

# Web aplikacija za vođenje restorana

---

Mihić, Dejan

Undergraduate thesis / Završni rad

2024

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://um.nsk.hr/um:nbn:hr:200:976558>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom](#).

Download date / Datum preuzimanja: **2025-02-23**

Repository / Repozitorij:

[Faculty of Electrical Engineering, Computer Science and Information Technology Osijek](#)



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

**Stručni prijediplomski studiji Računarstvo**

**WEB APLIKACIJA ZA VOĐENJE RESTORANA**

**Završni rad**

**Dejan Mihić**

**Osijek, 2024.**

**FERIT**FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA  
I INFORMATIJSKIH TEHNOLOGIJA OSIJEK**Obrazac Z1S: Obrazac za ocjenu završnog rada na stručnom prijediplomskom studiju****Ocjena završnog rada na stručnom prijediplomskom studiju**

<b>Ime i prezime pristupnika:</b>	Dejan Mihić
<b>Studij, smjer:</b>	Stručni prijediplomski studij Računarstvo
<b>Mat. br. pristupnika, god.</b>	AR 4795, 27.07.2020.
<b>JMBAG:</b>	0165039622
<b>Mentor:</b>	Marina Peko, dipl. ing. el.
<b>Sumentor:</b>	
<b>Sumentor iz tvrtke:</b>	
<b>Predsjednik Povjerenstva:</b>	Robert Šojo, univ. mag. ing. comp.
<b>Član Povjerenstva 1:</b>	Marina Peko, dipl. ing. el.
<b>Član Povjerenstva 2:</b>	mr. sc. Željko Štanfel
<b>Naslov završnog rada:</b>	Web aplikacija za vođenje restorana
<b>Znanstvena grana završnog rada:</b>	<b>Programsko inženjerstvo (zn. polje računarstvo)</b>
<b>Zadatak završnog rada:</b>	Izraditi web aplikaciju za vođenje restorana. Aplikacija se sastoji od korisničkog dijela - slanje narudžbe, rezervacije i dodatne opcije te administrativnog dijela za obradu narudžbi (radnici objekta) i SuperUser dijela (poslodavci). Aplikacija uključuje vizualizaciju prostora objekta, te je vidljivo koji su to stolovi zauzeti, a koji slobodni u određenom periodu dana. Tema rezervirana za: Dejan Mihić
<b>Datum ocjene pismenog dijela završnog rada od strane mentora:</b>	22.09.2024.
<b>Ocjena pismenog dijela završnog rada od strane mentora:</b>	Izvrstan (5)
<b>Datum obrane završnog rada:</b>	26.09.2024.
<b>Ocjena usmenog dijela završnog rada (obrane):</b>	Izvrstan (5)
<b>Ukupna ocjena završnog rada:</b>	Izvrstan (5)
<b>Datum potvrde mentora o predaji konačne verzije završnog rada čime je pristupnik završio stručni prijediplomski studij:</b>	26.09.2024.

**FERIT**FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA  
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK****IZJAVA O IZVORNOSTI RADA**

Osijek, 26.09.2024.

**Ime i prezime Pristupnika:**

Dejan Mihić

**Studij:**

Stručni prijediplomski studij Računarstvo

**Mat. br. Pristupnika, godina upisa:**

AR 4795, 27.07.2020.

**Turnitin podudaranje [%]:**

4

Ovom izjavom izjavljujem da je rad pod nazivom: **Web aplikacija za vođenje restorana**

izrađen pod vodstvom mentora Marina Peko, dipl. ing. el.

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 pristupnika:

# SADRŽAJ

<b>1. UVOD</b> .....	<b>1</b>
1.1. Zadatak Završnog rada .....	2
<b>2. PREGLED POSTOJEĆIH APLIKACIJA</b> .....	<b>3</b>
2.1. Zoho inventory .....	3
2.2. Unleashed .....	3
2.3. Shopify POS .....	4
2.4. Revel Systems .....	5
<b>3. KORIŠTENE TEHNOLOGIJE ZA IZRADU APLIKACIJE</b> .....	<b>6</b>
3.1. Node.js .....	6
3.2. Vue.js .....	6
3.3. HTML .....	7
3.4. CSS .....	8
3.5. Tailwind CSS .....	8
3.6. TypeScript .....	9
3.7. Firebase .....	9
<b>4. PROGRAMSKO RJEŠENJE</b> .....	<b>11</b>
4.1. Struktura web aplikacije .....	11
4.2. Korisnički definirani tipovi podataka aplikacije .....	13
4.3. Prijava, registracija i pristup .....	15
4.4. Pretraživanje sirovina i proizvoda .....	17
4.5. Pohrana, ažuriranje, brisanje i dohvaćanje podataka iz baze podataka .....	18
<b>5. VIZUALNA REPREZENTACIJA APLIKACIJE</b> .....	<b>22</b>
5.1. Prijava i registracija .....	22
5.2. Korisnički račun .....	23
5.3. Skladište .....	24
5.4. Blagajna .....	32

<b>5.5. Upravljanje .....</b>	<b>34</b>
<b>5.6. Narudžbe i računi.....</b>	<b>38</b>
<b>6. ZAKLJUČAK.....</b>	<b>40</b>
<b>LITERATURA .....</b>	<b>41</b>
<b>SAŽETAK.....</b>	<b>42</b>
<b>ABSTRACT .....</b>	<b>43</b>
<b>ŽIVOTOPIS.....</b>	<b>44</b>

## 1. UVOD

Čest je slučaj da u restoranima i kafićima korisnici istih čekaju na red kako bi naručili hranu i piće te se ponekad događa kako pojedini korisnici ostanu neprimijećeni ili čekaju duže nego bi trebali što uzrokuje njihovo nezadovoljstvo uslugom. Najčešće restorani za vođenje i upravljanje koriste alate u obliku desktop softverskih rješenja, a za krajnje korisnike nude web portale za prikaz ponude i drugo. Ova aplikacija nastoji riješiti takav problem tako što nudi jedinstveno sučelje za pristup aplikaciji kako djelatnicima restorana tako i samim korisnicima, ali i cjelokupni postupak vođenja jednog ugostiteljskog objekta.

Web aplikacija omogućava prijavu i registraciju korisnika kako bi stekli pravo pristupa dodatnim mogućnostima aplikacije. Neregistrirani korisnici mogu pristupiti prijavi i registraciji te početnoj stranici, ali ne mogu obavljati nikakve operacije, dok su za registrirane korisnike definirane četiri razine pristupa. Svi korisnici koji samostalno obave registraciju imaju najnižu razinu pristupa te mogu vršiti narudžbe proizvoda. Svaki korisnik s višom razinom pristupa djelatnik je restorana. Svaki korisnik i djelatnik u bilo kojem trenutku može vidjeti topografski prikaz objekta i vidjeti zauzetost pojedinih stolova u ovisnosti o vremenu izdavanja račun za određeni stol te svaki korisnik ima uvid u vlastite narudžbe i račune. Djelatnici, ovisno o pravima pristupa, imaju različite mogućnosti pa tako djelatnici najniže razine mogu pristupiti blagajni za izdavanje računa, listi izdanih računa, storniranju računa i listi za praćenje narudžbi korisnika. Djelatnici srednje razine, odnosno menadžeri, uz prethodno navedeno, mogu pristupiti skladištu koje omogućava uvoz robe, kreiranje različitih sirovina, kreiranje proizvoda, kreiranje različitih kategorija kojima ti proizvodi pripadaju, imati uvid u zalihe kojim se raspolaže te ažuriranje i brisanje istih. Najveća razina pristupa je administrator koja uz sve prethodno navedeno nudi mogućnost upravljanja svim elementima aplikacije pa tako i dodavanjem novih djelatnika, njihovo brisanje, ažuriranje te također može upravljati stolovima, njihovim oblikom, imenom i pozicijom. Samo administrator može promijeniti ulogu korisnika i kreirati djelatnika. Postupak korištenja aplikacije detaljno je objašnjen u nastavku ovoga rada.

U drugome poglavlju prikazana su već neka postojeća rješenja te su pojašnjene razlike između takvih rješenja i ovoga rada. Treće poglavlje prikazuje koje su tehnologije korištene za izradu ovoga rada i njihov kratak opis. Četvrto poglavlje nudi pojedine funkcionalnosti aplikacije prikazane kroz neka od programskih rješenja korištenih za ispravan rad aplikacije te peto poglavlje daje detaljan opis rada i prikaz sučelja pojedinih dijelova aplikacije.

## **1.1. Zadatak Završnog rada**

Cilj ove aplikacijom omogućiti korisnikovo uključivanje u sam rad restorana, na način da može vršiti i pratiti narudžbe proizvoda, pratiti zauzetost restorana po stolovima. Ponuditi jedinstveno sučelje za rad restorana, odnosno ugostiteljskog objekta tako što svi korisnici i djelatnici pristupaju istom sučelju, ali s različitim pravima pristupa i mogućnostima kojim aplikacija nudi za danu razinu pristupa.

