Sveučilište u Zagrebu

Fakultet elektrotehnike i računarstva

Seminar 1

Ak. god. 2021./2022.

Web aplikacija za generiranje dinamičkih izvještaja

Dario Mesić

Datum predaje: *17.1.2022.*

Nastavnik: *Mirko Sužnjević*

Sadržaj

1. Opis projektnog zadatka

Tema ovog seminarskog rada je izrada web aplikacije koja će generirati dinamičke izvještaje. Aplikacija će omogućiti korisnicima lakši pristup njihovim privatnim ili javnim podacima te će ih prikazati na oku ugodan način.

Budući da živimo u svijetu u kojemu se svaki dan susrećemo s mnoštvo različitih i mnogobrojnih podataka, korisno je razviti aplikaciju koja će ih na neki način posložiti i prikazati korisniku u obliku pametne tablice. Za ovaj projekt koristila se takozvana "Pivot tablica" koja omogućuje korisniku da prikazuje podatke u dvije osi. Dakle, zapravo je riječ o složenijoj vrsti tablice koja će podatke prikazati na x i y osi.

Praktični dio ovoga rada je izrada "frontend" web aplikacije u frameworku *Vue*, ali prije svega treba se dobro upoznati sa strukturom te svim funkcionalnostima aplikacije kako bi se mogle prikazati njegove mogućnosti.

Budući da je podataka puno i da moraju biti na raspolaganju svim korisnicima, bili oni vješti s informatičkim sposobnostima ili ne, gotovo je sigurno da će razina informatičke pismenosti drastično varirati od korisnika do korisnika. Iz tog razloga vrlo je važno da korisničko sučelje bude intuitivno i jednostavno za korištenje, a istovremeno funkcionalno i izgledom atraktivno. Kako postići ciljeve povećanja zainteresiranosti postojećih i privlačenja novih članovima ako je aplikacija ružna i zbunjujuća za korištenje?

Treba imati na umu kako već postoji puno različitih rješenja na ovu temu na Internetu. Međutim, većina se tih aplikacija plaća jer dolaze od strane multimilijunskih tvrtki među kojima je i *DevExtreme*. Ova aplikacija je uzela jednu od njihovih "open-source" komponenti, "Pivot tablicu", i napravila od nje komponentu koja će biti prikladna za područje koje će se svidjeti korisniku.

Rijetko koja web stranica nudi ovakve mogućnosti već ukomponirane i najčešće traže da se ukomponiraju u vlastitoj web stranici. S obzirom na to da nisu svi ljudi rođeni kao "web developeri", ova aplikacija će ponuditi najbolje za oba svijeta. "Pivot tablice" mogu se

pronaći jedino u *Excel* tablicama. Međutim ukomponiranje podataka u takvu tablicu je puno složenije i zahtijeva puno vremena dok se ovdje to može napraviti u par koraka.

2. Arhitektura i dizajn sustava

Arhitektura programske podrške se sastoji od tri podsustava:

- **Web stranica**
- **API server**
- **Baza podataka**

2.1. Web stranica

Za izradu web stranice - sučelja putem kojeg krajni korisnici koriste sustav koristio se **Vue.js** - *Javascript MVC framework* otvorenog koda za kreiranje korisničkih sučelja i web aplikacija. Svrha web stranice (aplikacije) je da poslužuje informacije te daje pristup korisniku (posjetitelju stranice) na intuitivan i oku ugodan način. Web aplikacija koristi REST API server za dohvat podataka koji se onda prikazuje korisniku u obliku "Pivot tablice" s grafovima.

2.2. API server

Funkcionalnost koju sustav definira dostupna je putem sučelja - REST API servera. Ovaj dio cijelokupnog sustava definira načine obrade podataka te funkcionalnosti istog. Neke od velikih prednosti razdvajanja samog prikaza korisniku Web stranice i dijela implementacije funkcionalnosti API servera je da se u budućnosti može relativno lako dodati nove funkcionalnosti u web aplikaciju ili u bilo kojoj drugu aplikaciju koja želi koristiti ovo sučelje a ima za to potrebne ovlasti. Takoder olakšan je razvoj ako je sama funkcionalost sustava razdvojena od prikaza korisniku.

Za implementaciju Web API servera koristi se razvojni okvir (framework) ASP.NET. ASP.NET je okvir za web-aplikacije otvorenog koda na strani poslužitelja dizajniran za web

razvoj za izradu dinamičkih web stranica. Razvio ga je *Microsoft* kako bi programerima omogućio izradu dinamičkih web stranica, aplikacija i usluga.

Arhitektura sustava ovog sustava implementiranog pomoću Spring boot razvojnog okvira temelji se na MVC (Model-View-Controller) konceptu. Karakteristike MVC koncepta je odvojen razvoj pojedinih dijelova aplikacije što rezultira između ostaloga sistematski podijeljenom kodu te jednostavnije razvijanje i testiranje istog. Aplikacija se spaja na API koji je framework kreirao te uzima podatke direktno s njega [5].

2.3. Baza podataka

Baza podataka je alat za prikupljanje i organizaciju podataka. One mogu pohraniti podatke o osobama, proizvodima, narudžbama i bilo čemu drugome. Mnoge baze podataka započinju kao popisi u programima za obradu teksta ili kao proračunske tablice. Kako popis raste, na njemu se pojavljuju zalihosti i nedosljednosti među podacima.

Za aplikaciju praćenja senzorskih očitanja koristila se baza *PostgreSQL* koja je služila za pristup svim podatcima koje su se koristile u aplikaciji. Razlog korištenja baze podataka je u brzoj i jednostavnoj pohrani, izmjeni i dohvaćanju podataka za daljnju obradu.

2.3.1. PostgreSQL

PostgreSQL je besplatan sustav za upravljanje bazama podataka otvorenog koda. Riječ je o platformi koja nudi različite vremenske serije s ciljem implementacije, promatranja, učenja i automatizacije svih vrsta sustava, aplikacija i poslovnih procesa u različitim radnim okruženjima. Glavne značajke baze su:

- sposobnost promatranja i automatizacije ključnih sustava, infrastrukture, aplikacija i poslovnih procesa,
- analizirati i automatizirati senzore i uređaje u stvarnom vremenu, koji generiraju informacije za administrativne zadatke,
- više opcija instrumentacije koje otkrivaju obrasce uporabe i odatle se mogu stvoriti nove poslovne mogućnosti.

Mnogi popularni online servisi koriste *PostgreSQL* za održavanje svojih glavnih relacijskih baza podataka, među kojima su *Reddit*, *Skype*, *Instagram*, *The Guardian* i drugi. U semi-

narskom radu pristup bazi je bio onemogućen tako da se podatke uzimalo preko *Rest API* sučelja koje je opisano kasnije [2].

3. Implementacija

Za implementaciju aplikacije koristili su se različiti razvojni alati s namjerom da se na najlakši način kreiraju određeni dijelovi aplikacije. Razvojni alati su skup programskih alata koji omogućavaju kreiranja aplikacija za određene skupine programa, programskog okruženja, hardvera, OS-s i sličnih platformi. Sastoje se od određene skupine gotovih programskih rješenja koji imaju mogućnost komunikacije sa specifičnim programom ili platformom. Najpoznatiji razvojni alati za korištenje su *GitHub, Visual Studio* itd.

3.1. Korištene tehnologije i alati

Korištene tehnologije za izradu ove aplikacije objašnjene su i detaljno opisane u ovome poglavlju. Najprije je opisan programski jezik u kojem je aplikacija pisana, *JavaScript* zajedno s editorom u kojem se programiralo, *Visual Studio Code*, zatim korištenje "frontend" grafičkog sučelja *VueJS* komponenti *Bootstrap* te *DevExtreme*, te uporaba *JSON* i *Rest API-a*.

