnosti koja pruža ova web aplikacija jest da korisnici mogu izraditi svoje konfiguracije vozila, te ih zatim spremiti tako da kasnije imaju pristup njima u bilo kojem trenutku. Svaki korisnik ima mogućnost brisanja i uređivanja spremljenih konfiguracija.

Najprije se opisuje idejno rješenje samog zadatka te nakon toga se opisuju trenutni postojeći portali, raznih proizvođača, koji imaju mogućnost konfiguracije vozila. Nakon toga su opisane i navedene tehnologije koje su korištene prilikom razvoja ove web aplikacije. Nakon opisa tehnologija, opisan je i rad same web aplikacije te njene funkcionalnosti i mogućnosti.

1.1 Zadatak završnog rada

Kratko opisati postupak konfiguracije automobila za potencijalnog kupca. Dati pregled sličnih aplikacija i međusobno ih usporediti. Zatim je potrebno definirati zahtjeve za web aplikaciju pomoću koje će potencijalni kupci moći konfigurirati izgled i opremu automobila. Za potrebe web aplikacije potrebno je modelirati i izraditi bazu podataka te opisati taj postupak. Potrebno je opisati postupak izrade web aplikacije kao i njene funkcionalnosti nakon izrade.

1.2 Idejno rješenje

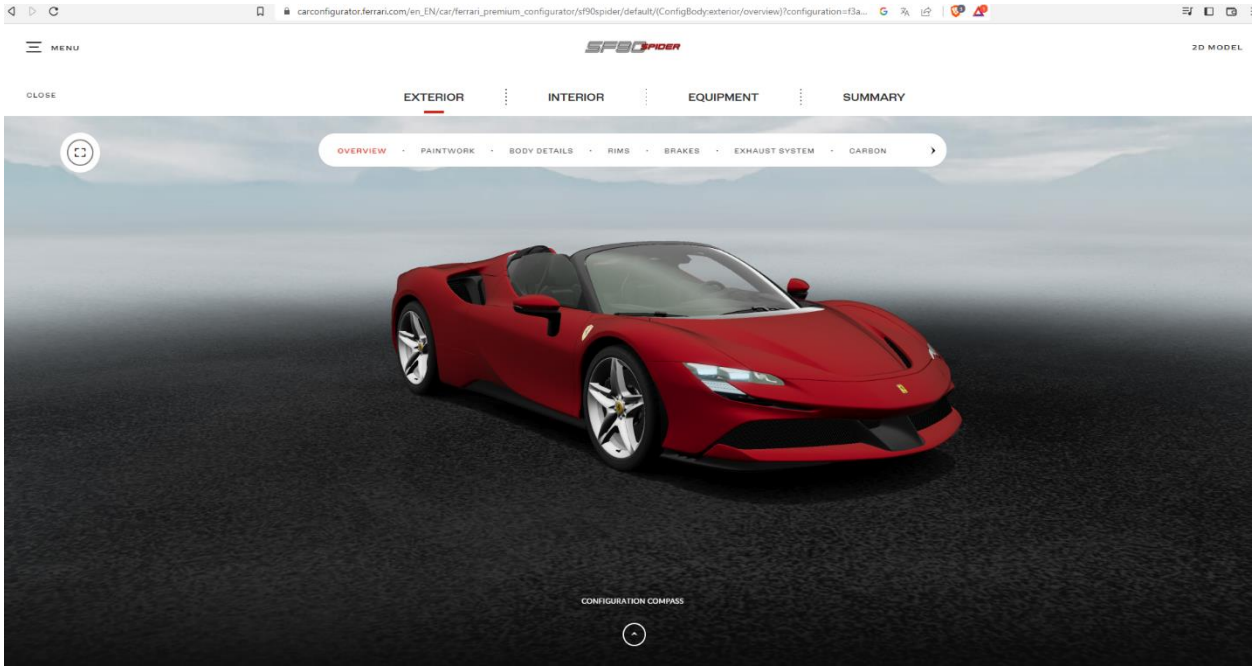
Ideja samog sustava je olakšavanje organizacije korisnikovih konfiguracija. Korisnik ima mogućnost napraviti proizvoljan broj konfiguracija te ima mogućnost uređivanja istih. Nadalje, ima mogućnost na početku izabrati od trenutno 3 ponuđenih vozila od kojih može konfigurirati svoje vozilo. Broj ponuđenih vozila se uvijek može povisiti s dodavanjem novih vozila i njihovih fotografija. Nakon izbora samog vozila, korisnik dolazi do sljedećeg izbornika gdje bira eksterijer vozila. Ima mogućnost izabrati boju vozila te njegov stil. Nakon vanjštine, korisnik dobiva mogućnost biranja interijera vozila koje mu najviše odgovara. Na kraju samog izbora eksterijera i interijera, korisnik dolazi na stranicu u kojoj ima mogućnost ukratko pregledati svoje izbore i odabrati je li zadovoljan sa svojim izborom. Nakon toga prihvaća odabranu konfiguraciju te mu se konfiguracija sprema na njegov profil, gdje uvijek ima mogućnost brisanja ili uređivanja iste konfiguracije.

2. POSTOJEĆI PORTALI

Unutar ovog poglavlja navodi se nekoliko već postojećih portala koji pružaju mogućnost korisniku konfiguraciju vozila.

2.1 Ferarri configurator

Web aplikacija *Ferarri konfigurator* pruža, od svih navedenih, najkvalitetnije korisničko iskustvo prilikom konfiguriranja vozila. Dodatna opcija koju pruža navedena web aplikacija jest da korisnik osim što svoje vozilo vidi u 2D formatu prilikom konfiguracije može vozilo prikazati u 3D formatu kako bi imao što realističnije iskustvo prilikom konfiguracije vozila. Postoji mogućnost okretanja kamere za 360° oko vozila. Na slici 2.1. vidi se izgled stranice gdje vidimo izabrani model vozila u 3D formatu, te iznad vozila specifikacije koje korisnik može mijenjati.

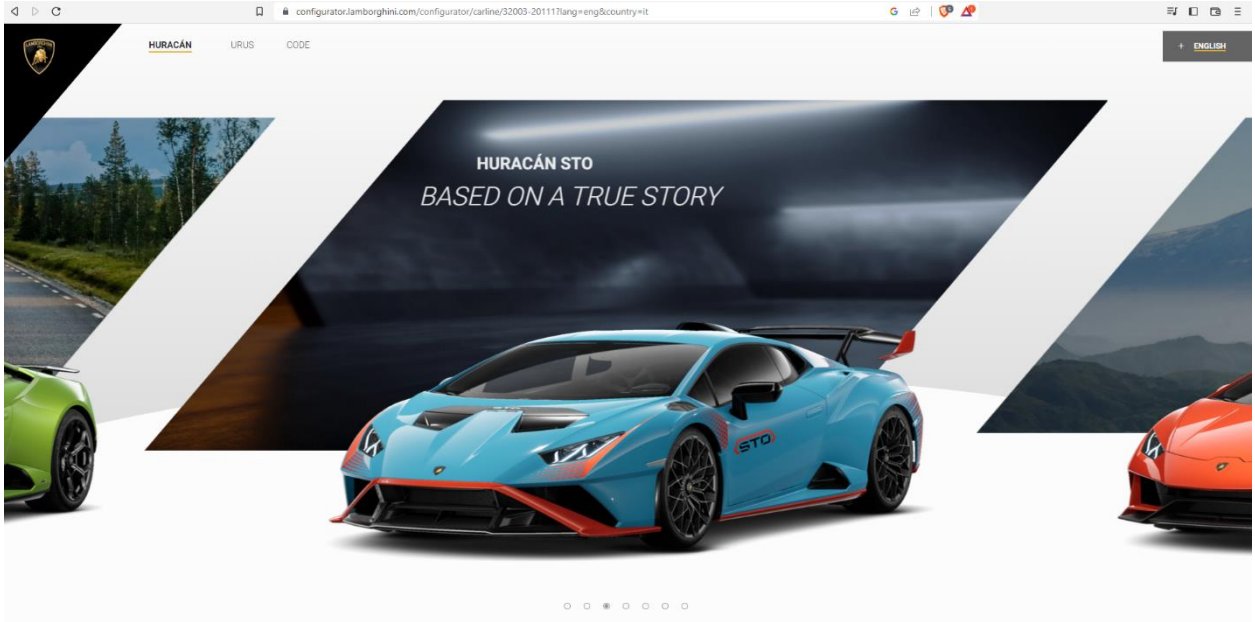


Slika 2.1 *Izgled stranice https://carconfigurator.ferrari.com/en_EN*

2.2 Lamborghini configurator

Web aplikacija *Lamborghini konfigurator* prilikom ulaska na stranicu daje mogućnost odabira jednog od 7 postojećih modela automobila koji se koristi za daljnju konfiguraciju. Moguće je promijeniti kut prikaza automobila tijekom konfiguracije, ali ne i samostalno okretanje kamere. Glavne specifikacije vozila koje se mogu promijeniti jesu boja vozila, boju te oblik felgi te dodavati detalje na vozilo. Što se tiče konfiguriranja unutarnjeg dijela vozila moguće je izmijeniti