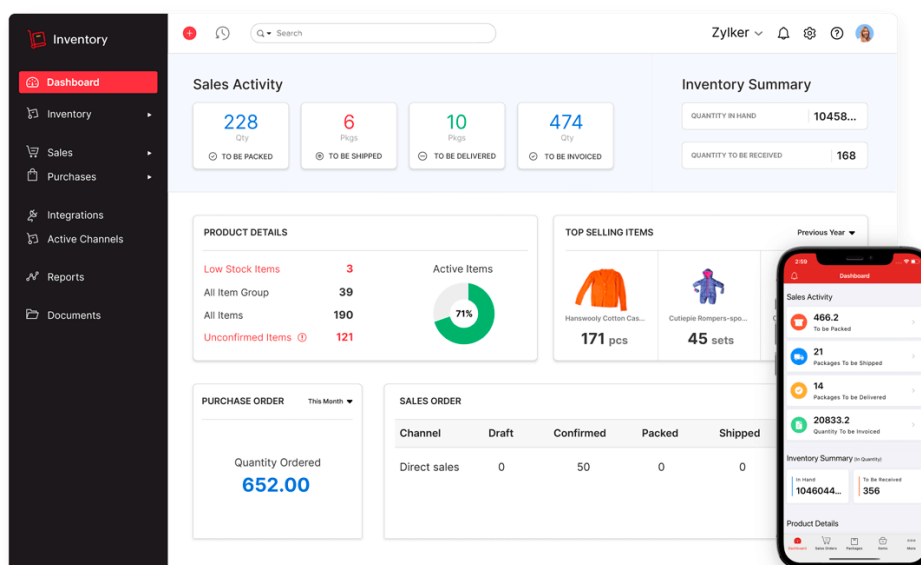


## 2. PREGLED POSTOJEĆIH APLIKACIJA

U ovome poglavlju prikazana su neka od postojećih rješenja, njihov generalni dizajn, razlike u odnosu na ovaj rad te koja je njihova povezanost.

### 2.1. Zoho inventory

*Zoho Inventory* je softversko rješenje temeljeno na oblaku (engl. *cloud based*) za upravljanje zalihama koje pomaže tvrtkama u učinkovitoj organizaciji njihovih skladišnih operacija. Omogućava praćenje zaliha u stvarnom vremenu, upravljanje narudžbama, isporukama i vraćanjem proizvoda. *Zoho Inventory* također podržava izdavanje računa, kreiranje narudžbenica i upravljanje dobavljačima. Integrira se s raznim platformama poput *Shopify* i *Amazon* te s drugim poslovnim alatima u Zoho ekosustavu. Ovaj softver nudi analitičke alate za izvješćivanje i prognoziranje, čime pomaže tvrtkama u donošenju informiranih poslovnih odluka (Slika 2.1.) [1]. Ovaj rad nudi slične mogućnosti kao što su praćenje stanja zaliha u stvarnome vremenu, ali specifično samo za restorane i srodne ugostiteljske djelatnosti te ne nudi analitiku.

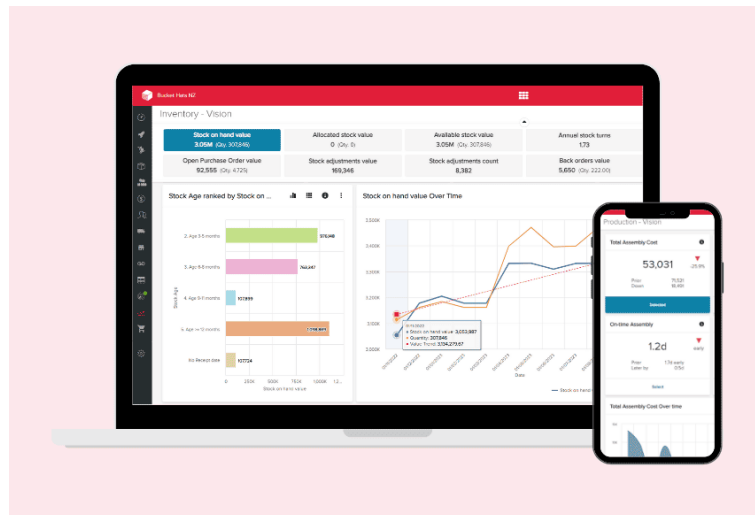


Sl. 2.1. Prikaz sučelja zoho inventory aplikacije

### 2.2. Unleashed

*Unleashed* je softversko rješenje za upravljanje zalihama koje omogućava tvrtkama praćenje zaliha u stvarnom vremenu, upravljanje skladištem i optimizaciju proizvodnih procesa. Osigurava učinkovito praćenje narudžbi, upravljanje dobavljačima i praćenje isporuka. *Unleashed* također omogućava izdavanje računa, kreiranje narudžbenica i analizu podataka kroz detaljna izvješća i

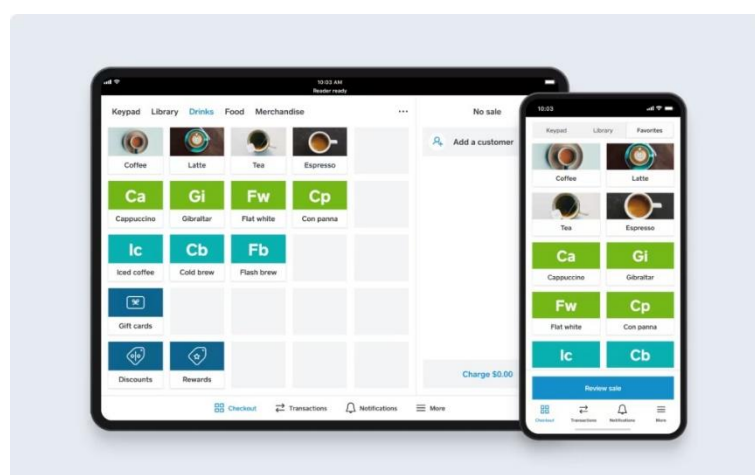
prognoze. Integrira se s raznim platformama poput *Shopify* i *Amazon* te s poslovnim alatima kao što su *Xero* i *QuickBooks*, čime pomaže tvrtkama u poboljšanju operativne učinkovitosti i donošenju informiranih poslovnih odluka (Slika 2.2.) [2]. *Unleashed* nudi slične mogućnosti vođenja i upravljanja zalihama kao i izdavanje računa kao i ovaj rad, ali u širem spektru djelatnosti, kao i neke napredne značajke.



Sl. 2.2. Prikaz sučelja aplikacije unleashed

### 2.3. Shopify POS

*Shopify POS* (engl. *point of sale*) je sustav koji omogućava prodaju proizvoda u fizičkim trgovinama te upravljanje prodajom i zalihama. Integrira se s *Shopify e-commerce* platformom, čime omogućava sinkronizaciju podataka o zalihama i prodaji između online i fizičkih trgovina.



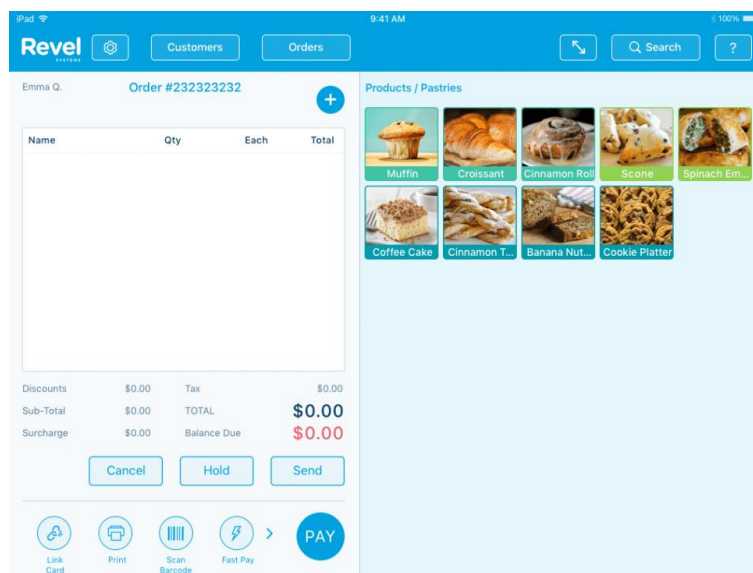
Sl. 2.3. Prikaz sučelja Shopify POS aplikacije

*Shopify POS* nudi alate za izdavanje računa, obradu plaćanja, upravljanje zalihama, praćenje

prodaje u stvarnom vremenu i analitiku. Sustav također podržava upravljanje korisnicima i lojalnost te se može integrirati s raznim aplikacijama za proširenje funkcionalnosti (Slika 2.3.) [3]. *Shopify POS* u odnosu na ovaj rad nudi sličan pristup kod prikaza pojedinih proizvoda te njihovo upravljanje, ali nudi i napredne mogućnosti oko plaćanja što nije dio ovoga rada.

## 2.4. Revel Systems

*Revel Systems* je POS sustav dizajniran za restorane i maloprodaju. Omogućava upravljanje prodajom, zalihama, narudžbama i izdavanjem računa. Sustav nudi alate za analitiku i izvješćivanje u stvarnom vremenu, što pomaže u donošenju informiranih poslovnih odluka. *Revel Systems* također podržava integraciju s različitim poslovnim alatima i aplikacijama, omogućava upravljanje zaposlenicima i korisnicima te nudi funkcionalnosti za online narudžbe i dostavu (Slika 2.4.) [4]. *Revel System* najbližiji ovome radu, ali nudi i mnoge dodatne mogućnosti kao što je analitika, iako ne nudi grafički prikaz objekta te nije prilagođen za ugostiteljske djelatnosti.