3.1.1. Programski jezik JavaScript

JavaScript je skriptni programski jezik, koji se izvršava u web pregledniku na strani korisnika. Napravljen je da bude sličan *Javi*, zbog lakšega korištenja, ali nije objektno orijentiran kao *Java*, već se temelji na prototipu i tu prestaje svaka povezanost s programskim jezikom *Java*.

3.1.2. Visual Studio Code

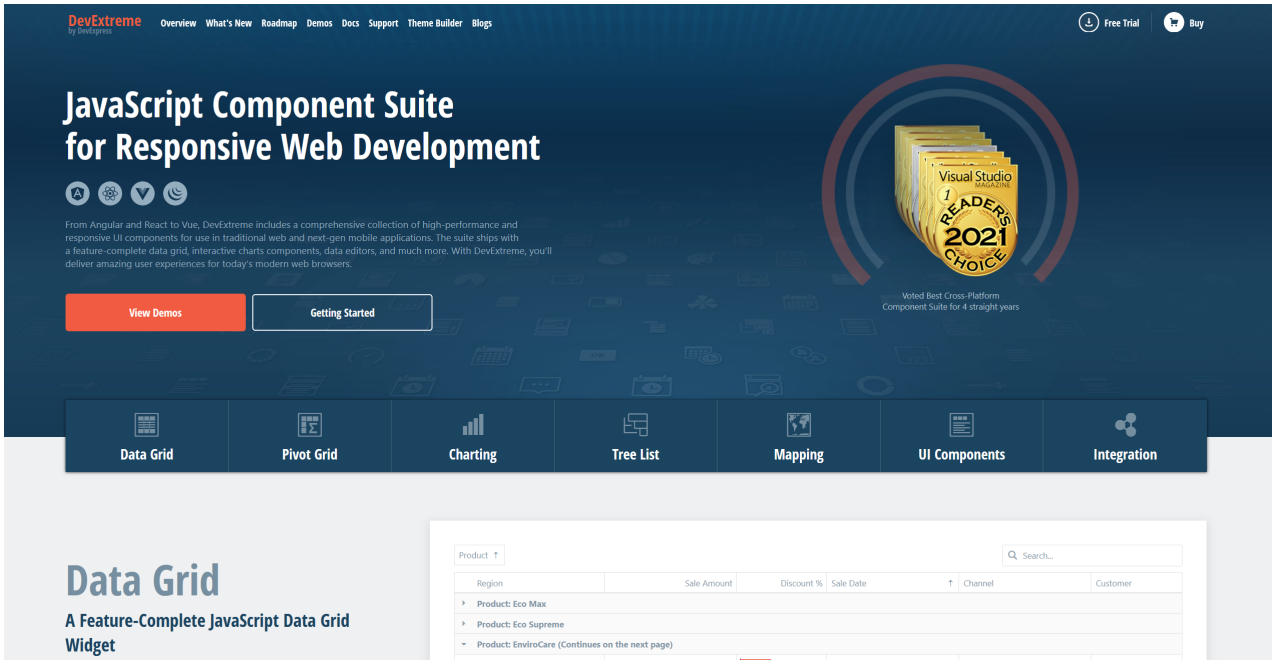
Visual Studio COde Visual Studio Code je uređivač izvornog koda koji je napravio *Microsoft* za **Windows**, **Linux** i **macOS**. Značajke uključuju podršku za otklanjanje pogrešaka, isticanje sintakse, inteligentno dovršavanje koda, isječke, refaktoriranje koda i ugrađeni Git [3].

3.1.3. VueJS

VueJS je *JavaScript* okvir okvira otvorenog koda model-view-viewmodel za izradu korisničkih sučelja i aplikacija na jednoj stranici. Izradio ga je Evan You, a održavaju ga on i ostali članovi njegove grupe. [4].

3.1.4. DevExtreme

DevExtreme je komponenta napravljena od strane tvrtke DevExpress koja se bavi izradom software proizvoda. Riječ je o razno raznim "open-source" komponentama koje su dostupne za više programskih jezika, ali i grafičkih sučelja među kojima je i *VueJS*. U kontekstu ovog projekta, *DevExtreme* je ponudio komponentu "Pivot tablice" u koju se stavljaju korisnički podaci. Osim "Pivot tablice", grafovi koji prikazuju što se sve odvija u tablici su isto tako gotova komponenta koja se tamo pronašla.

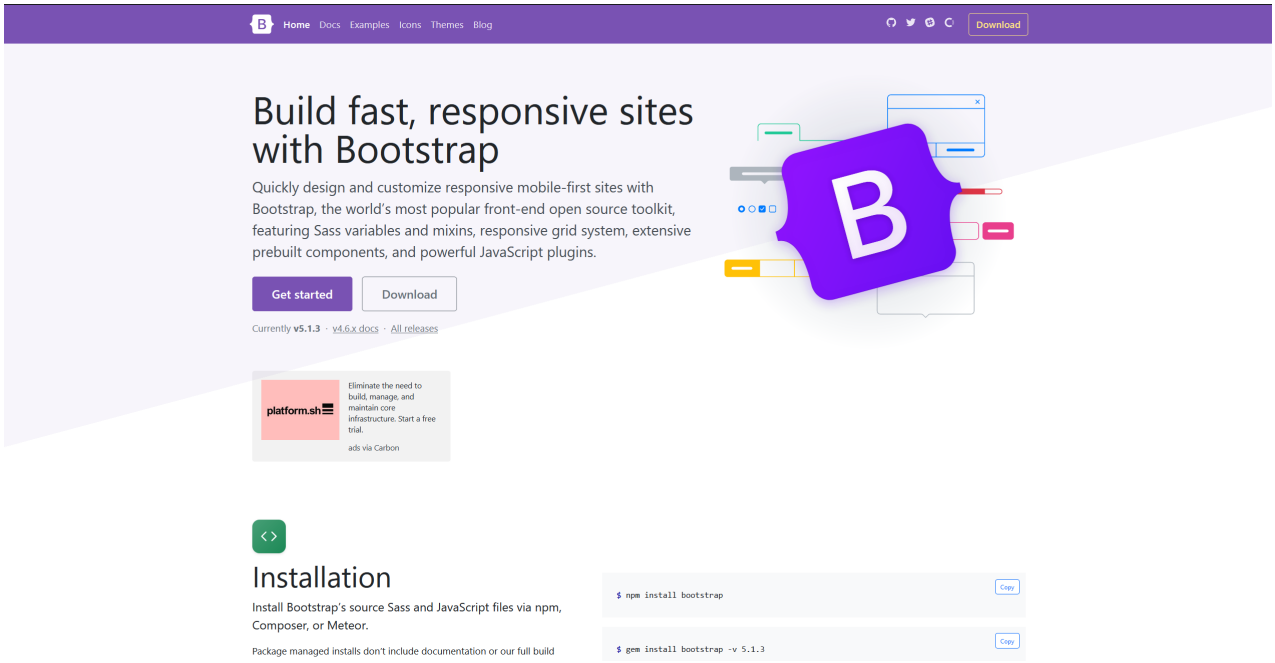


Slika 3.1: Komponenta DevExtreme

3.1.5. Bootstrap

Bootstrap je besplatni CSS okvir otvorenog koda usmjeren na responzivan "frontend" web razvoj za mobilne uređaje. Sadrži CSS i *JavaScript* predloške dizajna za tipografiju, obrasce, gumbe, navigaciju i druge komponente sučelja. U kontekstu ovog projekta, *Bootstrap* je bio

korišten za razne svrhe. Najprije se uz pomoću njegovih gotovih HTML i CSS komponenti dizajnirao izgled stranice za aplikaciju. Nakon toga se koristio za izradu navigacijske trake te na kraju za izradu bočne trake u kojoj su bile smještene sve stilske promjene.