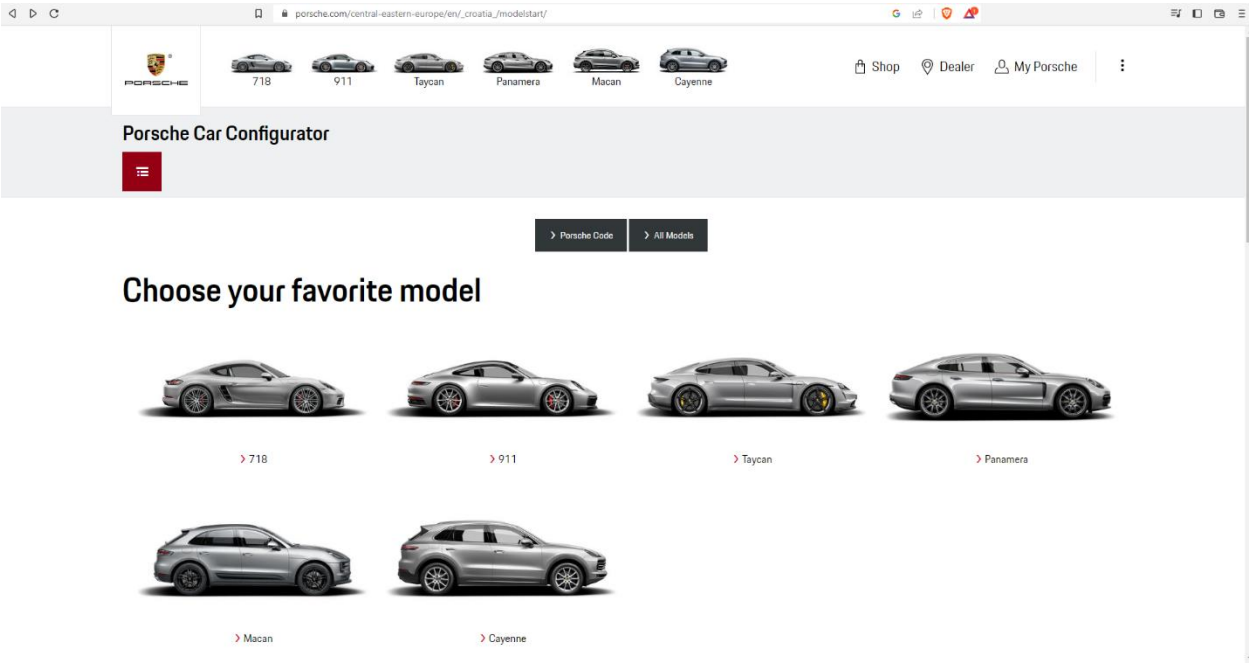
izgled volana i sjedala te dodavanje detalja i pomoćne opreme. Na slici 2.2. je prikazan izgled web stranice na kojoj se vidi početni izbornik na kojem su prikazani ponuđeni modeli vozila od kojih je moguće započeti konfiguraciju.



Slika 2.2 *Izgled stranice https://configurator.lamborghini.com/configurator/carline/32003-20111?lang=ita&country=it*

2.3 Porsche configurator

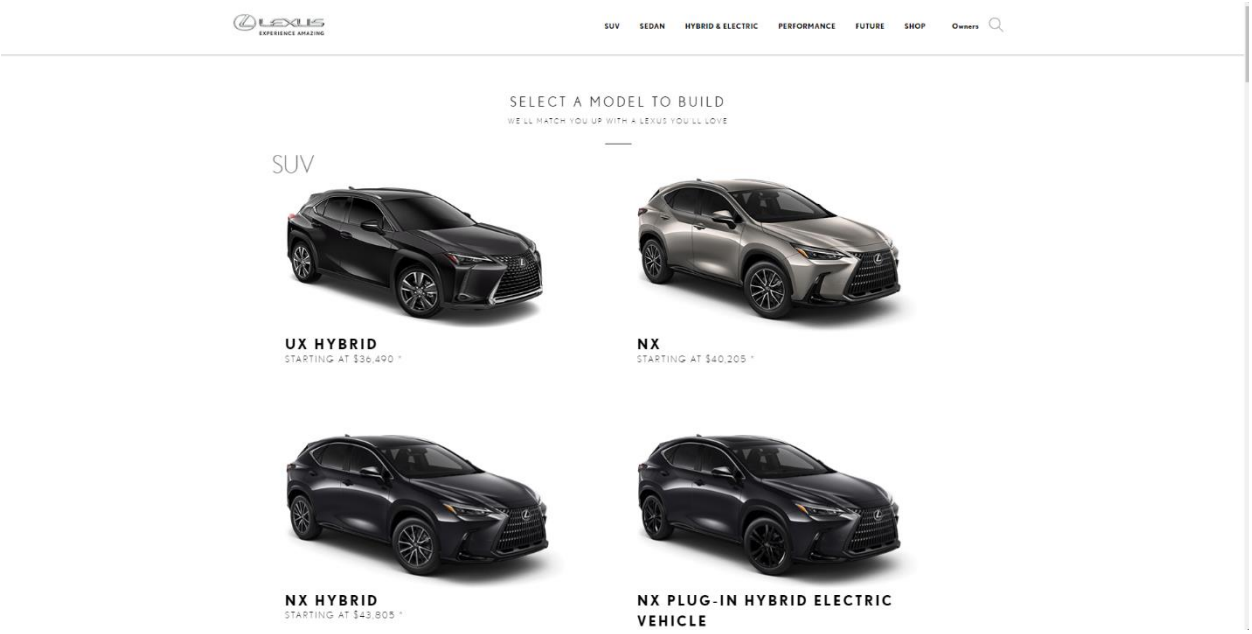
Porsche configurator web aplikacija za konfiguraciju automobila ima jednostavniji pristup izboru početnog modela vozila unutar kojeg se vide svi ponuđeni modeli vozila odjednom te tako čini stranicu preglednijom. Prilikom odabira modela stranica filtrira sve odabrane model te se nakon toga bira automobil za daljnju konfiguraciju. Nakon izbora automobila za konfiguraciju analogno kao i na prethodnim stranicama ima mogućnosti mijenjati interijer i eksterijer. Dodatno, korisnik ima priliku izabrati željeni tip goriva, broj konjskih snaga te želi li da vozilo ima pogon na sva 4 kotača ili samo na stražnja 2. Na slici 2.3. prikazan je početni izbornik opisane stranice.



Slika 2.3 *Izgled stranice https://www.porsche.com/central-eastern-europe/en/_croatia_/modelstart/*

2.4 Lexus configurator

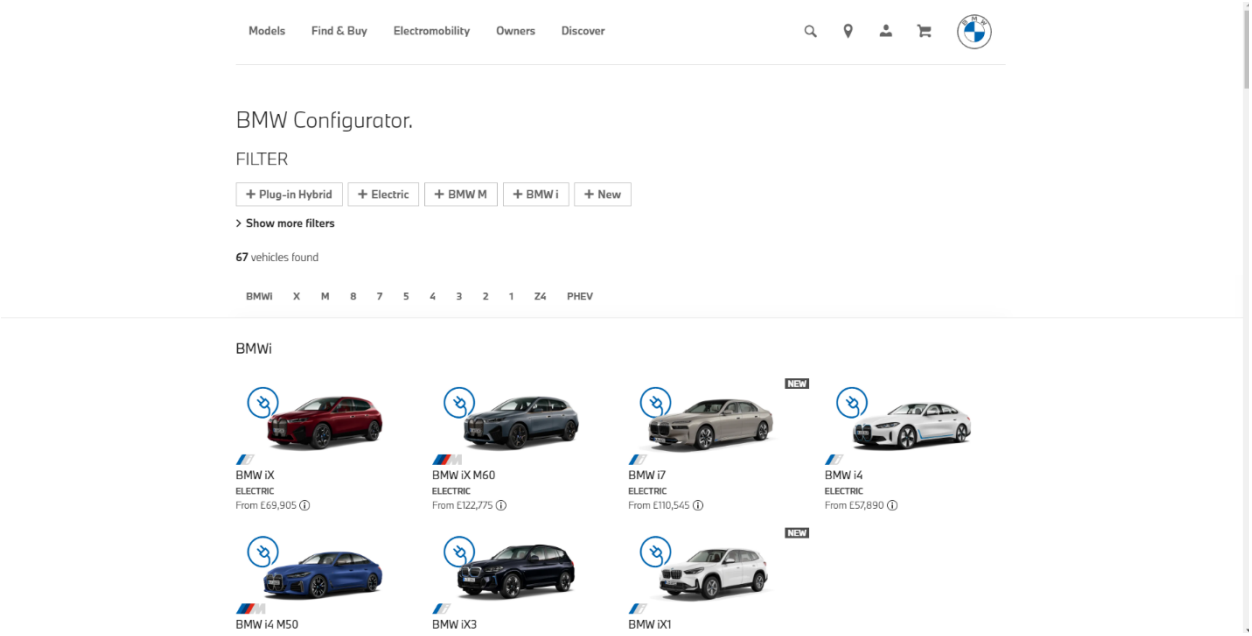
Na slici 2.4. vidimo izgled web aplikacije za konfiguraciju lexus automobila. Web aplikacija je napravljena na taj način da su sva vozila ispisana odmah na početku, a prilikom izbora vozila, vodi korisnika na daljnji izbor konfiguracije njegovog vozila. Prilikom konfiguracije vozila, u usporedbi s prethodno navedenim konfiguratorima, opcije za promjenu vozila su znatno manje.



Slika 2.4 *Izgled stranice https://www.lexus.com/build-your-lexus/#!/series*

2.5 Bmw configurator

Na slici 2.5. se vidi izgled početne stranice web aplikacije *BMW configurator*, koja služi za konfiguraciju vozila. Odmah pri vrhu stranice vidimo mogućnost filtriranja modela početnog automobila kako bi se korisniku olakšao i omogućio što brži dolazak do željenog vozila. Nakon izbora filtera, na dnu stranice se izlistavaju filtrirana vozila od kojih korisnik bira jedno kao svoje početno vozilo. Nakon izbora početnog vozila dolazi izbor interijera i eksterijera unutar kojih je moguće promijeniti boju automobila, oblik i boju felgi, boju interijera i slično. Također kao i kod prethodno navedene web aplikacije *Ferrari configurator* moguće je oko automobila te unutar istog okretati kameru za 360°.



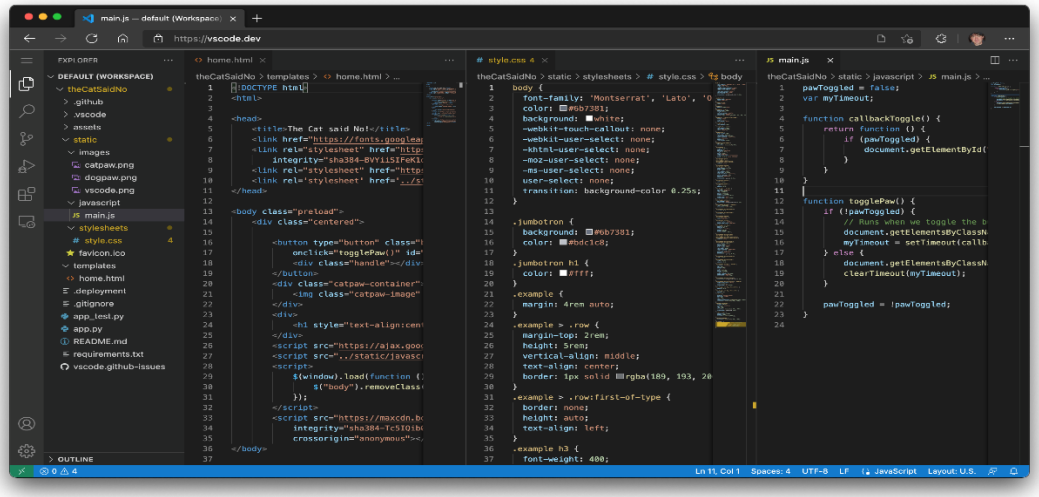
Slika 2.5 *Izgled stranice https://www.bmw.co.uk/en/configurator.html*

3. TEHNOLOGIJE

U ovom poglavlju su objašnjenje tehnologije koje su korištene prilikom izrade web aplikacije.