Sl. 2.4. Prikaz sučelja *Revel Systems* aplikacije

### 3. KORIŠTENE TEHNOLOGIJE ZA IZRADU APLIKACIJE

Za izradu skalabilnih, dobro optimiziranih i responzivnih web aplikacija potrebno je koristiti različite tehnologije koje to omogućavaju na efikasan način. U ovom poglavlju ukratko su navedene i objašnjene tehnologije potrebne za razvoj ove aplikacije.

#### 3.1. Node.js

Node.js je dizajniran kao asinkroni događajem vođeni JavaScript za izgradnju skalabilnih mrežnih aplikacija [5]. Node.js dolazi s NPM-om (engl. *Node Package Manager*), najvećim ekosustavom otvorenog koda za upravljanje paketima i bibliotekama.

U kontekstu Vue projekata, Node.js je ključan za postavljanje i upravljanje razvojnim okruženjem. Instalacija Vue CLI-a (engl. *Command Line Interface*), koji olakšava kreiranje Vue projekata, vrši se putem `node.js`-a. Node.js omogućava pokretanje lokalnog razvojnog poslužitelja koji automatski reflektira promjene u kodu, poboljšavajući produktivnost. Alati za bundling i optimizaciju, poput Webpacka i Vitea, rade na Node.js platformi, omogućavajući modularizaciju i minifikaciju koda.

#### 3.2. Vue.js

Vue.js je progresivni JavaScript *frontend* programski okvir (engl. *Framework*) za izgradnju korisničkih sučelja i jednostraničnih aplikacija (engl. *Single-page application*). Vue.js je neovisan projekt vođen zajednicom. Stvorio ga je Evan You 2014. godine kao osobni sporedni projekt. Danas Vue.js aktivno održava tim sastavljen od stalnih i volonterskih članova iz cijelog svijeta, a Evan je voditelj projekta [6]. Njegova reaktivna jezgra omogućava deklarativno vezivanje podataka, što olakšava upravljanje stanjima unutar aplikacije. Donosi brojne optimizacije i poboljšanja u odnosu na prethodnu verziju, uključujući bolju podršku za TypeScript, manju veličinu paketa i brže performanse. Jedna od ključnih novosti je uvođenje *composition API*-a (engl. *Application Programming Interface*), koji omogućava bolju organizaciju i ponovnu upotrebu koda, čineći složene aplikacije lakšima za održavanje. Podržava izgradnju velikih aplikacija kroz modularnu arhitekturu i jednostavnu integraciju s drugim alatima i bibliotekama. Podrška za SFC (engl. *Single File Component*) omogućava pisanje HTML-a (engl. *HyperText Markup Language*), CSS-a i JavaScript-a unutar jedne datoteke, što olakšava razvoj i održavanje komponenata (Slika 3.1.).

```
App.vue

<template>
  <ul>
    <ToDoListItem />
  </ul>
</template>

<script setup lang="ts">
import ToDoListItem from './components/ToDoListItem.vue'
</script>

<style>
ul {
  list-style: none;
  padding: 0;
  width: 300px;
}
</style>
```

Sl. 3.1. Prikaz izgleda jednostavne *vue* datoteke

Vue.js CLI olakšava postavljanje i upravljanje Vue.js projektima, uključujući konfiguraciju (engl. *configuration*), izgradnju (engl. *build*) i raspoređivanje (engl. *deployment*). Prilagodljiv je i može se koristiti za izgradnju malih dijelova stranica ili cijelih aplikacija, što ga čini svestranim alatom za razvoj.

### 3.3. HTML

HTML (engl. *HyperText Markup Language*) je najosnovniji građevni blok weba. Definira značenje i strukturu web sadržaja [7]. Koristi oznake (engl. *tags*) za definiranje elemenata kao što su naslovi, odlomci, linkovi, slike i drugi sadržaji (Slika 3.2.). Web preglednici interpretiraju HTML kod i prikazuju ga kao vizualno oblikovane stranice. HTML se često koristi zajedno s CSS-om i JavaScriptom za stvaranje interaktivnih i stiliziranih web stranica. Razvijen je kako bi omogućio jednostavno povezivanje i navigaciju između različitih web stranica putem hiperteksta.

```
<!DOCTYPE html>
<html>

  <head>
    <title>My First Webpage</title>
  </head>

  <body>
    <h1>
      My First Webpage
    </h1>
    <p>This is a paragraph...</p>
  </body>

</html>
```

Sl. 3.2. Primjer jednostavne HTML stranice

### 3.4. CSS

CSS (engl. *Cascading Style Sheets*) je programski jezik koji web stranici daje njen izgled i raspored. Zajedno s HTML-om, CSS je temelj za web dizajn. Bez njega, web stranice bi bile običan tekst na bijelim pozadinama. Omogućava definiranje vizualnog izgleda HTML elemenata, čime se postiže atraktivan i responzivan dizajn korisničkih sučelja.

```
.chair-45 {
  width: 13px;
  height: 13px;
  background-color: #6b7280;
  position: absolute;
  transform: rotate(45deg);
}
```

Sl. 3.3. Primjer css-a s klasnim operatorom

Kroz selektore i pravila, CSS primjenjuje stilove na HTML elemente, što omogućava fleksibilne rasporede i prilagodljivost različitim uređajima (Slika 3.3.). Napredne tehnike poput *flexboxa* i *grida* omogućuju stvaranje složenih rasporeda. Također, CSS omogućava animacije i prijelaze, dodajući dinamičke efekte bez potrebe za korištenjem JavaScripta.

### 3.5. Tailwind CSS

Tailwind CSS je CSS razvojni okvir koji pruža unaprijed definirane klase za brzo stiliziranje elemenata.

```
<template>
  <div class="flex flex-row justify-around">
    <div class="w-3/6 pl-3">
      <LabelAndDataTag :label="firstLabel" :data="firstData"></LabelAndDataTag>
    </div>
    <div class="flex flex-col justify-center">
      <div class="h-10 w-px bg-black"></div>
    </div>
    <div class="w-3/6 pl-3">
      <LabelAndDataTag :label="secondLabel" :data="secondData"></LabelAndDataTag>
    </div>
  </div>
</template>
```

Sl. 3.5. Primjer korištenja Tailwind css-a

Umjesto pisanja prilagođenog CSS-a, Tailwind omogućava korištenje malih klasa koje se kombiniraju za izradu složenih dizajna. Ovaj pristup ubrzava razvoj jer eliminira potrebu za pisanjem prilagođenih stilova, omogućavajući brže iteracije i promjene dizajna (Slika 3.5.).

Tailwind CSS dolazi s konfiguracijskom datotekom koja omogućava prilagodbu zadanih stilova i stvaranje vlastitih klasa. Tailwind CSS radi tako da skenira sve HTML datoteke, JavaScript komponente i ostale predloške za imena klasa, generira odgovarajuće stilove te ih zatim zapisuje u statičku CSS datoteku [9]. Tailwind CSS podržava responzivni dizajn kroz skup klasa za različite veličine ekrana, olakšavajući izradu mobilno-prilagođenih sučelja. Tailwind-ova JIT (engl. *Just In Time*) kompilacija generira samo potrebne stilove na temelju korištenih klasa, što značajno smanjuje veličinu CSS datoteke. Kompatibilan je s popularnim JavaScript *frontend* okvirima kao što su React, Vue i Angular, olakšavajući integraciju u postojeće projekte.

### 3.6. TypeScript

TypeScript je snažno tipizirani objektno-orientirani programski jezik otvorenog koda koji se nadograđuje na JavaScript. Razvija ga i podržava Microsoft, a predstavlja snažan nadskup JavaScripta koji uključuje sve njegove segmente. Svaki JavaScript kod je važeći TypeScript kod [10]. Omogućava programerima definiranje tipova podataka za varijable, funkcije i objekte, što pomaže u otkrivanju grešaka prije izvršavanja koda (Slika 3.6.). TypeScript je interoperabilan s postojećim JavaScript kodom, omogućujući postepenu migraciju i integraciju u projekte. Kompilira se u čisti JavaScript, čineći ga kompatibilnim sa svim preglednicima i JavaScript okruženjima

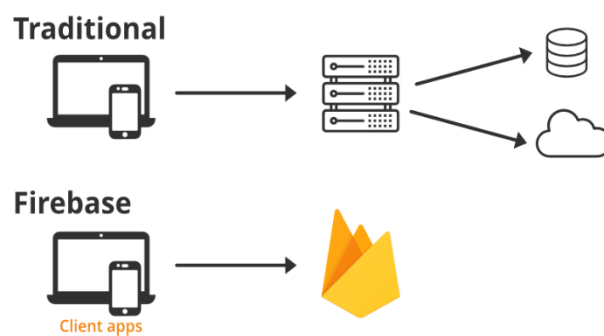
```
export function isValidCode(input: string): boolean {  
  const regex = /^d+$/;  
  return regex.test(input);  
}
```

Sl. 3.6. Primjer jednostavne TypeScript funkcije

### 3.7. Firebase

Firebase je usluga BaaS (engl. *Backend as a Service*) (Slika 3.7.) koja programerima pruža razne alate i usluge kako bi im pomogla u razvoju kvalitetnih aplikacija. Kategoriziran je kao NoSQL (engl. *Not only SQL*) baza podataka, koja pohranjuje podatke u dokumentima sličnim JSON-u (engl. *JavaScript Object Notation*) [11].