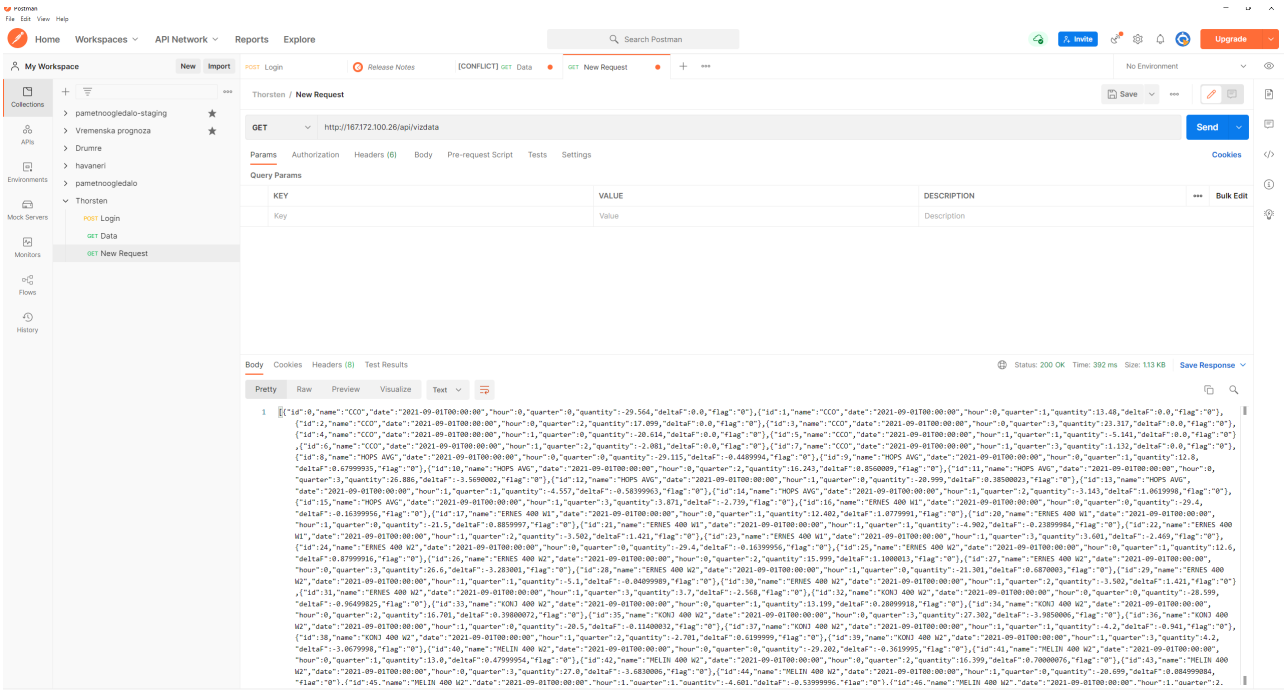


Slika 3.2: Komponenta Bootstrap

3.1.6. Rest API

Zadaća pozadinskog sustava je povezati bazu podataka u kojoj su spremljeni svi podaci sustava s klijentskim aplikacijama koje trebaju čitati i upravljati tim podacima uz poštovanje aplikacijske logike i ograničenja sustava. To je način pristupanja mrežnim resursima korištenjem "HTTP" protokola i odgovarajućih ruta koje označavaju željeni resurs.

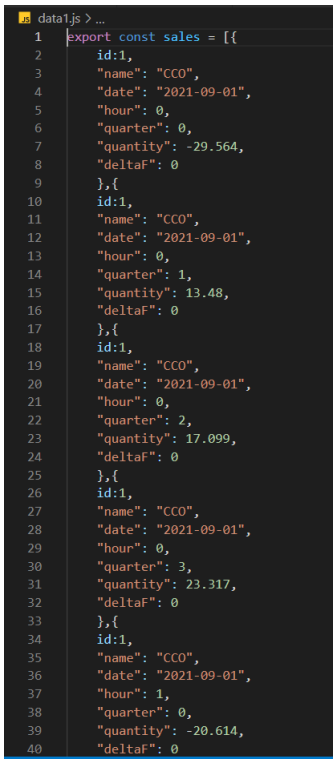
Odgovor najčešće dolazi u obliku *JSON* ili *XML* formata. Vanjske aplikacije mogu koristiti *REST API* za upite i ažuriranja aplikacijskih podataka. Njegovi resursi se mogu koristiti bez ikakve konfiguracije, a podrška kreira, radi upit, ažurira i briše operacije na resursima koristeći standardne metode "HTTP", "GET", "POST", "PUT" i "DELETE" [6].



Slika 3.3: Testiranje API sučelja pomoću programa Postman.

3.1.7. JSON

JSON je otvoreni standardni format datoteke i format za razmjenu podataka koji koristi ljudski čitljiv tekst za pohranu i prijenos podataka objekata koji se sastoje od parova atribut-vrijednost i nizova [1]. Kao rezultat API poziva dobio se *JSON* format i zato je bilo nužno poznavanje ovog teksta.



Slika 3.4: Primjer JSON datoteke korištene u aplikaciji

3.2. Ispitivanje programskog rješenja

U ovom potpoglavlju prikazani su najbitniji dijelovi koda koji su zaduženi za povezivanje s vanjskim sustavima, poput baze podataka, API sučelje itd.

Ispitni slučaj 1: Povezivanje s komponentom DevExtreme

Kako bi se koristile funkcionalnosti koje koristi *DevExtreme* komponenta, potrebno je bilo na početku instalirati sve njezine pakete s npm modulima. Nakon toga, *DevExtreme* se stavlja u "index.html" file i glavnoj "main.js" funkciji iz koje se pokreće čitava aplikacija. "Pivot ta-

blica" zajedno sa svim svojim funkcionalnostima uzeta je direktno iz paketa kao komponenta i onda je jednostavno ukomponirana u html datoteku.

Izvorni kod:

Listing 3.1: Pivot table

```
<DxPivotGrid
        id="pivotgrid"
        :ref="pivotGridRefKey"
        :data-source="dataSource"
        :allow-sorting-by-summary="true"
        :allow-sorting="true"
        :allow- filtering ="true"
        :show-borders="true"
        :show-column-grand-totals="columnGrandTotal"
        :show-row-grand-totals="rowGrandTotal"
        :show-row-totals="rowTotal"
        :show-column-totals="false"
        @cell-click="onCellClick"
        @exporting="onExporting"
        @cell-prepared="onCellPrepared"
        :on-context-menu-preparing="onContextMenuPreparing"
>
<DxFieldChooser :enabled="false"/>
<DxScrolling mode="standard" />
<DxFieldPanel
        :show-column-fields="showColumnFields"
        :show-data-fields="showDataFields"
        :show- filter - fields =" showFilterFields "
        :show-row-fields="showRowFields"
        :allow- field -dragging="true"
        : visible ="true"
/>
<DxExport :enabled="true"/>
</DxPivotGrid>
```

Ispitni slučaj 2: Povezivanje s API serverom

Kao što je već navedeno, Web aplikacija je preko API servera povezana s bazom podataka. Pristup API serveru se odvija preko URL-a koji se fetcha u posebnoj klasi i onda se response šalje direktno u HTML kod. Javascript zajedno s *VueJS*-om koristi već gotove funckije koje olakšavaju komunikaciju između Javascripta i HTML-a. U ovom slučaju koristila se asinkrona funkcija koja se poziva svaki puta kada se pokrene aplikacija.