3.1 Visual Studio Code razvojno okruženje

Visual Studio Code je uređivač izvornog koda i dostupan je za Windows, macOs i Linux. On u sebi ima ugrađenu podršku za JavaScript, TypeScript i Node.js i bogat ekosustav proširenja za druge programske jezike i vremena izvođenja (npr. C++, C#, Java, Python, PHP (Hypertext Preprocessor), Go, .NET (Domain NETwork)). Na slici 3.1. je prikazan izgled glavnog prozora Visual Studio Code-a. S lijeve strane vidimo *File Explorer*, koji nam daje pregled svih datoteka unutar odabrane mape. Na sredini je prikazan kod, koji je moguće ovoriti u više okvira, te se tako olakšava pregled samog koda. [1]



Slika 3.1 *Izgled Visual Studio Code okruženja*

3.2 JavaScript

JavaScript ili JS je poznatiji kao skriptni jezik za izradu web aplikacija, no može se koristiti i u mnogim okruženjima koja nisu preglednici. JavaScript kao što je rečeno je skriptni jezik koji sadržava veći broj paradigmi koje se temelje na prototipu koji je dinamičan i podržava objektno orijentirane, imperativne i funkcionalne stilove programiranja. Najčešća upotreba JavaScript programskog jezika jest izvođenje na strani klijenta web aplikacija, što se može koristiti za

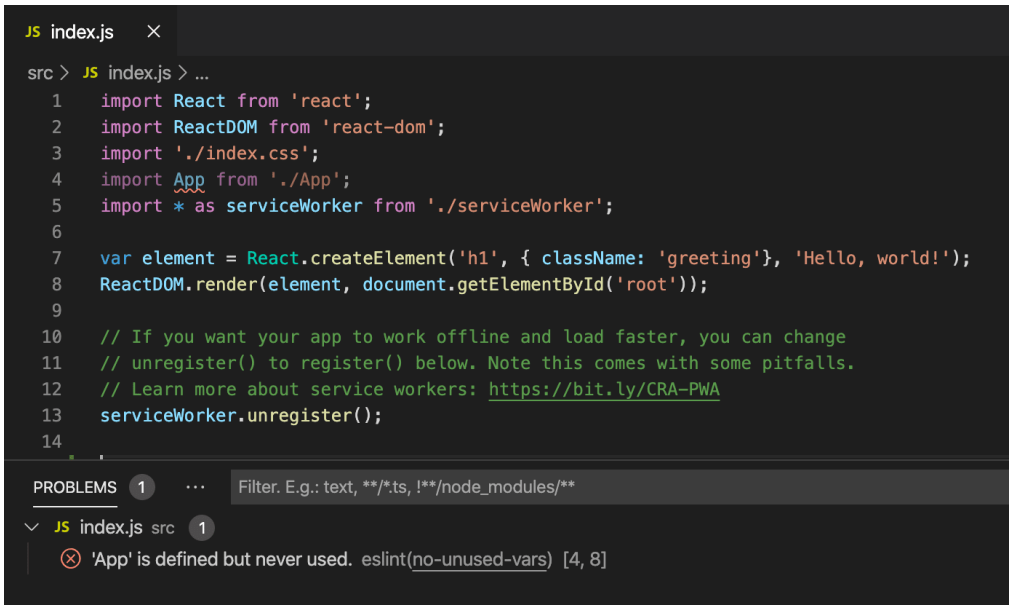
programiranje ponašanja web aplikacije u slučaju pojave događaja. Pomoću njega također dodajemo i animacije samim elementima unutar stranice. [2]

3.3 Yarn

Yarn je alat za upravljanje ovisnostima(engl. d*ependency management*). Yarn omogućuje sigurno, brzo i pouzdano korištenje i dijeljenje koda s drugim programerima iz cijelog svijeta što ga čini jednim od popularnijih upravitelja u ovom području. Omogućuje korištenje rješenja drugih programera za različite probleme, što zapravo olakšava razvoj softvera. Način na koji dobivamo dijeljeni kod jest kroz pakete, a paket sadrži sav kod koji se dijeli, kao i datoteku package.json (manifest) koja opisuje paket. [3]

3.4 React

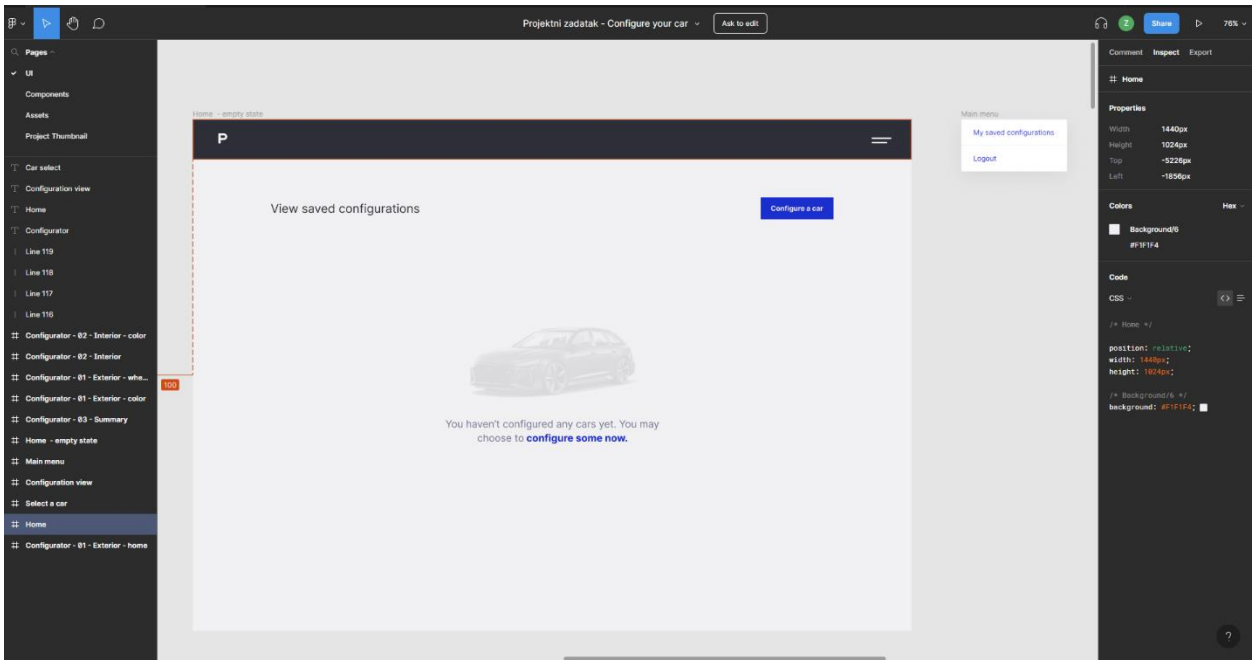
React je besplatna JavaScript biblioteka otvorenog koda koja služi developerima za lakšu izgradnju korisničkih sučelja. Osim korisničkog sučelja, olakšano je i kontroliranje korisnikovih akcija i reagiranje na iste. Održava ga Meta (Facebook) i zajednica pojedinačnih programera i tvrtki. React se može koristiti kao baza u razvoju jednostraničkih, mobilnih ili poslužiteljskih aplikacija s okvirima poput Next.js. On se bavi samo upravljanjem stanja i prikazivanjem istog u *document project modelu (DOM-u)*, tako da stvaranje React aplikacija obično zahtjeva korištenje dodatnih knjižnica za usmjeravanje, kao i određene funkcionalnosti na strani klijenta.[4] Na slici 3.2. je prikazan primjer React koda koji, nakon pokretanja, na stranici ispisuje rečenicu „Hello, world!“.



Slika 3.2 *Primjer React koda*

3.5 Figma

Figma je uređivač vektorske grafike i alat za izradu prototipa koji se prvenstveno temelji na webu, s dodatnim izvanmrežnim značajkama koje omogućuju desktop aplikacije za macOS i Windows. [5] Na slici 3.3. je prikazan izgled web aplikacije koji je napravljen unutar Figme radi lakšeg prikazivanja ideje kako treba izgledati.