Pružna niz usluga kao što su baza podataka u realnom vremenu (engl. *real time*), autentifikacija korisnika, pohrana u oblaku (engl. *cloud storage*) i analitika (engl. *analytics*). Omogućava brzu i jednostavnu sinkronizaciju podataka između klijentskih aplikacija i servera, što olakšava razvoj aplikacija koje zahtijevaju funkcionalnosti u stvarnome vremenu. U usporedbi s SQL (engl. *Structured Query Language*) bazama podataka, koje koriste tablice i relacije, NoSQL baza omogućava lakše skaliranje i brže operacije čitanja i pisanja. Za razliku od SQL baza podataka koje koriste strukturirani jezik za upite i manipulaciju podacima. Firebase koristi jednostavnije metode za rad s podacima, ali može biti manje učinkovita za složene upite koji uključuju višestruke relacije.



Sl. 3.7. Prikaz komunikacije tradicionalne SQL i Firebase NoSQL baze podataka s klijentom



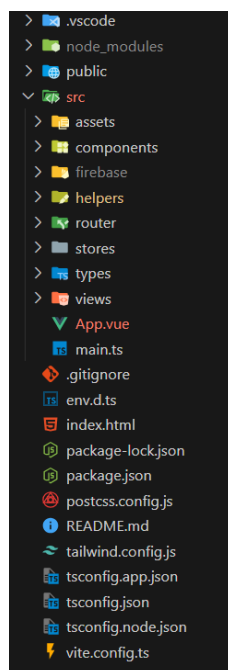
## 4. PROGRAMSKO RJEŠENJE

Ovo poglavlje prikazuje i opisuje glavne smjernice i rješenja kako kreirati projekt, kako je rad strukturiran i koje su to osnovne funkcionalnosti koje omogućavaju ispravan rad aplikacije.

### 4.1. Struktura web aplikacije

Sama struktura web aplikacije za upravljanje restoranima definirana je prilikom kreiranja samog projekta te je ona postepeno proširena dodavanjem novih mapa (engl. *folders*), ovisnosti i konfiguracijskih datoteka (Slika 4.1.).

Najvažnije datoteke iz strukture su:



Sl. 4.1. Prikaz strukture web aplikacije

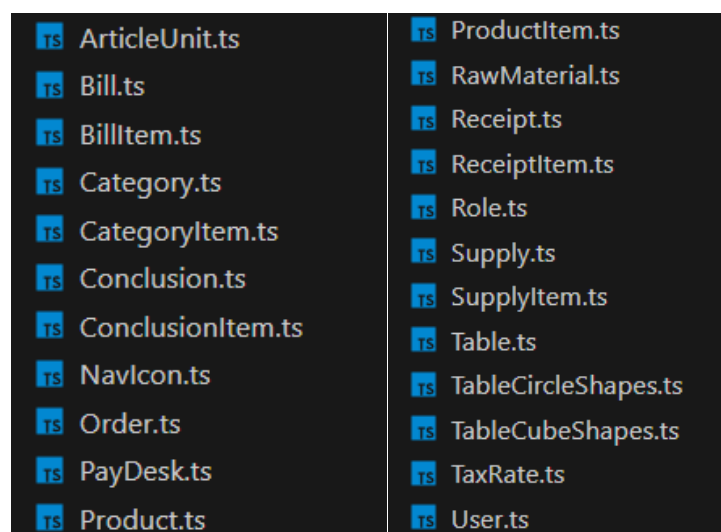
- node\_modules – direktoriji koji sadrži sve instalirane ovisnosti koje aplikacija koristi za svoj rad
- src – direktoriji koji sadrži sav programski kod aplikacije:
  - assets – direktoriji sadrži statičke datoteke kao što je osnovni css aplikacije i slike
  - components – direktoriji koji sadrži sve pojedinačne komponente aplikacije kao što su, forme, gumbi, navigacija i sl. te svaka komponenta sadrži pripadajuće HTML tagove, Tailwind CSS klase i TypeScript za funkcionalnost koju komponenta ima

- firebase – je direktoriji koji sadrži TypeScript kod za konfiguraciju Firebase-a kako bi se moglo pristupiti samoj bazi
  - helpers – direktoriji sadrži funkcije koje se koriste na više od jednog mjesta kako bi se izbjegao duplicirani kod
  - router – direktoriji koristi se za konfiguraciju i definiciju svih ruta kojima se može pristupiti putem web preglednika, odnosno omogućava navigaciju između različitih pogleda ili stranica.
  - stores - Unutar ovoga direktorija nalaze se različite Pinia pohrane (engl. *stores*) koje upravljaju globalnim stanjem aplikacije. Svako skladište pohranjuje podatke i logiku vezanu za određene dijelove aplikacije.
  - types – direktoriji u kojem se nalaze korisnički definirani tipovi podataka, pri čemu je svaki tip smješten u zasebnoj TypeScript datoteci. Ovi tipovi olakšavaju tipizaciju i provjeru tipova unutar aplikacije.
  - views – direktoriji u koje se nalaze Vue komponente koje predstavljaju različite stranice ili poglede u aplikaciji. Svaka komponenta u ovom direktoriju odgovara jednoj ruti i sadrži logiku i prikaz za određeni dio korisničkog sučelja.
  - App.vue – datoteka sadrži glavnu Vue komponentu koja definira osnovnu strukturu i izgled cijele aplikacije.
  - main.ts – Datoteka koja je ulazna točka Vue aplikacije koja inicijalizira i montira glavnu Vue instancu. U njoj se konfiguriraju globalne postavke, kao što su registracija *routera* i pohrana te se povezuje *App.vue* s *index.html*.
- index.html - je glavna HTML datoteka koja služi kao osnovna točka ulaska za Vue.js aplikaciju.
  - package-lock.json - sadrži precizne informacije o verzijama svih instaliranih npm paketa kako bi se osigurala dosljednost u izgradnji i razvoju Vue.js aplikacije.
  - package.json - služi kao konfiguracijska datoteka koja sadrži informacije o projektu, poput imena, verzije, skripti za izgradnju i pokretanje, kao i sve ovisnosti koje su potrebne za rad Vue aplikacije.

- `tailwind.config.js` - je datoteka konfiguracije u kojoj se definiraju prilagodbe i opcije za Tailwind CSS okvir kako bi se prilagodio stiliziranje i ponašanje CSS-a u Vue projektu.
- `tsconfig.json` - je konfiguracijska datoteka koja se koristi kako bi se definirala specifična podešavanja, kao što su postavke kompilacije, putanje modula i opcionalne prilagodbe za upravljanje TypeScript kodom.
- `Vite.config.ts` - je TypeScript datoteka konfiguracije koja se koristi u *Vite* projektnom okruženju za definiranje specifičnih postavki i prilagodbi za izgradnju i razvoj Vue.js aplikacije.

## 4.2. Korisnički definirani tipovi podataka aplikacije

U ovome projektu za svaki objekt kreiran je tip podataka kako bi se osigurala jasnija semantika i struktura podataka unutar koda, čime se olakšava održavanje i proširivost softvera. Svaki tip sadrži definiciju, u obliku sučelja, koje podatke sadrži te kojeg su ti podaci tipa. Na taj se način smanjuje mogućnost grešaka, jer je sve jasno definirano (Slika 4.2.). U nastavku je prikazana jedna od definicija korisnički definiranog tipa podatka (Slika 4.3.).



Sl. 4.2. Prikaz svih korisničkih tipova podataka aplikacije

- `ArticleUnit.ts` – je enumeracija ili *enum* klasa koja sadrži konstante, u ovom slučaju jedinice mjere, koje neki proizvod može imati.
- `Bill.ts` – je datoteka koja sadrži definiciju *Bill* tipa podatka, a koji sadrži sva svojstva potrebna za pohranu izdanog računa.

- *BillItem.ts* – datoteka koja sadrži definiciju tipa podatka *BillItem* koja sadrži sve podatke potrebne da se pohrani podatak o stavci računa kojeg se izdaje korisniku prilikom izdavanja računa.
- *Category.ts* – sadrži definiciju tipa podatka *Category* koji sadrži sve podatke potrebne da se kreira, pohrani i manipulira kategorijama koje u sebi sadrže liste proizvoda koje joj pripadaju kako bi korisnici aplikacije na jednostavan način razlikovali proizvode prema imenu kategorije u koju pripadaju kao npr. gazirani sokovi, negazirani sokovi i sl.
- *CategoryItem.ts* - definira *CategoryItem* tip podatka koji je jedan od podataka *Category* tipa podatka, a sadrži identifikator proizvoda koji se nalazi u nekoj kategoriji.
- *Conclusion.ts* – definira *Conclusion* tip podatka koji sadrži sva svojstva za pohranu zaključka blagajne.
- *ConclusionItem.ts* – definira *ConclusionItem* tip podatka koji sadrži sva svojstva za pohranu proizvoda unutar zaključka.
- *NavIcon.ts* – definira *NavIcon* tip podatka koji sadrži sva potrebna svojstva za prikaz i praćenje stanja navigacije za uređaje s manjim zaslonom.
- *Order.ts* – definira *Order* tip podatka koji sadrži sva svojstva potrebna za pohranu narudžbi.
- *PayDesk.ts* – definira *PayDesk* tip podatka koji sadrži sva svojstva potrebna za rad s blagajnom.
- *Product.ts* – definira *Product* tip podatka za Proizvode koji se nalaze u prodaji, definicija sadrži sve podatke potrebne kako bi se opisao jedan proizvod.
- *ProductItem.ts* – definira tipa podatka *ProductItem* koji se odnosi na pojedine sirovine koje jedan proizvod sadrži te definira sva svojstva potrebna za opis takvog podatka.
- *RawMaterial.ts* – definira tip podatka *RawMaterial* koji se odnosi na sirovine te definira sva svojstva za opis sirovine i njenu pohranu.
- *Receipt.ts* – definira tip podatka *Receipt* koji se odnosi na primku koja definira sav ulaz sirovina u aplikaciju te su definirana sva svojstva koje takav tip podatka treba imati za opis primke.
- *ReceiptItem.ts* – definira tip podatka *ReceiptItem* koji se odnosi na pojedinačnu sirovinu koja se dodaje u primku te definira sva svojstva potrebna za opis takvog podatka.
- *Role.ts* – je datoteka koja definira *Role enum* klasu u kojoj su definirane sve uloge koje jedan korisnik može imati.