Izvorni kod:

Listing 3.2: Communication with API

```
const API_URL = 'http:// localhost :8080/api/VizData';
```

```
class DataService {
  getData() {
    return fetch(API_URL, {
      method: "GET",
  })
  .then((response) => {
      if (!response.ok) {
          throw new Error(response.error)
      }
      return response.json();
  })

  }
}

export default new DataService();
```

Ispitni slučaj 3: Prikaz podataka u tablici

Nakon što su podaci preuzeti sa servera, vrijeme je da ih se prikažu na tablici. Za to je zaslužna created() funkcija definirana od strane *VueJS* framerworka koja se pokreće svaki put kada se pokrene i aplikacija. *DevExtreme* komponenta prima preuzet sa servera i pohranjuje ga u tablicu. Na isti način bi došlo do promjene u slučaju nekog drugog dataseta.

Izvorni kod:

Listing 3.3: Inserting data to Pivot table

```
created(){
        this.getData();
},
methods: {
        // DOHVACANJE PODATAKA IZ BAZE

        async getData(){
        var podaci =[];
        await DataService.getData().then(
        (res) => {
        podaci = res;
        },
        error => {
        this.content =
                (error.response && error.response.data && error.response.data.message) ||
                error.message ||
                error.toString();
        this.response = true;
        console.log(error);
        }
        );
        this.dataSource= new PivotGridDataSource({
        fields: [{
        caption: 'Date',
        width: 120,
```

```
                dataField : 'date',
                area: 'row',
                sortBySummaryField: 'Total',
                expanded: true, // VAZNO!!!!!!!!
                }, {
                caption : 'Hour',
                dataField : 'hour',
                width: 150,
                area: 'row',
                expanded: true,
                },
                {
                caption : 'Name',
                dataField : 'name',
                width: 150,
                area: 'column',
                expanded: true,
                },
                {
                caption : 'Quarter',
                dataField : 'quarter',
                width: 150,
                area: 'row',
                expanded: true,
                },{
                caption : 'Quantity',
                dataField : 'quantity',
                dataType: 'number',
                summaryType: 'sum',
                width: 150,
                format :1,
                area: 'data',
                expanded: true,
                // summaryDisplayMode: 'sum'
                },{
                caption : 'deltaf (mHz)',
                dataField : 'deltaF',
                dataType: 'number',
                summaryType: 'sum',
                width: 150,
                area: 'data',
                format :2,
                expanded: true,
                // summaryDisplayMode: 'sum'
                }],
                store : podaci,
                })
                },
        }
```

Ispitni slučaj 4: Prikaz podataka an grafu

Kako bi se podaci iz tablice prikazali na grafu, potrebno ih je povezati direktno s tablicom, tako da kada god dođe promjena u tablici, promjena se prikaže na grafu. Ovo je već isto tako ugrađeno kao *DevExtreme* komponenta tako da nije bilo potrebno puno kodiranja. Od *VueJS*

funkcija pomogao je mounted() koji se pokreće svaki put kada se dogodi neka promjena u aplikaciji. Izvorni kod:

Listing 3.4: Chart data

```
mounted() {
    const pivotGrid = this.$refs[pivotGridRefKey].instance;
    const chart = this.$refs.chart.instance;
    pivotGrid.bindChart(chart, {
    dataFieldsDisplayMode: 'splitPanes',
     alternateDataFields: false,
    });
}
```

Ispitni slučaj 5: Promjena postavki definiranih od strane korisnika

Korisnik je najbitniji u ovoj aplikaciji, tako da mu je mnogo stavki koje aplikacija nudi stavljeno na mogućnost mijenjanja. Na taj način korisnik može mijenjati pozadinsku boju tablice, koje stavke želi prikazati u tablici ili ne.. Veliku ulogu ovdje je imala stavka komunikacije između komponenti koje *VueJS* nudi. Kada se dogodi promjena u jednoj komponenti, šalje se funkcija koju prima druga komponenta i automatski dolazi do promjene kod nje. Ovdje je veliku važnost imala metoda watch koju nudi *VueJS* jer svaki puta kada se promijeni određena varijabla, poziva se ta funkcija koja sadrži promijenjenu vrijednost te varijable. Izvorni kod:

Listing 3.5: Communication between components

```
<template>
    ..
    <div class="col-6">

<h3>Background color</h3>

        <color-picker style="margin-top:15px" :width=200 :height=200 v-model="boja"></color-picker>
        <p style="left: 50%;position: absolute;margin-top:10px">{{ boja }}</p>

    </div>
    ..
</template>
<script>
..
watch:{
    boja: function(){
        this.$emit('clickedItem', this.boja);
    }
}
</ script >
```

3.3. Upute za puštanje u pogon

Kako bi korisnik mogao dobiti uvid u aplikaciju, omogućena su mu dva pristupa opisana u sljedećim poglavljima.

3.3.1. Produkcijsko okruženje

Najlakši način za pokretanje aplikacije je preko URL-a. Većina današnjih web aplikacija se pokreće na taj način i zato je najbolje omogućiti ovakvoj aplikaciji da bude dostupna online bilo kojem korisniku. Kao produkcijsko okruženje koristio se *Heroku*. *Heroku* je cloud platforma kao usluga koja podržava nekoliko programskih jezika. *Heroku* dozvoljava svakom prijavljenom korisniku da ima 5 različitih aplikacija pohranjenih preko URL-a. Ovo je izuzetno dobar način za pohranu bilo koje aplikacije, ali pokretanje same zna biti dosta spor proces i mukotrpan ako se stalno treba nešto mijenjati u aplikaciji. Međutim, s obzirom da se nije raspolagalo s puno resursa, ovaj način je bio najfleksibilniji. Link za aplikaciju dostupan je preko *GitHub-a* te je spreman za uporabu u stvarnom vremenu.

3.3.2. Lokalno pokretanje za razvoj

Za lokalno pokretanje aplikacije potrebno je preuzeti projekt s *GitHub-a* te importati ga u *Visual Studio Code*. Nakon toga potrebno je otvoriti /src folder u terminalu te pokrenuti naredbu npm install kako bi se instalirali svi paketi koje aplikacija sadrži. Nakon što se instaliraju svi paketi, aplikaciju je potrebno pokrenuti na istoj poziciji u terminalu, naredom npm run serve. Nakon toga pokrenut će se localhost u kojem će korisnik imati uvid u aplikaciju, Korisnik ima mogućnost pokretanja aplikacije na različitim uređajima, poput računala, tableta, mobitela itd. Treba imati na umu kako aplikacija nije pretjerano skalabilna, tako da će se na manjim zaslonima izgubiti na određenim funkcionalnostima. Ako korisnik želi zaustavili testiranje aplikacije potrebno je ugasiti terminal te localhost na pretraživaču.