Slika 3.3 *Izgled web aplikacije u Figmi*

3.6 HTML

HTML (HyperText Markup Language) je standardni jezik za označavanje dokumenata dizajniranih za prikaz u web pregledniku. Pomažu mu tehnologije kao što su Cascading Style Sheets (CSS) i skriptni jezici kao što je JavaScript. Web preglednici primaju HTML dokumente s web poslužitelja ili iz lokalne pograne i pretvaraju dokumente u multimedijske web stranice. HTML opisuje strukturu web stranice semantički i izvorno uključuje znakove za izgled dokumenta. [6]

3.7 CSS

Cascading Style Sheets je jezik stilskih tablica koji se koristi za opisivanje prezentacije dokumenta napisanog u označenom jeziku kao što je HTML ili XML. On je temeljni dio *World Wide Weba*,

uz HTML i JavaScript. Dizajniran je na način da omogućuje odvajanje prezentacije i sadržaja, uključujući izgled, boje i fontove. Samo to razdvajanje može poboljšati dostupnost sadržaja;

pružiti veću fleksibilnosti i kontrolu u specifikaciji karakteristika prezentacije; omogućiti višestrukim web stranicama dijeljenje formatiranja navođenjem relevantnog CSS-a u zasebnoj .css datotecim što smanjuje složenost i ponavljanje u strukturnom sadržaju. [7] Na slici 3.4. je prikazan primjer CSS koda napisanog unutar Visual Studio Code-a.

```
# style.css ×
react-academy-html-css > # style.css > ...
}
    You, 14 seconds ago • Uncommitted changes
body{
  margin: 0 auto;
}
.wrapper{
  max-width: 800px;
  margin: 0 auto;
  width: 100%;
  padding: 0 40px;
}

header{
  background-color: rgb(0, 47, 60);
  display: flex;
  flex-direction: row;
  justify-content: space-between;
}

.header__logo{
  border-top: 10px;
  border-color: #008a5c;
  padding-top: 20px;
  text-transform: uppercase;
  color: #008a5c;
  text-decoration: none;
  justify-content: left;
}

.header__hero{
  text-align: center;
  background-color: rgb(0, 47, 60);
  color: rgba(255, 255, 255, 0.699);
  margin-top: 0px;
  justify-content: center;
}
h1{
  color: #008a5c;
}
```

Slika 3.4 *Primjer CSS koda*

3.7.1 Emotion

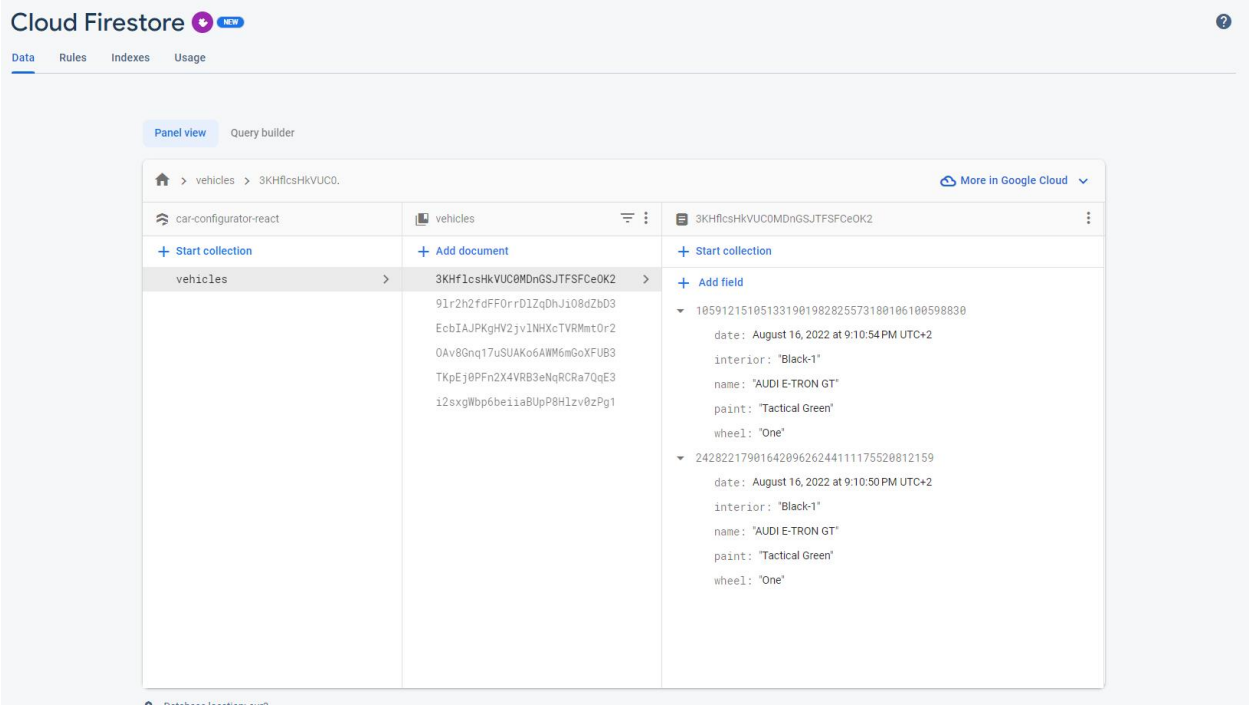
Emotion je knjižnica dizajnirana za pisanje css stilova pomoću JavaScript-a. Pruža moćnu i predvidljivu kompoziciju stila uz izvrsno iskustvo programera sa značajkama kao što su izvorne karte, oznake i uslužni programi za testiranje. Podržani su i stilovi nizova i objekata. [8] Na slici 3.5. je prikazano dodavanje Emotion knjižnice u projekt preko terminala.

```
Gric@DESKTOP-OSSQJKG MINGW64 /d/Programiranje/React/test-projekt (master)
$ yarn add @emotion/react
yarn add v1.22.19
warning package-lock.json found. Your project contains lock files generated by tools other than Yarn. It is advised not to mix package managers in order to avoid resolution inconsiste
ncies caused by unsynchronized lock files. To clear this warning, remove package-lock.json.
[1/4] Resolving packages...
[2/4] Fetching packages...
[3/4] Linking dependencies...
warning "react-scripts > eslint-config-react-app > eslint-plugin-flowtype@8.0.3" has unmet peer dependency "@babel/plugin-syntax-flow@^7.14.5".
warning "react-scripts > eslint-config-react-app > eslint-plugin-flowtype@8.0.3" has unmet peer dependency "@babel/plugin-transform-react-jsx@^7.14.9".
warning "react-scripts > react-dev-utils > fork-ts-checker-webpack-plugin@6.5.2" has unmet peer dependency "typescript@>= 2.7".
warning "react-scripts > eslint-config-react-app > @typescript-eslint/eslint-plugin > tsutils@3.21.0" has unmet peer dependency "typescript@>=2.8.0 || >= 3.2.0-dev || >= 3.3.0-dev ||
>= 3.4.0-dev || >= 3.5.0-dev || >= 3.6.0-dev || >= 3.6.0-beta || >= 3.7.0-dev || >= 3.7.0-beta".
warning "@emotion/react > @emotion/babel-plugin > @babel/plugin-syntax-jsx@7.18.6" has unmet peer dependency "@babel/core@^7.0.0-0".
warning "@emotion/react > @emotion/babel-plugin@11.9.2" has unmet peer dependency "@babel/core@^7.0.0".
[4/4] Building fresh packages...
success Saved lockfile.
success Saved 11 new dependencies.
info Direct dependencies
└─ @emotion/react@11.9.3
info All dependencies
├─ @emotion/babel-plugin@11.9.2
├─ @emotion/cache@11.9.3
├─ @emotion/memoize@0.7.5
├─ @emotion/react@11.9.3
├─ @emotion/serialize@1.0.4
├─ @emotion/sheet@1.1.1
├─ @emotion/unitless@0.7.5
├─ babel-plugin-macros@2.8.0
├─ csstype@3.1.0
├─ find-root@1.1.0
└─ hoist-non-react-statics@3.3.2
Done in 9.27s.
```

Slika 3.5 *Dodavanje Emotion knjižnice u projekt*

3.8 Cloud Firestore

Firestore je baza podataka koja pohranjuje podatke u obliku dokumenta, a ne u relacijske baze kao što to radi SQL programski jezik te spada u klasu *NoSql document databases*. Baza podataka izgrađena je za automatsko skaliranje (ovisno o potrebi), ima visoke performanse te se lako razvija. Firestore baza podataka funkcionira bez servera, te se njome može lako manipulirati. [9] Na slici 3.6. je prikazan izgled same baze koja se koristi za izrađenu web aplikaciju, gdje pod *collections* imamo ID-eve koji predstavljaju jedinstvene oznake korisnika te svaki korisnik ima svoju listu vozila koja se vežu na njega.



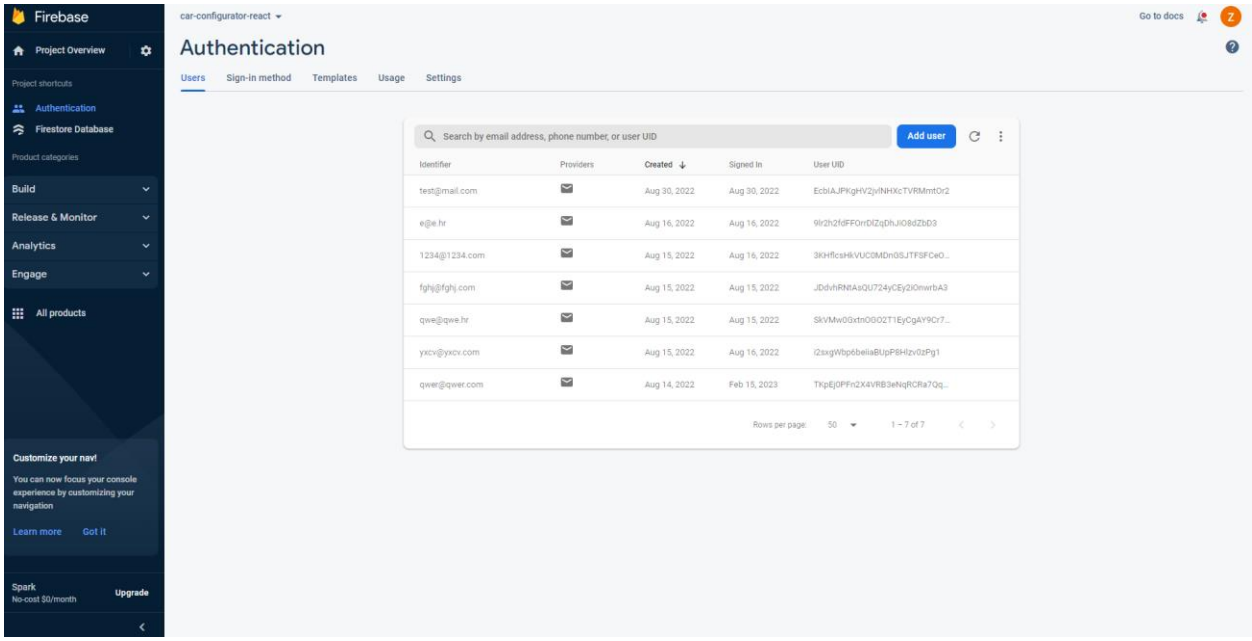
Slika 3.6 *Izgled Firestore baze*

4. FUNKCIONALNOSTI WEB APLIKACIJE

U ovom poglavlju je opisana sama funkcionalnost aplikacije, te na koji način određene funkcionalnosti klijentske i poslužiteljske strane aplikacije rade.