- `Supply.ts` – definira tip podatka `Supply` koji se odnosi na zalihe svih sirovina te definira sva svojstva koja treba imati za opis stanja zalihe u aplikaciji.
- `SupplyItem.ts` – definira tip podatka `SupplyItem` koji se odnosi na sirovinu koja se nalazi u zalihama te definira sva svojstva za opis pojedinačne stavke zaliha.
- `Table.ts` – definira tip podatka `Table` koji se odnosi na stolove unutar objekta restorana te definira sva potrebna svojstva za opis stola.
- `TableCircleShapes` i `TableCubeShapes` – su `enum` klase u kojima su definirani svi oblici stolova koje jedan stol može poprimiti uključujući i položaj i razmještaj stolica kao i njihov broj.
- `TaxRate.ts` – je datoteka u kojem je definiran `enum` klasa s definiranim stopama poreza.
- `User.ts` – je datoteka koja sadrži definiciju `User` tipa podatka koji opisuje jednog registriranog korisnika web aplikacije (slika 4.3.).

```
import type { Role } from "./Role"

export interface User{
  uid: string,
  email: string,
  firstName: string,
  lastName: string,
  imageUrl: string,
  imageName: string,
  born: string,
  phoneNumber: string,
  role: Role
}
```

Sl. 4.3. Prikaz definicije `User` tipa podatka

### 4.3. Prijava, registracija i pristup

Da bi korisnik mogao obavljati narudžbe proizvoda putem aplikacije, prvo se treba registrirati. Nakon toga slijedi prijava s lozinkom i elektroničkom poštom. Za registraciju potrebno je unijeti lozinku i elektroničku poštu te ponoviti lozinku. Prilikom klika na tipku prijave dolazi do provjere ispravnosti lozinke. Lozinka mora sadržavati minimalno 8 znakova, barem jedno veliko i malo slovo, barem jedan broj te barem jedan specijalan znak. Za provjeru elektroničke pošte koristi se funkcija koja provjerava da li unesena pošta sadrži barem jedan alfanumerički znak nakon kojeg sadrži znak `@` te da li sadrži pod domenu i domenu od 2 do 6 slova. Prilikom kreiranja korisničkog računa korisniku se dodjeljuje uloga (engl. `Role`) koja definira koji sadržaj se prikazuje

prijavljenom korisniku. Svim novokreiranim korisnicima dodijeljena je uloga „*USER*“ (Slika 4.4.).

Prilikom kreiranja računa i prijave koriste se ugrađene Firebase funkcije koristeći Firebase *auth* sustav, dok se dodatne informacije o korisniku pohranjuju u *firestore* kolekciju naziva „*users*“. Koje su dodatne informacije pohranjene vidljivo je na slici 4.4.

```
async updateUserData(uid: string, email: string){
  try {
    const docRef = await setDoc(doc(db, "users", uid), {
      uid: uid,
      email: email,
      firstName: '',
      lastName: '',
      imageUrl: '',
      imageName: '',
      born: '',
      phoneNumber: '',
      role: Role.user
    } as User)
  } catch (error) {
    toast.error('Neuspješno kreiranje podataka korisnika')
    console.error("Adding user data error: ", error)
  }
},
```

Sl. 4.4. Kreiranje korisničkih podataka

Svaka krajnja točka web aplikacije definira tko ima prava pristupa tako što su za svaku putanju navedeni meta podaci (Slika 4.5.). Iz navedenog primjera vidljivo je da je „*requiresAuth*“ postavljen na vrijednost „*true*“ što znači da korisnik mora biti autentificiran te je „*requiresRole*“ postavljen na polje dozvoljenih uloga koje imaju pravo pristupa. Kada su oba uvjeta zadovoljena omogućen je pristup navedenoj odredišnoj točki. Za definiranje i postavljanje pravila pristupa odredišnim točkama koristi se *Vue router* u kojem su definirane sve odredišne točke, prava pristupa te koje se komponente pozivaju prilikom pristupanja određenoj putanji.

```
},
{
  path: '/user-orders-and-bills',
  name: 'UserOrdersAndBillsView',
  component: () => import('../views/UserOrdersAndBillsView.vue'),
  meta: { requiresAuth: true, requiresRole: [Role.user, Role.staff, Role.manager, Role.admin] }
},
]
```

Sl. 4.5. Prikaz jedne odredišne točke

Provjera pristupa određenoj krajnjoj točki ostvarena je tako što *Vue router* prilikom navođenja na krajnju točku provjerava da li ta odredišna točka sadrži meta podatak koji zahtjeva autentifikaciju (Slika 4.6.). Ako je autentifikacija potrebna provjerava se je li korisnik autentificiran, ako nije preusmjerava se na početnu stranicu, u suprotnom provjerava se je li za pristupanje potrebna

određena uloga. Ako se u polju dozvoljenih uloga nalazi uloga korisnika koji pristupa, *Vue router* navodi na željenu rutu u suprotnom korisnik nema prava pristupa te se preusmjerava na početnu stranicu. Početna stranica je krajnja točka kojoj imaju pristup svi registrirani korisnici, ali i oni koji to nisu.

```
if (to.matched.some(record => record.meta.requiresAuth)) {
  if (currentUser) {
    if (to.matched.some(record => record.meta.requiresRole)) {
      const userStore = useUserStore()
      const userRole = userStore.user?.role
      const requiredRoles: Role[] = to.meta.requiresRole as Role[]

      if (requiredRoles.includes(userRole!)) {
        next()
      } else {
        next({ name: 'HomeView', query: { redirect: to.fullPath } })
      }
    } else {
      next()
    }
  } else {
    next({ name: 'HomeView', query: { redirect: to.fullPath } })
  }
}
```

Sl. 4.6. Provjera autentifikacije unutar *vue router-a*

#### 4.4. Pretraživanje sirovina i proizvoda

Prilikom stvaranja proizvoda i kategorija proizvoda svaki proizvod sastoji se od određenog broja sirovina, jednako tako svaka kategorija sadrži određeni broj proizvoda. S obzirom na to da broj proizvoda i sirovina može biti duga lista potrebno je omogućiti pretraživanje sirovina i proizvoda kako bi se lista suzila na samo one proizvode i/ili sirovine koje korisnik želi dodati. Za pretraživanje koristi se polje gdje korisnik unosi naziv proizvoda ili sirovine koju želi pronaći (Slika 4.7.).

```
<div class="flex flex-row justify-center w-full mb-2">
  <input
    v-model="searchQuery"
    type="text"
    placeholder="Pretraži sirovine..."
    class="border border-gray-300 rounded px-2 py-1"
  />
</div>
```

Sl. 4.7. Prikaz html-a polja za pretraživanje sirovina

Za pretraživanje se koristi input tag koji je putem *v-model-a* povezan s varijablom *searchQuery*. *V-model* je Vue direktiva za dvosmjerno povezivanje podataka gdje svaka promjena unutar inputa utječe na promjenu vrijednosti varijable *searchQuery*, jednako tako promjena varijable utječe na promjenu inputa. Početna vrijednost varijable postavljena je na prazan *string*, kada korisnik unutar polja unese određeno slovo i kreira riječ, za svaki unos slova *computed* svojstvo (Slika 4.8.) pretražuje sva imena sirovina te filtrira samo one koje sadrže unesenu riječ od strane korisnika. *Computed* svojstvo je reaktivno svojstvo ugrađeno unutar Vue projekta koje na svaku promjenu varijable koja se nalazi unutar svojstva pokreće kod koji se nalazi unutar streličaste funkcije.

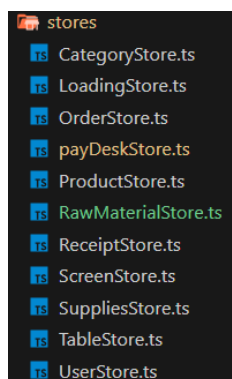
```
const filteredMaterials = computed(() => {
  const query = searchQuery.value.toLowerCase()
  return rawMaterials.value.filter(material =>
    material.name.toLowerCase().includes(query)
  )
})
```

Sl. 4.8. *computed* svojstvo za pretraživanje sirovina

## 4.5. Pohrana, ažuriranje, brisanje i dohvaćanje podataka iz baze podataka

Pohrana, ažuriranje, brisanje i dohvaćanje podataka iz baze je ključno za rad aplikacije, jer bez pristupa bazi aplikacija se ne može koristiti. Smisao aplikacije je da se može pokrenuti bilo gdje, na bilo kojem uređaju koje podržava web preglednike najnovije generacije kako upravljanje aplikacijom ne bi bilo centralizirano na jednom uređaju unutar objekta, već na više njih, gdje je dovoljno samo otvoriti domenu na kojoj se nalazi ova aplikacija i prijaviti se.