3.4. Mogući problemi

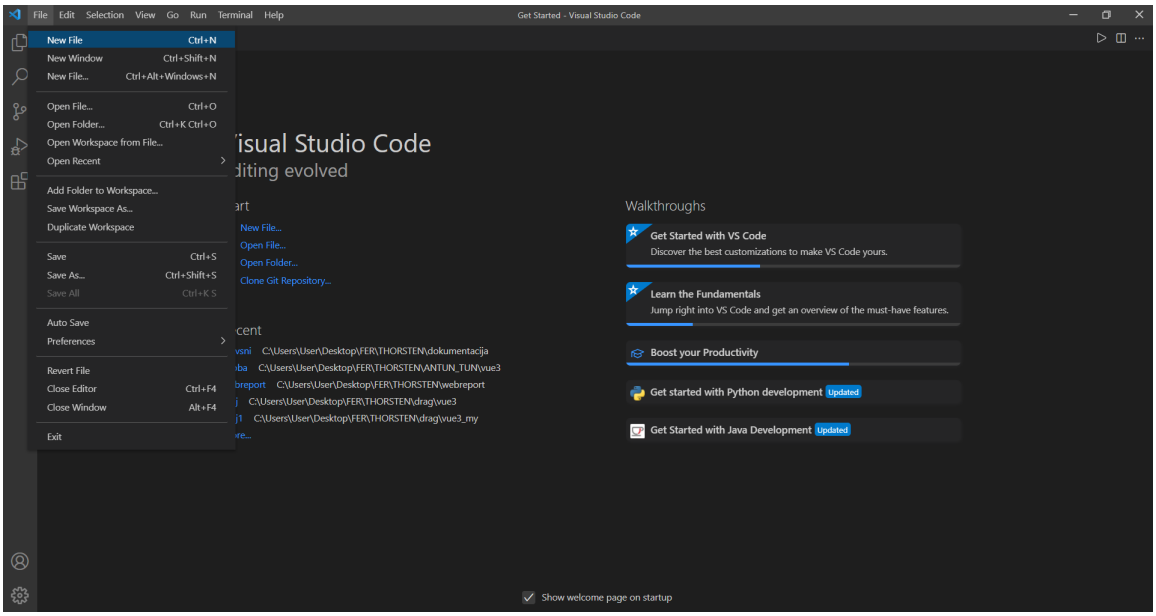
Naravno, postoje određeni problemi koji se mogu dogoditi prilikom izvođenja programa. Najčešći problem je da računalo neće moći pokrenuti aplikaciju lokalno zbog prevelike veličine paketa, ako se to dogodi najbolje je aplikaciju pokrenuti preko *Heroku*-a. Još jedna mogućnost je da jedna od baze podataka jednostavno ne bude funkcionalna te tako korisnik neće moći uopće pristupiti svojim podacima. U takvom slučaju "Pivot tablica" i grafovi će biti prazni.

4. Web aplikacija

Kako bi korisnici mogli koristiti svoje podatke pohranjene u bazi, izrađena je web aplikacija kao glavni način interakcije sa sustavom. Koristeći službenu dokumentaciju te raznu literaturu aplikacije se izrađivala kontinuirano, kroz više iteracija, nadograđivala novijim tehnologijama, modernijim bibliotekama kako bi rezultat na kraju bio što bolji.

4.1. Stvaranje projekta

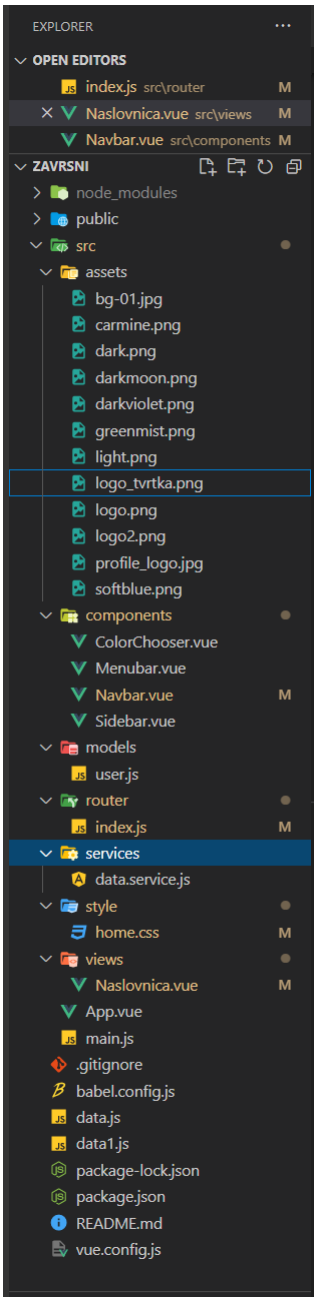
Ulaskom u *Visual Studio Code* nude se mogućnosti stvaranja novog projekta, učitavanje projekta iz drugih izvora ili već gotovih. Kako bi se stvorila aplikacija odabiremo **New File**. (slika 5.1.).



Slika 4.1: Stvaranje novog projekta

Projekt se sastoji od više glavnih dijelova od kojih svaki ima veliki faktor u kreiranju aplikacije. Struktura je zamišljena tako da se u node modules folderu nalaze svi paketi i

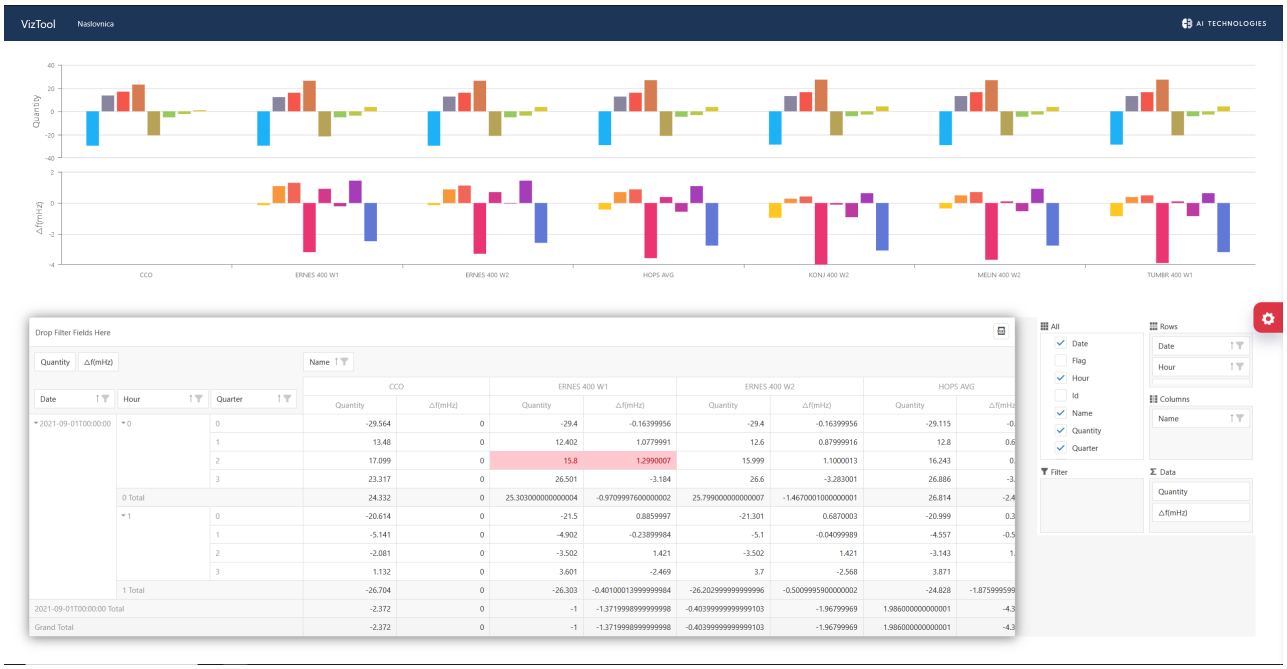
biblioteke potrebni za pokretanje aplikacije i povezivanje s gotovim komponentama skinutih s Interneta. U public folderu pronalazimo html file koji sadrži čitavu našu aplikaciju zajedno sa skriptama preuzetih s Interneta. Ovo su zapravo dva foldera u kojima programer ne djeluje aktivno, već ih jednostavno ažurira kad je potrebno instalirati neku novu komponentu. Ako pratimo strukturu projekta, vidjet ćemo kako prvo na red dolazi src folder. Ovdje se nalaze svi folderi koje programer aktivno mijenja kako bi svoju aplikaciju učinio što boljom i preglednijom. Najvažniji djelovi tog foldera su i "assets" gdje se pronalaze sve slike, "components" sa svim komponentama, odnosno grafovi i tablice, "router" za inicijaliziranje svih stranica(pošto je riječ o "Single Page Application" potrebna je samo jedna ruta), "style" sa stilovima, "views" sa našom stranicom.. Sve su ovo bitne komponente bez kojih aplikacija ne bi bila proizvod kakav je na kraju. Još jedna bitna komponenta je i "main.js" file u kojem se definira *VueJS* framework zajedno sa stilovima koje nudi *Bootstrap* i *DevExtreme*. Osim src foldera na kraju se nalazi i package-lock-json i package.json koji u sebi sadrže imena i verzije svih komponenti potrebnih za pokretanje i izvođenje naših aplikacija.