4.1. Baza podataka

Baza podataka je napravljena na servisu Firebase te se koristio Cloud Firestore. Neke od raznih mogućnosti koje nam pruža upravljačka ploča Firebase-a su na primjer: dodavanje novih korisničkih profila, mogućnost prijave preko Google računa, itd. Na slici 4.1. vidimo trenutno aktivne profile i njihove podatke, poput identifikacijske oznake (ID). Na slici 4.2. je prikazan primjer koda koji služi za unos novog vozila u bazu podataka koji se veže na određenog korisnika. Funkcija *handleClick* radi na način, da provjerom *if* grananja, radi se provjera je li postoji već vozilo s tim ID-em, ako da, radi se samo o uređivanju postojećeg vozila, te treba promijeniti vrijednosti i spremiti ih. Ako nema vozila s trenutnim ID-em, novo vozilo se onda sprema u bazu podataka.



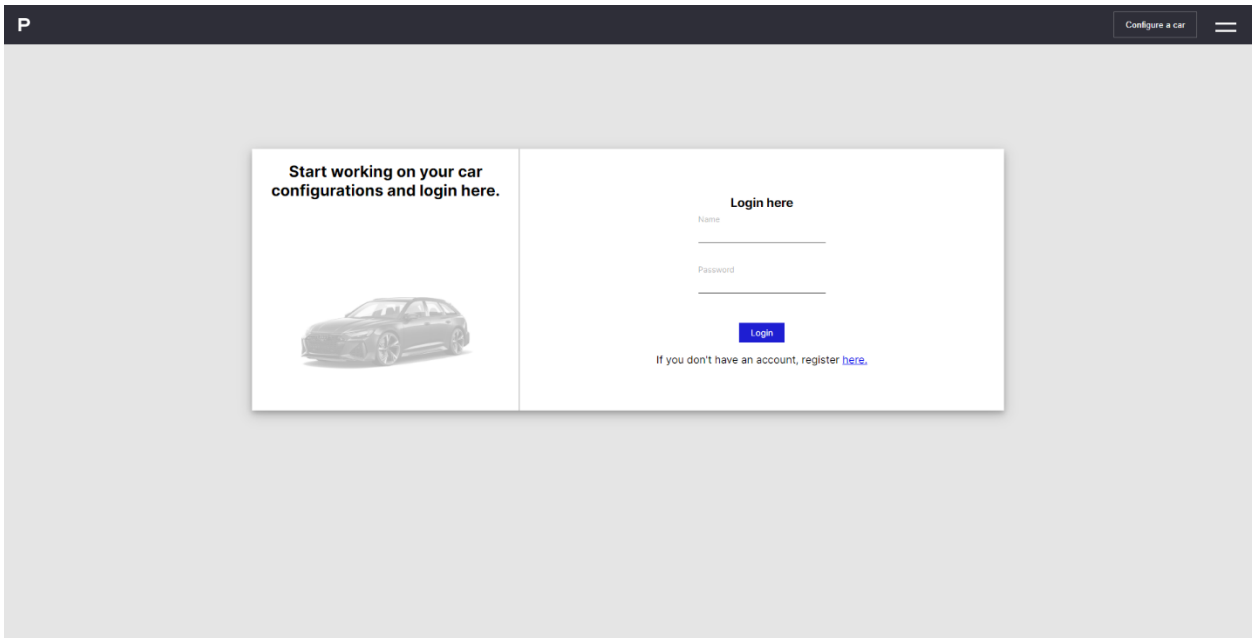
Slika 4.1 *Izgled Firebase upravljačke ploče*

```
function setListener() {
  const unsubscribe = onSnapshot(
    doc(db, "vehicles", user),
    (querySnapshot) => {
      const vehicles = querySnapshot.data() as Vehicle[];
      setCars(vehicles);
    }
  );
}

async function handleClick() {
  if (carEditToId && carEditToId !== "") {
    const docRef = await updateDoc(doc(db, "vehicles", user), {
      [carEditToId]: {
        name: car,
        paint,
        wheel,
        interior,
        date: new Date(),
      },
    });
    setCar("");
    setPaint("");
    setWheel("");
    setInterior("");
    navigate("/home");
    return;
  }
  const tempId = crypto.getRandomValues(new Uint8Array(16)).join("");
  const tempData = {
    ...cars,
    [tempId]: {
      name: car,
      paint: paint,
      wheel: wheel,
      interior: interior,
      date: getDate(),
    },
  };
  const docRef = await setDoc(doc(db, "vehicles", user), tempData);
  navigate("/home");
}
```

Slika 4.2 *Primjer koda za unos u bazu*

4.2. Prijava i registracija

Kako je aplikacija namijenjena za veći broj korisnika, te korisnike moramo razlikovati unutar svoje baze. Svaki korisnik ima svoj jedinstveni identifikacijski broj po kojem se spremaju nova vozila. Ako se radi o novom korisniku tada se prvo koristi registracije, a ako se radi o postojećem korisniku on koristi prijavu na web stranicu. Na slici 4.3 je prikazan izgleda stranice gdje se vidi stranica za prijavu korisnika. Razlika stranica za prijavu i registraciju jest da prilikom registracije u bazi podataka se stvara novi korisnik te se generira njegov ID, a ako je riječ o prijavi provjerava se ispravnost unesenih podataka. Nakon registracije, odnosno ako su podaci pravilno uneseni tijekom prijave korisnik dolazi na naslovnu stranicu web aplikacije.



Slika 4.3 *Izgled stranice za prijavu*

Na slici 4.4. je prikazan dio koda koji služi za prijavu na web aplikaciju. Vrši se provjera odgovara li upisani email i lozinka od strane korisnika podacima zapisanima u bazi podataka. Ako odgovara, korisnik se uspješno prijavljuje na web aplikaciju i pozicionira ga se na naslovnu stranicu.

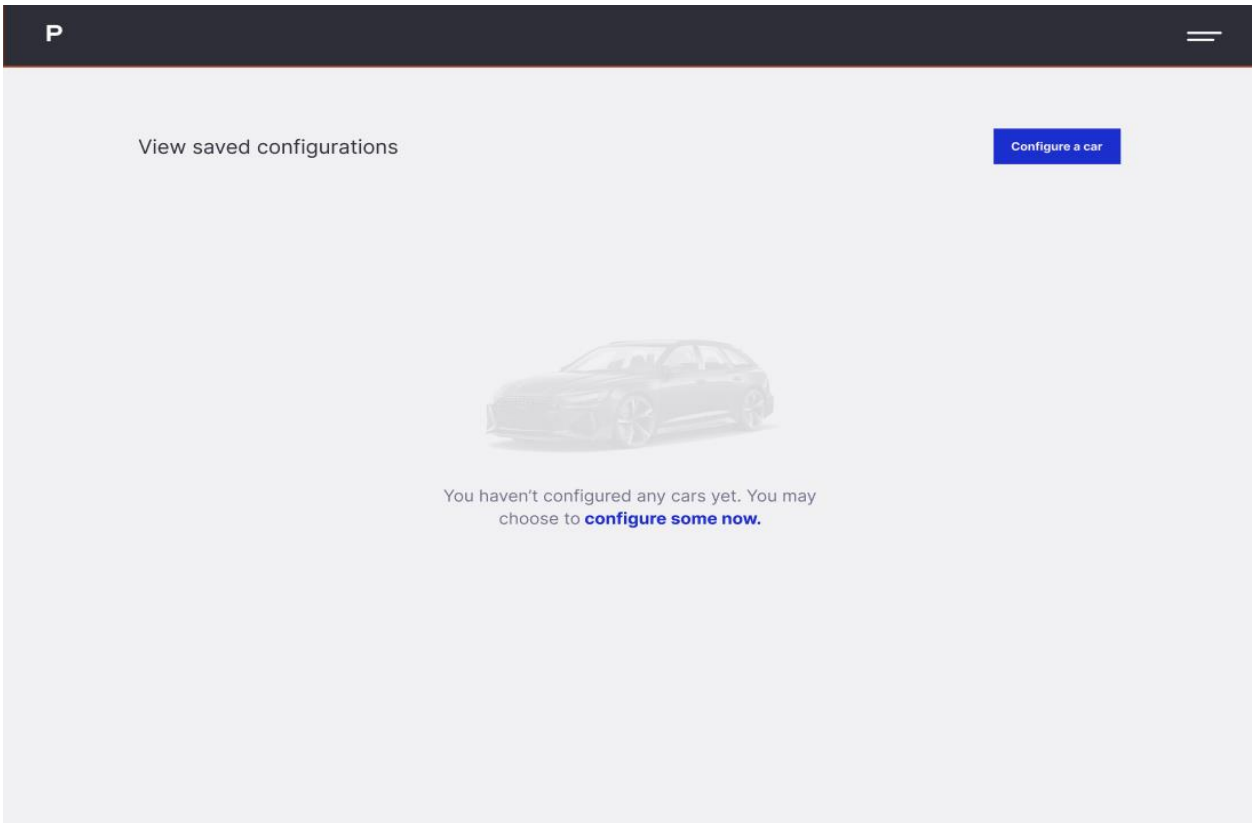
```
function handleLogin(e: React.FormEvent) {
  e.preventDefault();

  if (!email || !password) {
    console.log("error");
    return;
  }

  signInWithEmailAndPassword(auth, email, password)
    .then((userCredential) => {
      // Signed in
      navigate("/home");
      const user = userCredential.user;
      setUser(user.uid);
    })
    .catch((error) => {
      const errorCode = error.code;
      const errorMessage = error.message;
      console.log("Error login");
    });
}
```