Za upravljanje pohranom u bazu za potrebe aplikacije koristi se *pinia store*, biblioteka koja omogućava da se upravlja stanjima koje aplikacija može poprimiti te da se definiraju sve funkcije za pohranu, dohvaćanje, ažuriranje i brisanje podataka iz baze. *Pinia store* omogućava kreiranje i definiranje više različitih pohrana kako bi se odvojile različite cjeline jedne od drugih i kako bi kod bio pregledniji (Slika 4.9.).



Sl. 4.9. Datoteke za upravljanje stanjima aplikacije



- `CategoryStore.ts` – je datoteka za upravljanje stanjima kategorija proizvoda, gdje je svaki proizvod dodijeljen nekoj od kategorija
- `LoadingStore.ts` – je datoteka koja upravlja stanjem koji korisniku prikazuje kada se neki podaci učitavaju u aplikaciju ili pohranjuju u bazu
- `OrderStore.ts` – je datoteka koja upravlja stanjima svih narudžbi korisnika
- `payDeskStore.ts` – je datoteka koja upravlja radom blagajne, od zaprimanja narudžbi do izdavanja računa, kreiranje nove blagajne, prijave i odjave, zaključivanje i onemogućavanje rada blagajne
- `ProductStore.ts` – je datoteka koja upravlja stanjima proizvoda, nudi sve funkcionalnosti za kreiranje novih proizvoda, njihovo ažuriranje i brisanje
- `RawMaterialStore.ts` – je datoteka za upravljanje sirovinama te nudi sve funkcionalnosti za kreiranje sirovina, njihovo ažuriranje i brisanje
- `ReceiptStore.ts` – je datoteka koja nudi funkcionalnost uvoza sirovina te pohranjuje listu svih obavljenih uvoza
- `ScreenStore.ts` – je datoteka koja drži podatke o trenutnoj visini i širini prozora preglednika na kojem se aplikacija pokreće kako bi se ispravno prikazale sve komponente aplikacije
- `SuppliesStore.ts` – je datoteka koja dohvaća i pohranjuje sva stanja zaliha te trenutno stanje
- `TableStore.ts` – je datoteka koja nudi sve funkcionalnosti dodavanja, ažuriranja i brisanja i dohvaćanja stolova iz baze te za svaki stol drži trenutno stanje u kojem se stol nalazi
- `UserStore.ts` – je datoteka koja drži sva stanja korisnika, nudi funkcionalnosti za stvaranje korisnika, prijavu, odjavu, autentifikaciju

U ovom poglavlju dani su neki primjeri čiji principi se primjenjuju i u ostalim pohranama. Svaki pohrana ima svoju definiciju i stanje kojim upravlja (Slika 4.10.). Konvencija je da se za naziv varijable pohrane koristi riječ *use* prije imena same pohrane. Potrebno je prije varijable naziva pohrane dodati ključnu riječ *export* kako bi se pohrana mogla koristiti unutar svih dijelova Vue aplikacije. Također za definiciju koristi se *defineStore* funkcija koja definira cijelu pohranu. Pohrana se sastoji od svojstava kao što su *state*, *actions*, *getters* i dr. State je svojstvo koje drži sva stanja koje zahtjeva određena pohrana u slučaju pohrane proizvoda to je lista proizvoda koja je

```
export const useProductStore = defineStore('productStore', {
  state: () => ({
    products: [] as Product[]
  }),
});
```

Sl. 4.10. Prikaz definicije pohrane za upravljanje proizvodima

*Product[]* tipa podatka. Svaka promjena stanja liste proizvoda reflektira se na sva mjesta unutar aplikacije koja koristi to stanje.

Svojstvo pohrane *actions* koristi se za definiciju funkcija koje upravljaju stanjem same pohrane (Slika 4.11.). Asinkrona funkcija, ključna riječ *async*, označava funkciju koja se ne izvršava sekvencijalno, već je njeno vrijeme izvršavanja nepredvidivo i najčešće ovisi o trećoj strani. U kontekstu pohrane vrijeme izvršavanja funkcije ovisi o brzini odgovora Firebase baze podataka. Cijeli kod za dodavanje se nalazi unutar *try – catch* bloka koji omogućava hvatanje pogreške ako do nje dođe prilikom komunikacije s bazom podataka te kako se greška ne bi propagirala na druge dijelove aplikacije te kako bi se moglo utvrditi zašto je do greške došlo. Za zapisivanje podataka u Firebase koriste se *firestore* ugrađene funkcije *doc* i *setDoc* sa slike 4.11., gdje je *doc* funkcija zadužena za stvaranje kolekcije i predstavlja referencu na kolekciju, a *setDoc* funkcija zadužena je za stvaranje novog objekta unutar kolekcije. Prilikom poziva asinkrone funkcije koristi se ključna riječ *await* koja omogućava da se daljnji kod ne izvršava sve dok funkcija ne vrati vrijednost. Za ažuriranje objekta kolekcije koristi se *firestore* funkcija *updateDoc* te za brisanje *deleteDoc*. Svako od navedenih *firestore* funkcija potrebno je proslijediti referencu koja prima konfiguraciju baze, naziv kolekcije te identifikacijsku oznaku za objekt kojem se pristupa.

```
actions:{
  async addTable(table: Table){
    try {
      table.creationDate = new Date().toLocaleString()
      table.lastTimeUsed = new Date().toISOString()
      const addRef = doc(collection(db, 'tables'))
      table.dbId = addRef.id
      await setDoc(addRef, table as Table)
      console.log("Table added")
      toast.success("Stol dodan")
    } catch (e) {
      console.error("adding table: ", e)
      toast.error('Podreška prilikom dodavanja Stola')
    }
  },
}
```

Sl. 4.11. Primjer asinkrone funkcije za pohranu u *firestore* kolekciju

Za dohvaćanje objekata iz baze koriste se *firestore onSnapshot* i *docChanges* funkcije koje dohvaćaju sve objekte kolekcije te prate promjene koje su se dogodile nad objektima kolekcije. Ukoliko je netko od korisnika dodao novi objekt, izmjeni postojeći ili ga obriše, svi ostali korisnici automatski će detektirati promjenu te ažurirati stanje objekta. Prilikom dohvaćanja objekata *firestore* omogućava upite na temelju uvjeta ka što su *where* i *orderBy*, što omogućava dohvaćanje objekata koji zadovoljavaju uvjet prema nekim od svojstava koje objekt sadrži ili dohvaćanje objekata odgovarajućim redoslijedom prema zadanome svojstvu (Slika 4.12.).

```
async fetchOrders(){
  const collectionRef = collection(db, orderCollection)
  const q = query(collectionRef, where("isApproved", "==", false))

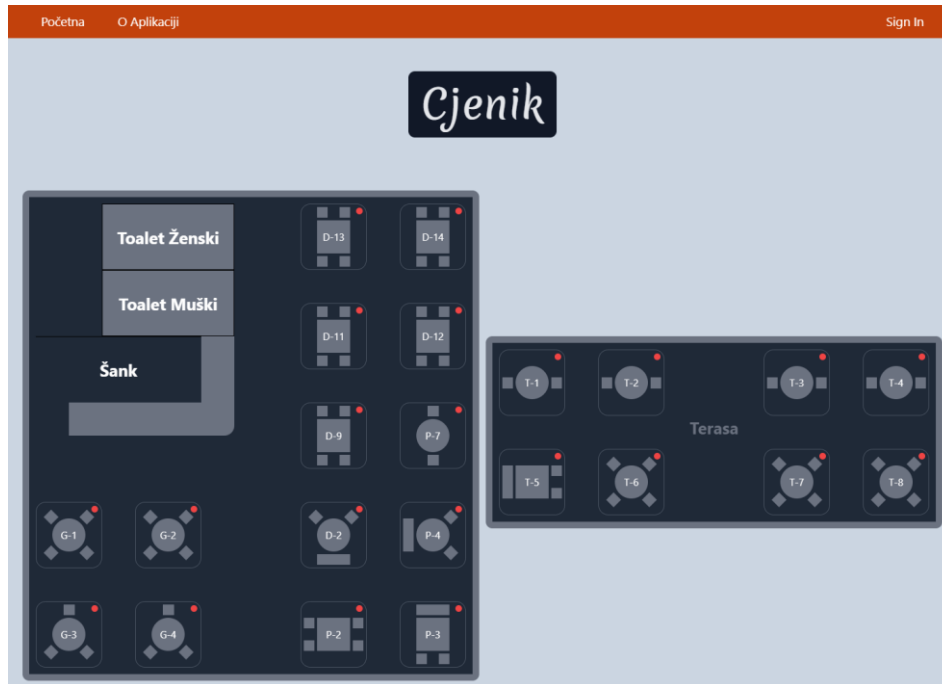
  const unsubscribe = onSnapshot(q, (snapshot) => {
    snapshot.docChanges().forEach((change) => {
```

Sl. 4.12. Dio koda za dohvaćanje narudžbi i korištenje upita

Za pohranu i dohvaćanje pohranjenih slika korisnika, proizvoda i kategorija koristi se Firebase *storage*. Za pohranu slike potrebno ju je smanjiti smanjenjem rezolucije kako bi bila pogodnija za pohranu potrebno ju je pretvoriti u *Blob* tip podatka te se korištenjem *storage* funkcije *uploadBytes* slika pohranjuje u bazu, a za dohvaćanje putanje pohranjene slike koristi se *getDownloadUR'* funkcija, dok se za njeno brisanje koristi *deleteObject* funkcija.