Slika 4.2: Struktura projekta aplikacije Viztool

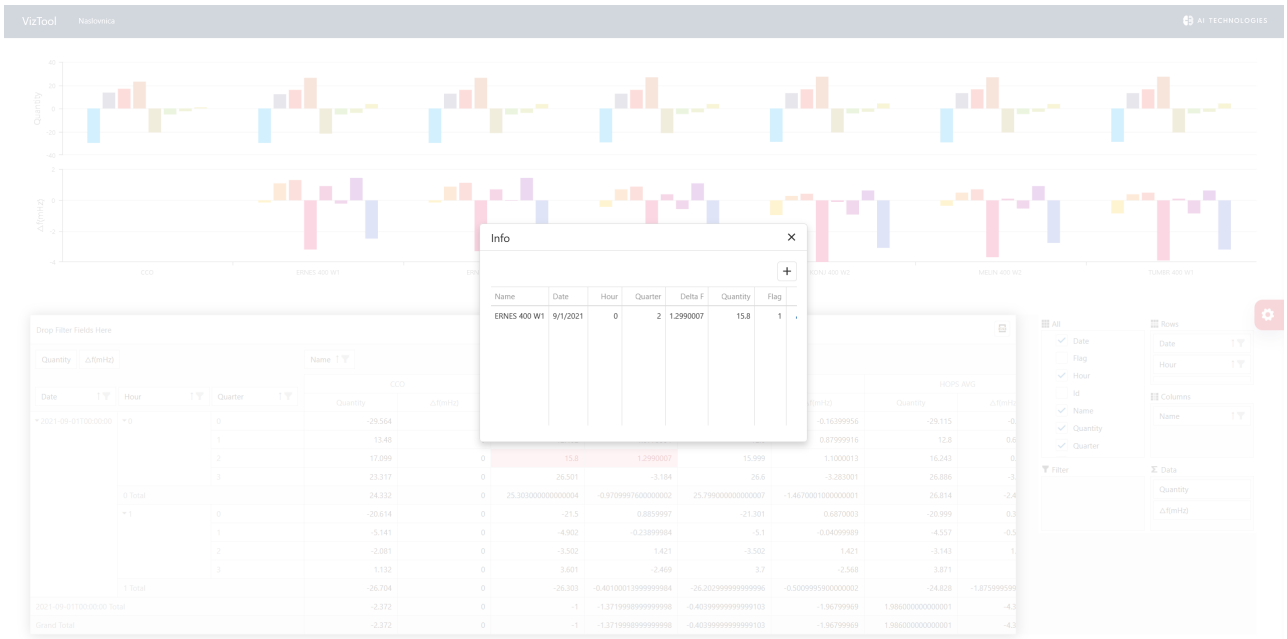
4.2. Funkcionalnosti aplikacije

Nakon pokretanja sustava, korisnik je preusmjeren na početnu stranicu koja nudi sve mogućnosti koje su već obrađene. Dakle korisnik odabire podatke koje želi prikazati u svojoj "Pivot tablici", raspoređuje ih po osi, te prikazuje grafički. Najvažnije karakteristike je zapravo prikazana s desne strane "Pivot tablice" i predstavlja postavke tablice. Ovdje korisnik može odabrati u koju kolonu ili redak ide koji podatak, koji podatak želi filtrirati ili prikazati u ovisnosti o drugim.



Slika 4.3: Welcome page

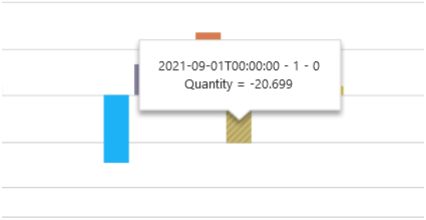
Također jedna od važnih karakteristika je editiranje podataka. Nakon što korisnik pritisne na neki od podatka prikazuje mu se Popup sa svim vrijednostima vezanim za taj podatak. U tom trenutku korisnik može promijeniti bilo koju od tih vrijednosti i one mu se automatski ažuriraju u tablici.



Slika 4.4: Edit podataka

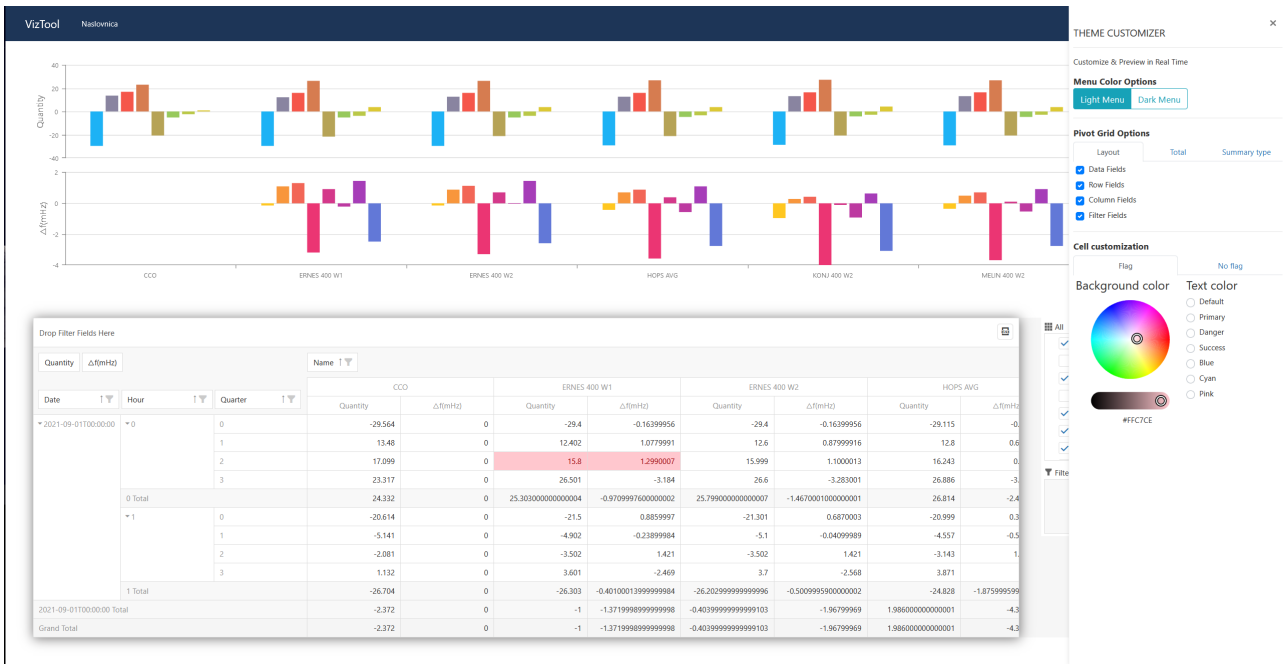
Nakon što mu se grafički prikažu podaci, korisnik hoveranjem na bilo koji stupac može

dobiti dodatne informacije o vrijednostima za taj određeni podatak.