Slika 4.4 *Primjer koda za prijavu*

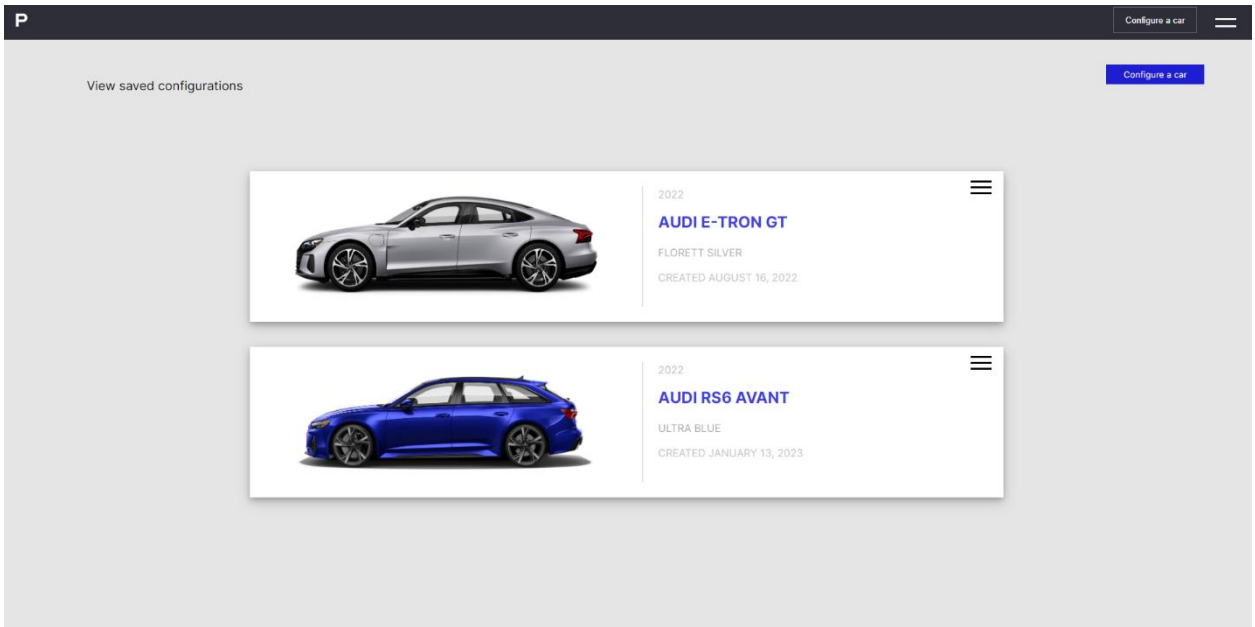
4.3. Naslovna stranica

Naslovna, odnosno početna stranica je prvo što je korisniku prikazano nakon prijave, odnosno registracije. Ako korisnik nema spremljene konfiguracije, kao što je prikazano na slici 4.5 na naslovnoj stranici se tada korisnika obavijesti da nema postojećih konfiguracija te mu se daje mogućnost izrade iste pritiskom tipke *Configure a car*.

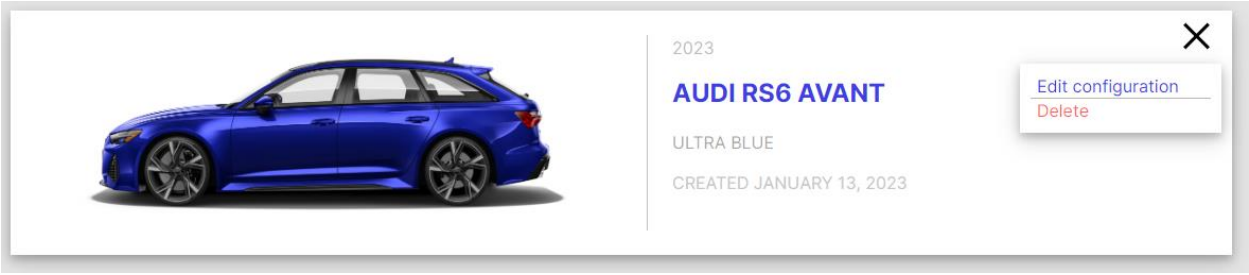
Ako korisnik ima izrađenu barem jednu konfiguraciju na naslovnoj stranici korisniku se prikazuju postojeće konfiguracije. Također kao i u prijašnjem slučaju ima mogućnost stvaranja nove konfiguracije. Primjer ovakve naslovne stranice vidimo na slici 4.6. gdje je prikazan izgled stranice, te spremljene konfiguracije korisnika. Prilikom pregleda spremljenih konfiguracija korisnik ima opciju urediti ili obrisati konfiguraciju ako to želi. Na svakoj konfiguraciji postoji tipka *hamburger*, čijim pritiskom dobivamo odabir želimo li uređivati ili brisati konfiguraciju. Navedenu mogućnost je prikazana na slici 4.7. Ako korisnik obriše svoju konfiguraciju, te se konfiguracija uspješno ukloni iz baze podataka, bude obaviješten o uspješnom izvršavanju naredbe, odnosno brisanju konfiguracije što je prikazano na slici 4.8.



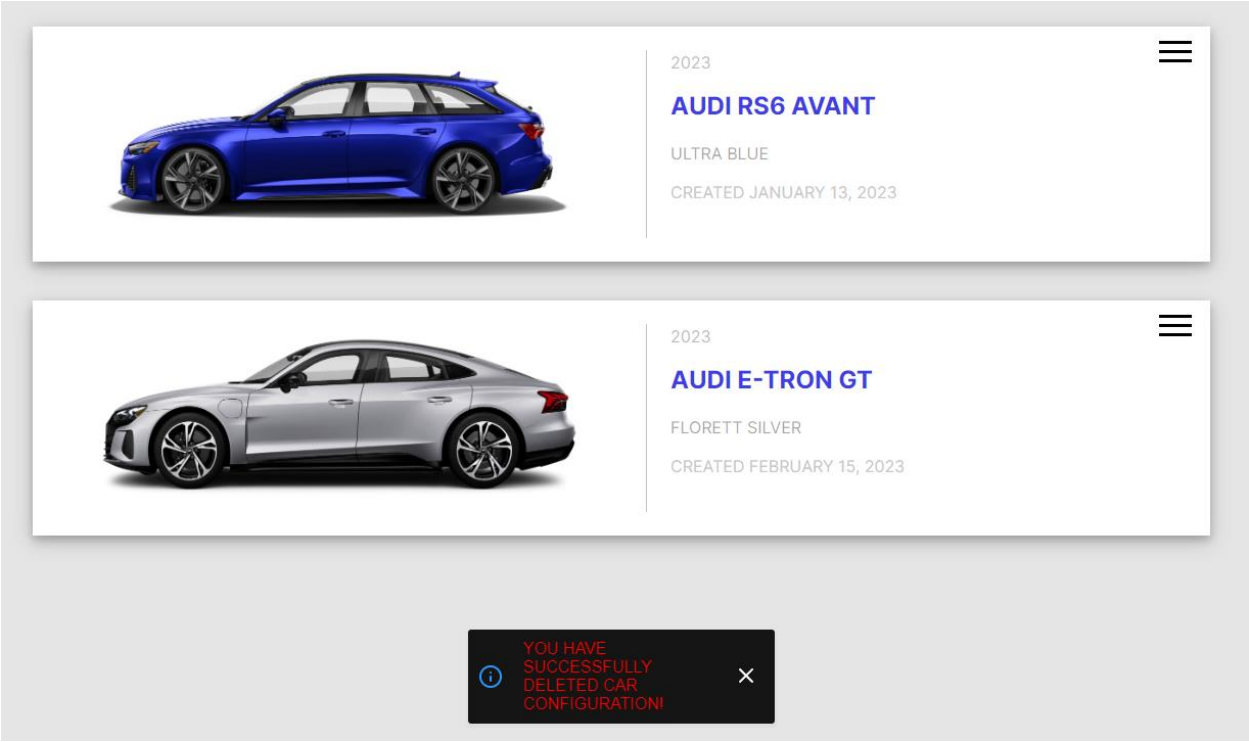
Slika 4.5 *Izgled naslovne stranice bez konfiguracija*



Slika 4.6 *Izgled naslovne stranice s konfiguracijama*



Slika 4.7 *Dodatne mogućnosti za brisanje ili uređivanje konfiguracije*



Slika 4.8 *Izgled stranice prilikom brisanja konfiguracije*

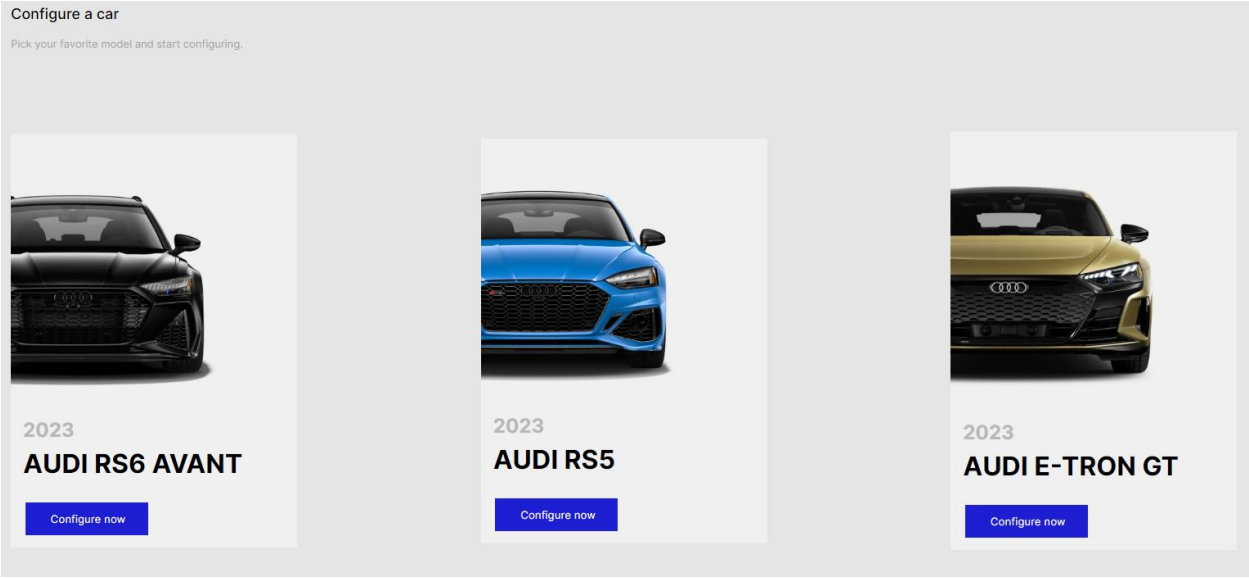
Na slici 4.9 prikazan je primjer koda koji služi za prikazivanje konfiguracija na naslovnoj stranici, ako one postoje. Kod uzima potrebne podatke iz baze podataka koji se unutar baze pretražuju uz pomoć korisnikovog jedinstvenog identifikacijskog ključa. Također u kodu je prikazano postizanje željenog dizajna na kojem se ispisuju potrebni podaci i fotografija korisnikove konfiguracije.

```
return (
  <div key={id}>
    <ConfigurationsCarousel>
      <div className="renderCars">
        <div className="renderCars_img">
          <img
            className="configurationsImg"
            src={`/Assets3/View=Side, Color=${carData.paint}, Wheel Style=${carData.wheel}.png`}
          />
        </div>
        <div className="renderCars_info">
          <p>2023</p>
          <h2>{carData.name}</h2>
          <p>{carData.paint}</p>
          <p>
            Created{" "}
            {carData.date.toDate().toLocaleDateString("en-US", {
              year: "numeric",
              month: "long",
              day: "numeric",
            })}
          </p>
        </div>
        <div className="configurations-hamburger">
          <Hamburger
            distance="sm"
            label="Show menu"
            toggled={menuOpen}
            toggle={setMenuOpen}
            onToggle={(toggled) => {
              if (toggled) {
                setMenuOpen(true);
              } else {
                setMenuOpen(false);
              }
            }}
          />
          {menuOpen && (
            <div className="configurations-menu">
              <button
                onClick={() => {
                  editConfig(carData, id);
                  setMenuOpen(false);
                  setCarEditToId(id);
                  setCar(carData.name);
                  setPaint(carData.paint);
                  setWheel(carData.wheel);
                  setInterior(carData.interior);
                }}
              >
                Edit configuration
              </button>
              <button onClick={() => handleDelete(carData, id)}>
                Delete
              </button>
            </div>
          )}
        </div>
      </div>
    </ConfigurationsCarousel>
  </div>
);
```