## 5. VIZUALNA REPREZENTACIJA APLIKACIJE

U ovome poglavlju prikazan je vizualna reprezentacija svih pogleda, odnosno stranica koje aplikacija ima, koja je svrha pojedine stranice te opis kako se koristi. Početna stranica kada korisnik nije prijavljen prikazana je na slici 5.1.



Sl. 5.1. Početna stranica web aplikacije neprijavljenih korisnika

### 5.1. Prijava i registracija

Ako korisnik nije prijavljen na početnoj stranici može pristupiti cjeniku gdje su prikazane svi proizvodi u ponudi te pripadajuće cijene. Također, korisnik ima uvid u topografski prikaz restorana gdje su prikazani stolovi, njihovi oblici, nazivi te indikatori koji pokazuje kada je obavljena zadnja prodaja za pripadajućim stolom. Da bi korisnik mogao pristupiti dodatnim mogućnostima stranice potrebno je obaviti prijavu, odnosno registraciju ako korisnik već nije registriran. Četiri je razine pristupa „*USER*“ svi registrirani korisnici koji nisu zaposlenici, „*STAFF*“ zaposlenik niže razine, „*MANAGER*“ menadžer i „*ADMIN*“ administrator. Za prijavu korisnik treba kliknuti „*Prijavi se*“ tipku unutar navigacije koja vodi do stranice za prijavu (Slika 5.2.). Na stranici za prijavu nalazi se dva polja za unos lozinke i elektroničke pošte, tipka za prijavu te tekst „*Nemate kreiran račun? Registrirajte se*“ koji klikom navodi na stranicu za registraciju. Ako korisnik unese neispravnu lozinku i/ili elektroničku poštu unutar forme prikazat će se poruka s greškom, a ako je korisnik ispravno prijavljen bit će preusmjeren na početnu stranicu aplikacije.

EMAIL:

---

PASSWORD:

---

**Prijavi se**

Nemate kreiran račun? Registrirajte se

Sl. 5.2. Prikaz sučelja za prijavu

Za razliku od stranice za prijavu, stranica za registraciju sadrži jedno dodatno polje za lozinku koju treba ponoviti kako ne bi došlo do pogreške prilikom unosa lozinke. Ako su uneseni podaci ispravni kreira se račun korisnika te se korisnik navodi na stranicu za uređivanje vlastitog računa, a ako prilikom registracije neko od polja nije pravilno ispunjeno, doći će do prikaza greške.

## 5.2. Korisnički račun

Nakon što je korisnik uspješno kreirao svoj račun automatski je prijavljen i navođen na stranicu korisničkog računa (Slika 5.3.). S obzirom na to da je korisnik tek registriran polja za ime, prezime, telefon i datum rođenja kao i slika korisničkog računa su prazna i korisnik ih po potrebi može ažurirati, dok je polje za elektroničku poštu ispunjeno i postavljeno prilikom registracije korisnika. Ako korisnik račun ne želi ažurirati odmah, stranici korisničkog računa moguće je pristupiti putem navigacije u gornjem desnom uglu klikom na sliku korisnika koja je početno postavljena na siluetu korisnika kako bi indicirala na njenu svrhu.

Home About Skladište Blagajna Stolovi

Ime: **Dejan**

Prezime: **Mihić**

Telefon: **0977777777**

Datum rođenja: **26.01.1990.**

Email: **dejan@gmail.com**

**Odjava** [→] **Izmjeni** [✎]

Sl. 5.3. Prikaz korisničkog računa nakon ažuriranja

Za ažuriranje informacija o korisniku potrebno je kliknuti na tipku „*Izmjeni*“ koja navodi na stranicu za ažuriranje korisničkog računa. Svaki korisnik može ažurirati svoje ime, prezime, datum rođenja i telefon te može dodati sliku. Za ažuriranje imena i prezimena potrebno je unijeti minimalno jedno slovo, dok telefon treba sadržavati minimalno devet brojeva. Nakon što su željena polja ispunjena klikom na tipku „*Ažuriraj*“, informacije o korisniku su ažurirane te će korisnik biti vraćen na stranicu korisničkog računa.

Nakon ažuriranja informacija o korisniku iste će biti reflektirane na cijelu aplikaciju te će se automatski ažurirati i slika korisnika unutar navigacije što je vidljivo na slici 5.3.. Račun korisnika može se ažurirati neograničen broj puta dok ažuriranje elektroničke pošte nije moguće zato što je ona jedan od jedinstvenih identifikatora korisnika.

Ako se korisnik želi odjaviti iz aplikacije, to je moguće jedino na stranici korisničkog računa klikom na tipku „*Odjava*“, nakon koje je korisnik odjavljen i sve informacije korisnika su obrisane te se za ponovnu prijavu korisnik mora prijaviti lozinkom i elektroničkom poštom.

### **5.3. Skladište**

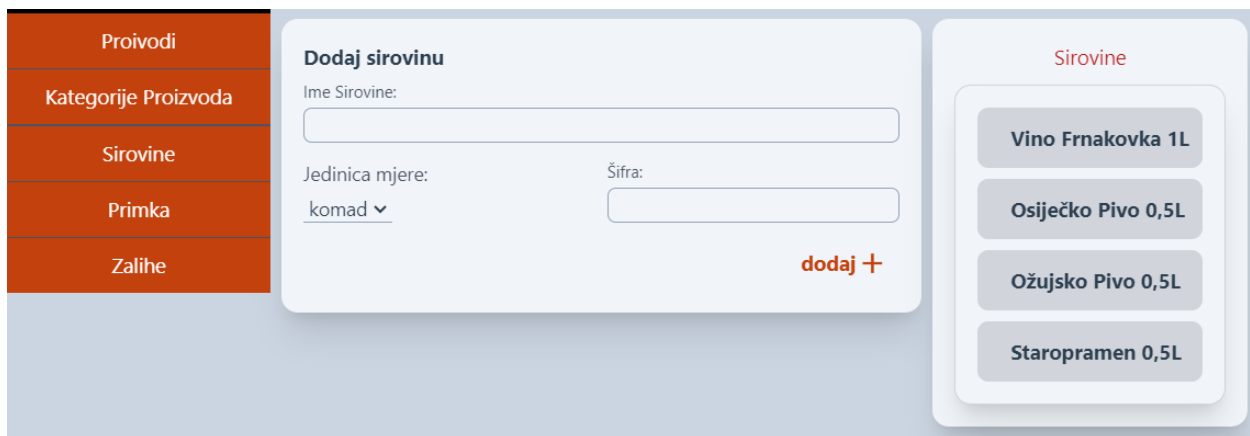
Skladište je jedno od temeljnih dijelova ove aplikacije, gdje se upravlja sa sirovinama, proizvodima, kategorijama proizvoda, uvozom robe, praćenje stanja zaliha i drugo. Za pristup skladištu korisnik mora biti zaposlenik restorana, odnosno mora biti ili menadžer ili administrator restorana. Zaposlenici su registrirani od strane administratora restorana, odnosno vlasnika te im sam vlasnik dodjeljuje prava prilikom kreiranja njihovog računa. Stranica skladišta ima svoju vlastitu navigaciju koja se sastoji od:

- Proizvodi
- Kategorije proizvoda
- Sirovine
- Primke
- Zalihe

Početna stranica skladišta je stranica za dodavanje novih proizvoda koja će kasnije biti detaljno objašnjena. Da bi zaposlenik mogao stvarati nove proizvode, uvoziti sirovine i stvarati zalihe prvo je potrebno dodati sirovine kojima restoran treba raspolagati. Sirovine predstavljaju sve one element koji su potrebni da bi neki od proizvoda bio izgrađen kao npr. pizza je proizvod koji se sastoji od sirovina kao što su tijesto, kečap, sir i dr., dok neki od sirovina mogu ujedno biti i

proizvodi kao što je npr. Coca Cola. Za dodavanje sirovina potrebno je u navigaciji skladišta kliknuti na tipku Sirovine te se otvara stranica za dodavanje sirovina.

Stranica za dodavanje sirovina sastoji od dva sučelja. Lijevo sučelje koristi se za dodavanje sirovina. Svaka sirovina mora imati svoje ime kako bi ih djelatnici razlikovali te ime mora sadržavati minimalno 2 slova. Svakoj sirovini dodjeljuje se i jedinica mjere kako bi bilo jasno definirano kako će se sirovina uvoziti i izvoziti iz aplikacije. Moguće opcije za odabir jedinice mjere su: „komad“, „litra“, „kilogram“ i „gram“. Svaka od sirovina mora imati i pripadajuću šifru koja je jedinstven i broječanog oblika. Ako korisnik unese šifru koja je već u upotrebi ispod polja za unos šifre dolazi do ispisa poruke „šifra već postoji“ te neće biti omogućeno stvaranje sirovine s tom šifrom (Slika 5.4.). Ako su sva polja ispravno popunjena klikom na tipku „dodaj“ sirovina je dodana u bazu.