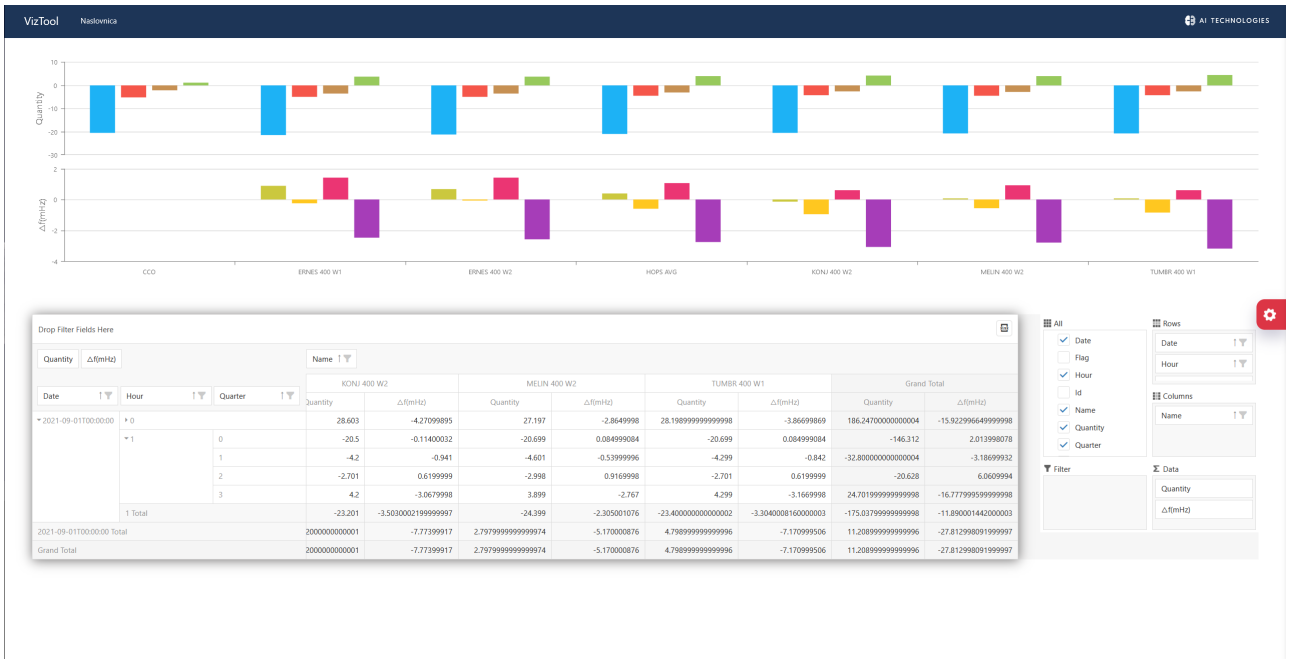
Slika 4.5: Info on hover

S desne strane zaslona, nalazi se crveni button koji nakon što se pritisne otvara menu-bar sa svim stilskim i funkcionalnim karakteristikama "Pivot tablice". Korisnik ovdje može promijeniti vrijednosti koje mu se prikazuju po čelijama, ali i pozadinsku boju za svaku od čelija kao i sami tekst.



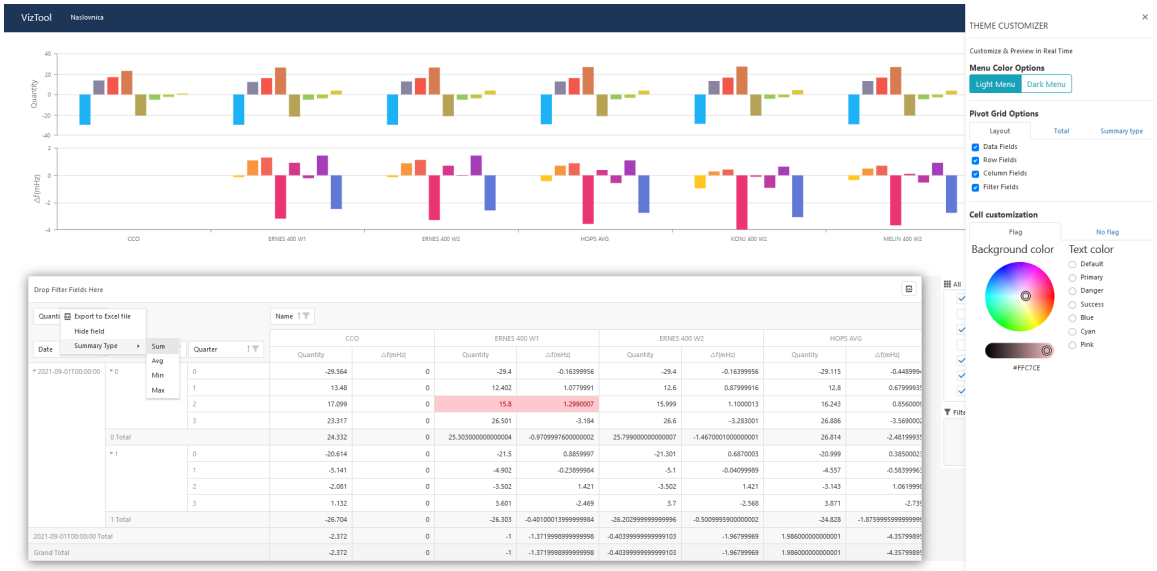
Slika 4.6: Stilski dio stranice

Još jedna od karakteristika stranice je drag and drop kod "Pivot tablice". Ako korisnik ne želi upravljati s postavkama sa strane, može određenu varijablu pritisnuti s mišem i postaviti gdje želi. Nakon toga će mu se automatski ažurirati podaci i svi grafovi. Također, ako korisnik pritisne neku varijablu u tablici(poput sata) svi podaci vezani za quarter će se sakriti i bit će prikazana njihova suma untar sata.



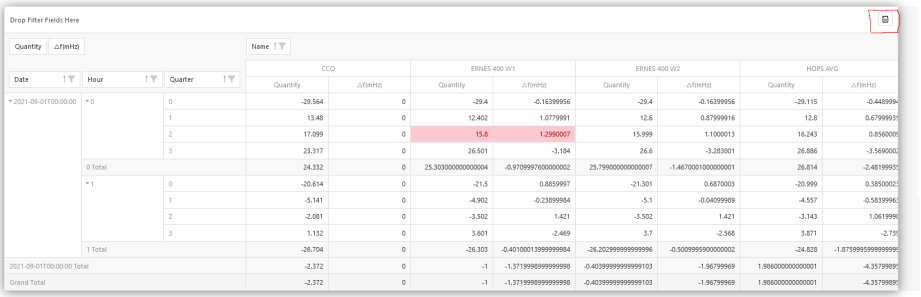
Slika 4.7: Uvlačanje varijable

Svaka od varijabli koja je ovisna o drugima ima opciju Summary typsea koja određuje na koji način će podaci biti prikazani nakon uvlačenja. Kao defaultna vrijednost postavljena je suma, ali također može biti prikazana minimalna, maksimalna ili prosječna.