Slika 4.9 *Primjer koda za izlistavanje konfiguracija*

4.4. Izbor vozila

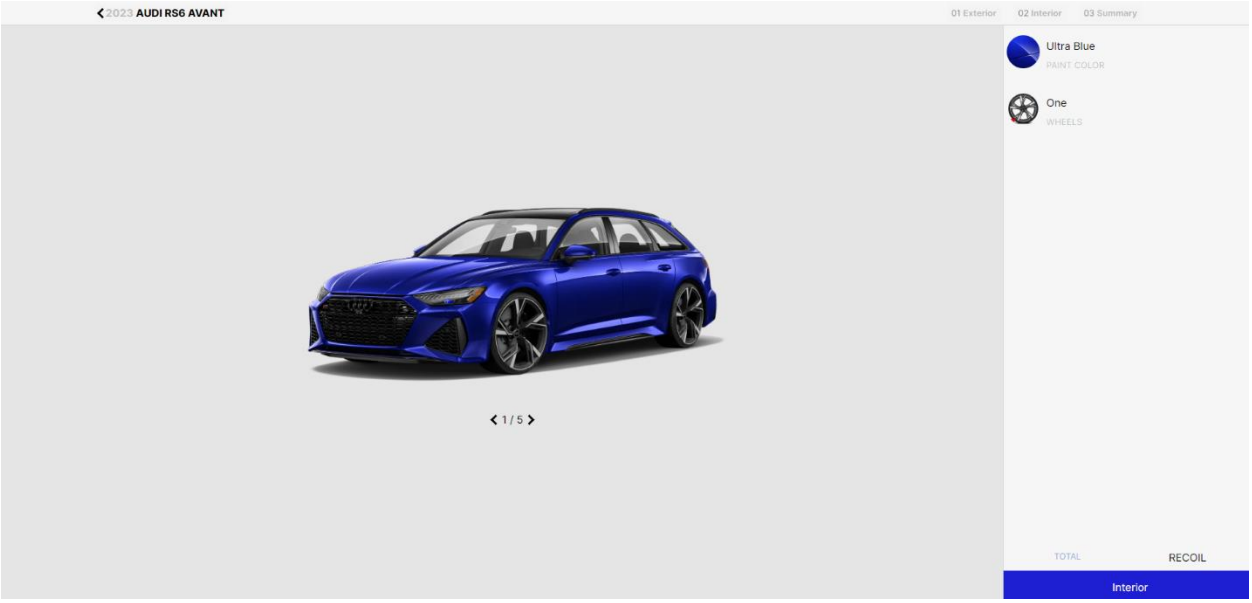
Pritiskom na tipku *Configure a car*, korisnik dobije mogućnost izbora između tri vozila koje želi konfigurirati što je prikazano na slici 5.0. Pritiskom tipke *Configure now* zaključuje svoj zbor, te dalje nastavlja s konfiguracijom vozila.



Slika 5.0 *Izgled stranice za izbor vozila*

4.5. Eksterijer vozila

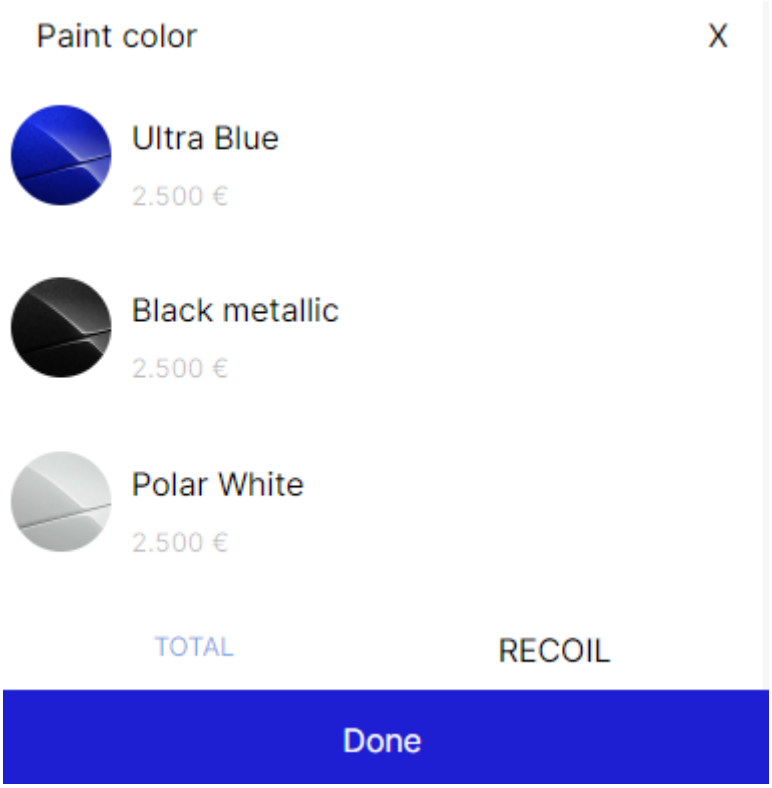
Na slici 5.1 prikazana je trenutna konfiguracija odabranog vozila gdje također u desnom stupcu vidimo korisnikov odabir boje auta te kotača, odnosno stila automobila. Korisnik ovdje ima mogućnost pregleda automobila s trenutnim izborom boje i stila vozila.



Slika 5.1 *Izgled stranice za vanjštinu vozila*

4.5.1 Boja vozila

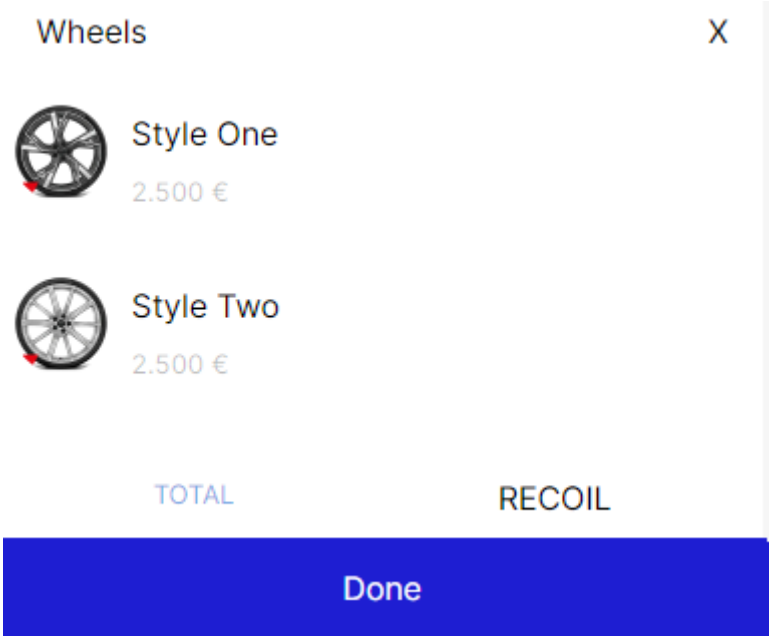
Na slici 5.2 prikazana je mogućnost dana korisniku prilikom izbora boje za vozilo koje se trenutno konfigurira. Prilikom izbora boje, vozilo mijenja boju u onu izabranu od strane korisnika.