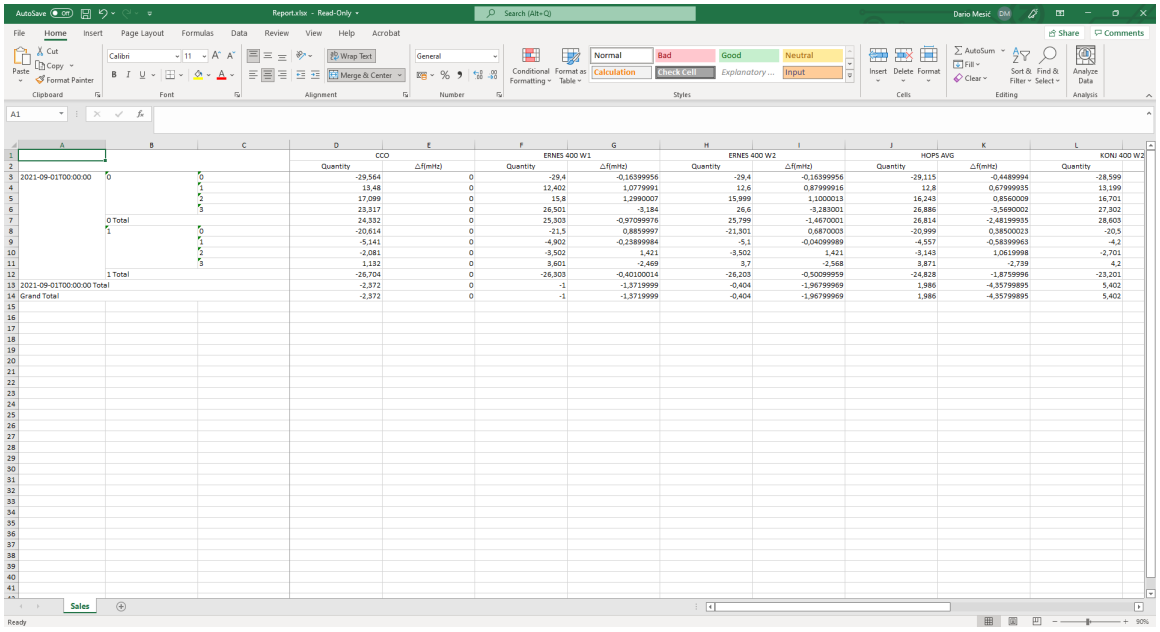
Slika 4.8: Summary varijable

Ako korisnik na kraju odluči da želi spremiti svoju tablicu i prikazati ju u Excelu, može to vrlo jednostavno napraviti pritiskom na button uz tablicu.



Slika 4.9: Excel export

Nakon spremanja, korisnik će imati uvid u tablicu u *Excelu* na isti način kao što je imao u aplikaciji.



Slika 4.10: Excel view

4.3. Zahtjevi i karakteristike sustava

Web aplikacija dostupna je za rad na svim Internet pretraživačima. Aplikacija je također testirana na različitim zaslonima, tako da je responzivna i za veće i za manje ekrane. Trenutno se ne bi mogla pokretati na mobitelima bez prilagođavanja veličine zaslona, ali to je nešto što bi se za budućnost moglo popraviti.

4.4. Moguća poboljšanja

Stvaranje aplikacije je proces koji nikad ne prestaje, ali u jednom trenutku trebamo odrediti minimum koji bi bio idealan za početak korištenja proizvoda. U ovom poglavlju navest će se neke ideje koje bi se mogle nadodati unutar aplikacije kako bi korisnik imao što bolji doživljaj aplikacije te kako bi mu aplikacija bila što korisnija u svakodnevnome životu.

4.4.1. Nadogradnja baze podataka

Trenutno se u bazi podataka nalazi samo jedan dataset s korisničkim podacima. Cilj je proširiti da korisnik može birati između različitih datasetova u aplikaciji. Također, bilo bi korisno kada bi korisnik mogao direktno komunicirati s bazom i kada bi sve promjene koje bi napravio u aplikaciji bile spremljene u bazi. *DevExtreme* nudi i ostale komponente koje bi se mogle ukomponirati u aplikaciju, poput različitih lista, grafova koje bi korisniku pružale još bolji uvid u svoje podatke.

4.4.2. Korisničko iskustvo

Ako je web aplikacija jedini način korisničke interakcije sa sustavom, ona mora biti vizualno atraktivna i pristupačna, što je najbolje prepustiti profesionalnom dizajneru prije puštanja aplikacije u uporabu. Kako bi se povećao potencijalni broj korisnika, osim web aplikacije moglo bi se stvoriti i mobilnu aplikaciju koja će na sličan način, pa možda i bolji, prikazati sve što i web aplikacija. S mobilnom aplikacijom, određene funkcionalnosti bi bile još lakše za uporabu poput drag and dropa, ili hovera na određeni stupac u grafu.

Također, mogla bi se stvoriti i "mala aplikacija" (engl. widget) koji će biti prikazan korisniku na početnoj stranici mobitela, a koji bi mogao sadržavati i graf s podacima.

4.4.3. Proširenje sustava

Kako aplikacija raste, tako je nužno da se i sustav mijenja i proširuje. Najvažnije proširenje je već definirano gore, a sadrži proširenje baze s novim datasetovima i ukomponiranje novih komponenti koje *DevExtreme* nudi. Jednog dana, ova aplikacija bi mogla postati i gotovi proizvod na Internetu. Za tako nešto bilo bi potrebno produkcijsku okolinu promijeniti, ali za to je potrebno više resursa. Također, trenutno se oslanjamo na komponentu *DevExtreme*,

a kada bi se proizvele vlastite komponente za grafove, proizvod bi postao još bolji i lakše bi došlo do više korisnika.

5. Zaključak

Zadatak za seminarski rad bio je izrada programske potpore za web aplikaciju koja omogućuje prikaz podataka u "Pivot tablici". Ona bi omogućila korisniku uvid u svoje važne podatke i pritom bi ih prikazala grafički u obliku bar plotova. Projekt je izrađen koristeći *Javascript* "frontend" frameworka *VueJS* koji je podatke uzimao sa servera smještenog u *PostgreSQL*, a poveznica između klijentske i serverske strane je činio API server.

Aplikacija ima veliku korisnost za svakog korisnika koji gomilu podataka želi prikazati na jednostavniji i ljepši način ili mu je potrebna za editiranje istih. Također, aplikacija nudi export komponente u *Excel* koji onda nudi neke druge mogućnosti.

Zaključak aplikacije je kako su ovakve stvari izuzetno korisne jer ih nema toliko na Internetu, pogotovo u besplatnoj verziji, tako da ne bi bilo loše razvijati ovakav rad dalje ili možda ga prikazati kao nekakvu mobilnu aplikaciju. Zadatak je podijeljen u dva ključna dijela. Prvi dio bio je sami izgled aplikacije, a drugi dio bio je povezivanje s bazom podataka

Najveći tehnički izazov bio je implementacija podataka iz baze podataka prema mojoj aplikaciji te prikaz u različitim grafovima i tablicama, no nakon određenog vremena i taj problem je bio svladan.

Ovaj seminarski rad zahtijevao je usvajanje tehničkog znanja pisanja *Javascript-a*, korištenje komponenti *Bootstrap* i *DevExtreme*, parsiranje kroz *JSON*, te služenje s različitim Aplikacijsko programskim sučeljima (API) te pisanje stručnoga rada. Najveća prepreka pri izradi aplikacije bila je učenje tehnologija koje komponenta *DevExtreme* nudi.

6. Literatura

[1] Json. 2022. URL https://en.wikipedia.org/wiki/JSON.

[2] Postgresql. 2022. URL https://en.wikipedia.org/wiki/PostgreSQL.

[3] Visual studio code. 2022. URL https://en.wikipedia.org/wiki/Visual_Studio_Code.

[4] Vue.js. 2022. URL https://en.wikipedia.org/wiki/Vue.js.

[5] IBM. Rest api. 2018. URL https://www.ibm.com/docs/hr/mam/7.6.1?topic=apis-rest-api.

[6] Raivat Shah. Introduction to rest apis. 2020. URL https://towardsdatascience.com/introduction-to-rest-apis-90b5d9676004.

Popis slika