Slika 5.2 *Izbor boje vozila*

4.5.2 Stil vozila

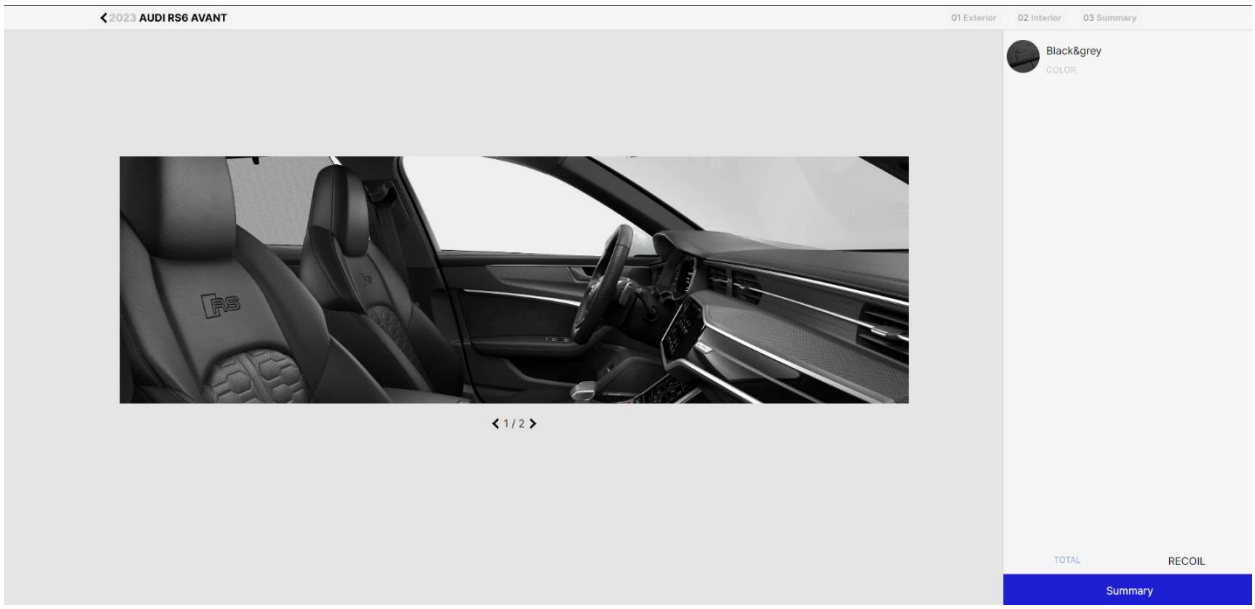
Na slici 5.3 prikazana je mogućnost korisniku da bira stil svog vozila. Prilikom izbora stila, vozilo mijenja felge, boju dijelova auta kao što je boja oko prozora, znak proizvođača,… Također svaki stil ima i cijenu koja, ako se odabere neki od stilova, se dodaje cijeni automobila.



Slika 5.3 *Izbor stila vozila*

4.6. Interijer vozila

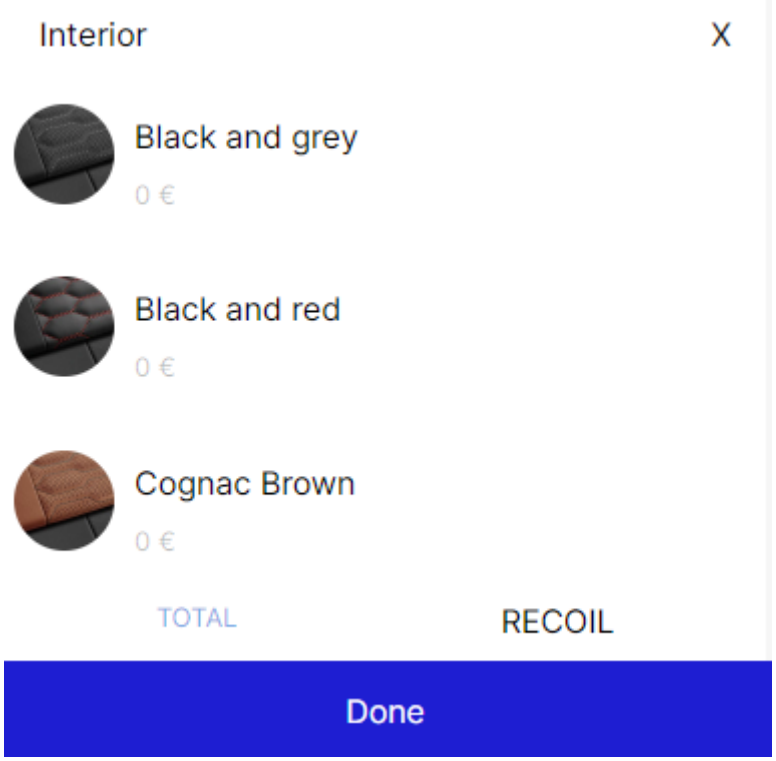
Na slici 5.4 prikazana je trenutna konfiguracija za interijer vozila. Ovdje korisnik ima mogućnost promijeniti boju interijera vozila.



Slika 5.4 *Izgled stranice za vanjštinu vozila*

4.6.1 Boja unutrašnjosti vozila

Na slici 5.5 prikazan je izbor boja za unutrašnjost vozila. U ovom prozoru korisnik izabire boju te se ona primjenjuje na trenutnu konfiguraciju vozila, odnosno mijenja se boja sjedala unutar vozila te detalji poput rukohvata i slično.

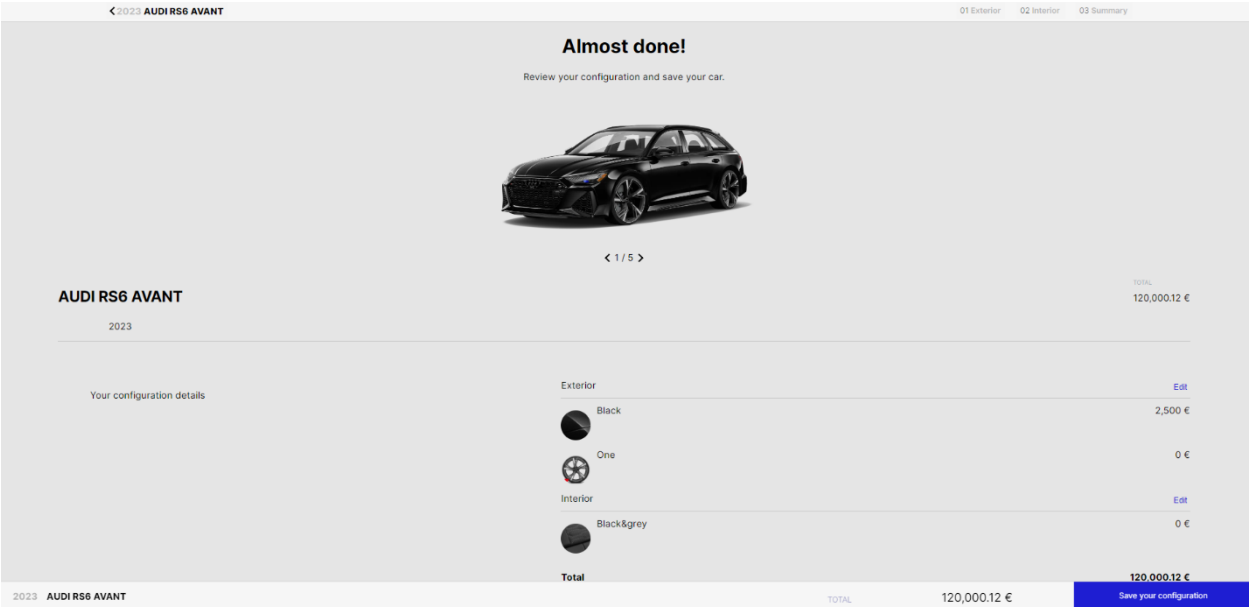


Slika 5.5 *Izbor boja za unutrašnjost vozila*

4.7 Sažetak

Nakon izbora eksterijera i interijera vozila, korisnik dolazi na stranicu koja mu daje sažetak konfiguracije. Ovdje korisnik ima pregled svih izbora konfiguracije vozila. Unutar sažetka pritiskom na tipku *Edit* moguće je promijeniti točno određeni dio napravljene konfiguracije. Nakon pregleda sažetka, pritiskom tipke *Save your configuration* konfiguracija se sprema u bazu podataka pod svojim jedinstvenim identifikacijskim ključem koji se veže na identifikacijski ključ korisnika.

Na slici 5.6 prikazan je izgled upravo opisane stranice sažetak konfiguracije.



Slika 5.6 *Izgled sažetka stranice*

5. ZAKLJUČAK

U ovom završnom radu objašnjeno je te su opisane sve metode i funkcionalnosti web aplikacije Car Configurator. Također je objašnjeno što je bilo sve potrebno, odnosno koje su tehnologije korištene prilikom izrade same web aplikacije. Svrha Car Configurator aplikacije jest da se korisnicima olakša izbor vlastite konfiguracije vozila, brzi pristup njima te uređivanja ili brisanja istih. Izradu i korištenje baze olakšao je Firebase, odnosno Firestore. Kroz izradu baze podataka pokazao se kao intuitivan alat, koji pruža korisniku brojne mogućnosti prilikom korištenja baze podataka. Nadalje React korišten kao programski okvir za izradu ove web aplikacije omogućio je razvoj aplikacije na efikasan način te su se unutar ovog programskog okvira razvile sve funkcionalnosti stranice navedene u ovom radu.

LITERATURA

[1] Visual Studio Code, https://code.visualstudio.com/ , kolovoz 2022.

[2] JavaScript, https://www.javascript.com/ . kolovoz 2022.

[3] Yarn, https://yarnpkg.com/ , kolovoz 2022.

[4] React, https://reactjs.org/ , kolovoz 2022.

[5] Figma, https://www.figma.com/about/ , kolovoz 2022.

[6] HTML, https://html.com/ , rujan 2022.

[7] CSS, https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS , rujan 2022.

[8] Emotion, https://emotion.sh/docs/introduction , rujan 2022.

[9] Firestore, https://cloud.google.com/firestore#all-features , rujan 2022.

SAŽETAK

Web aplikacija Car Configurator je jednostavna i intuitivna za korištenje. Omogućuje korisnicima jednostavan izbor željene konfiguracije, te uređivanja i brisanja iste. Za izradu same stranice korišten je React kao glavni dio klijentske strane aplikacije, te Firebase za bazu podataka. Radi lakše predodžbe izgleda korištena je Figma, kako bi željeni krajnji izgled stranice bio bolje prikazan.

Ključne riječi: konfigurator vozila, React, JavaScript, Firebase, Firestore, Figma

ABSTRACT

Web application – Car Configurator

The car configurator web application is easy to use and intuitive. It allows users to easily choose the desired configuration, as well as edit and delete it. For the making of the application, React was used as the main part of frontend, and Firebase was used for the database. Figma was used to make it easier to imagine the layout, in order to easily show the presentation of the desired final appearance of the web application.

Key words: Car configurator, React, JavaScript, Firebase, Firestore

ŽIVOTOPIS

Zovem se Zvonimir Grizelj, rođen 14.9.1996. u Osijeku. Godine 2015. završio sam Elektrotehničku i prometnu školu u Osijeku. Godine 2019. upisao sam stručni studij, smjer Računarstvo na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Tečno govorim i razumijem engleski jezik, te poznajem i osnove njemačkog jezika.

_____