Sl. 5.4. Prikaz stranice za dodavanje sirovina

Na desnoj strani stranice nalazi se sučelje unutar kojeg se nalazi lista svih dodanih sirovina te se prilikom dodavanja nove sirovine lista automatski ažurira. Klikom na neku od sirovina liste, otvara se sučelje za ažuriranje odabrane sirovine. Za ažuriranje sirovine vrijede ista pravila kao i za dodavanje nove sirovine s izuzetkom da aplikacija prepoznaje da je zauzeta šifra, šifra koja pripada istoj sirovini koja se ažurira te neće doći do greške ako šifra ostane nepromijenjena. Kako bi se sirovina ažurirala potrebno je kliknuti na tipku „ažuriraj“ te će se promjene automatski reflektirati na listu sirovina, kao i cijelu aplikaciju. Za brisanje sirovine potrebno je kliknuti na tipku „obriši“ nakon čega će sirovina biti izbrisana iz baze podataka.

Kada su sve sirovine koje su potrebne za stvaranje proizvoda kreirane, potrebno je uvesti sirovine u samu aplikaciju te takve sirovine predstavljaju zalihe kojima raspolaže restoran. Za uvoz sirovina koristi se stranica *Primke* (Slika 5.5.).

Stranica za uvoz sirovina sastoji se od tri dijela:

- Osnovne informacije
- Sučelje za uvoz sirovina
- Lista dodanih sirovina

**Primka: osnovne informacije**

ime Dobavljača: Roto d.o.o.

oib Dobavljača: 12345678912

Datum: 21.08.2024. Broj primke: 2

Oznaka dokumenta: R-12345-1

**Uvoz sirovina**

Šifra: 0002 Jed. mjere: komad

IME: Ožujsko Pivo 0,5L

Količina: 20,00 komad

Jed. cijena: 1,45 €

Iznos: 29,00 €

**Dodaj Sirovinu +**

**Lista dodanih sirovina**

R. broj:	Šifra:	ime:	jed. mjere:	Količina:	jed. cijena:	Iznos:	Obriši:	Ažuriraj:
1.	0001	Osječko Pivo 0,5L	komad	20,00	1.23 €	24.60 €	X	🔄

**Spremi**

Sl. 5.5. Prikaz stranice za uvoz sirovina

*Osnovne informacije* predstavlja sučelje za unos osnovnih informacija o uvozu sirovina kao što su: ime dobavljača, OIB dobavljača, datum unosa sirovina, broj primke i oznaka dokument po kojem se vrši unos sirovina. Oznaka dokument se preuzima iz računa kojeg dostavlja dobavljač prilikom dostave sirovina. Sva polja obavezna su za popunu. Aplikacija automatski dodjeljuje broj primke te korisnik ne može mijenjati taj broj. Ime dobavljača, kao i oznaka dokumenta moraju sadržavati minimalno 2 znaka, OIB dobavljača mora sadržavati točno 11 brojeva, datum unosa primke ne može biti stariji od 7 dana kako bi primka bila valjana. Ako neko od polja nije pravilno ispravljeno ispod svakog polja ispisuje se pripadajuća poruka pogreške.

Kada su osnovne informacije unesene potrebno je u sučelju za uvoz sirovina upisati podatke o sirovini koja se uvozi, a koja je prethodno definirana u sirovinama. Za uvoz sirovine potrebno je u padajućem izborniku odabrati jedno od ponuđenih imena sirovina ili upisati šifru sirovine te će se automatski prikazati i pripadajuće ime sirovine, vrijedi i obrnuto. Jedinica mjere ne može se mijenjati, već služi kao informacija zaposleniku. Za odabranu sirovinu potrebno je unijeti količinu koja se uvozi u ovisnosti o jedinici mjere te je potrebno unijeti jediničnu cijenu sirovine, koja

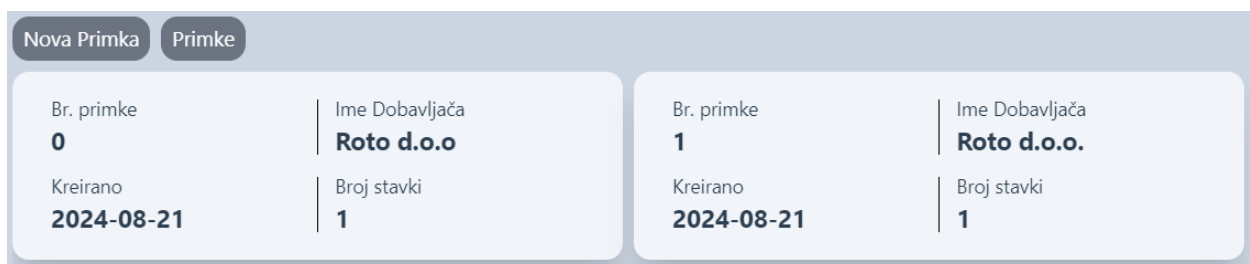


predstavlja nabavnu cijenu sirovine te će se automatski ažurirati iznos koji predstavlja umnožak jedinične cijene i količine. Ako korisnik izmjeni neku od već unesenih vrijednosti, sva polja će se automatski ažurirati. Za uspješno dodavanje sirovine potrebno je kliknuti na tipku „Dodaj Sirovinu“. Ako neka od polja nisu pravilno ispunjena ispod svakog polja bit će ispisana odgovarajuća poruka greške.

Prilikom dodavanja sirovine, svaka dodana sirovina nalazi se u listi dodanih sirovina. Svaka dodana sirovina može se obrisati iz liste klikom na pripadajuću ikonu u stupcu *obriši*, jednako tako svaka dodana sirovina može se i ažurirati klikom na ikonu pod stupcem za *ažuriranje*.

Nakon što su uneseni svi potrebni podaci o primci te unesene sve sirovine za uvoz i spremanje primke te ažuriranje stanja zaliha potrebno je kliknuti na tipku „Spremi“. Ako su unesene sve potrebne osnovne informacije i ako je dodana barem jedna sirovina, primka će biti spremljena u baze te će zalihe sirovina biti uvećane za unesenu količinu pojedinačnih sirovina. Ako niti jedna sirovina nije dodana u listu sirovina ispisat će se pripadajuća greška te primka neće biti spremljena u bazu. Kada je primka uspješno dodana moguće je pregledati listu svih primki koje su dodane tako što korisnik treba kliknuti na tipku „Primke“ koja se nalazi u navigaciji na stranici *Primka*

Na stranici za prikaz liste dodanih primki (Slika 5.6.) korisnik može vidjeti osnovne informacije o svakoj primci, a to su: broj primke, datum kreiranja, ime dobavljača i broj stavki, odnosno sirovina koje se nalaze u primci.



Nova Primka		Primke	
Br. primke	Ime Dobavljača	Br. primke	Ime Dobavljača
<b>0</b>	<b>Roto d.o.o</b>	<b>1</b>	<b>Roto d.o.o.</b>
Kreirano	Broj stavki	Kreirano	Broj stavki
<b>2024-08-21</b>	<b>1</b>	<b>2024-08-21</b>	<b>1</b>

Sl. 5.6. Prikaz korisničkog računa nakon ažuriranja

Klikom na jednu od primki s liste, korisnik ima mogućnost pregledati sve informacije o određenoj primci tako što će se otvoriti stranica za pregled odabrane primke. Svaka dodana primka mijenja stanje zaliha tako da ih uvećava za iznos količine pojedine sirovine koja je navedena u primci.

Da bi zaposlenik pregledao stanje zaliha unutar navigacije skladišta treba kliknuti na tipku „Zalihe“ nakon čega se otvara stranica za prikaz trenutnog stanja zaliha (slika 5.7.). Na stanje zaliha aplikacija utječe na dva načina. Primke na zalihe utječu na način da ih uvećavaju, dok na

smanjenje zaliha utječe prodaja proizvoda izdavanjem računa, tako da ih umanjuje za iznos prodane količine. Na stranici, osim što je prikazano trenutno stanje, prikazane su i informacije o tome koji je dokument uzrokovao promjenu, kada se točno dogodila ta promjena te koji zaposlenik je uzrokovao promjenu stanja zaliha.

Trenutno stanje
Lista svih stanja

### Zalihe- trenutno stanje

Ažuriranje uzrokovano dokumentom: **receipt numb. 5** | Ažurirano: **25. 06. 2024. 18:52:16**

Promjenu uzrokovao korisnik: **dejan@gmail.com**

R. broj:	Šifra:	ime:	jed. mjere:	Količina:
1.	123	Ožujsko Pivo 0.5l	komad	300
2.	1234	Osiječko 0.5l	komad	100
3.	12345	Staropramen 0.5l	komad	20
4.	0123456	Crno vino 1L	litra	6

Sl. 5.7. Prikaz stranice trenutnog stanja zaliha

Da bi korisnik pregledao prethodna stanja zaliha u navigaciji zaliha potrebno je kliknuti na tipku „*Lista svih stanja*“ nakon čega se otvara stranica sa svim prethodnim stanjima, gdje je trenutno stanje označeno zelenom, a svako prethodno crvenom bojom pozadine dokumenta.

Da bi se sirovine pretvorile u konkretan proizvod korisnik u navigaciji skladišta treba kliknuti na tipku „*Proizvodi*“ nakon čega se korisnika navodi na stranicu za dodavanje proizvoda (Slika 5.8.).

Dodaj Proizvod
Svi Proizvodi

Ime artikla:

Šifra:

Jedinica mjere:  | Porezna stopa:

Prodajna cijena:  €

**Sirovine**

Osiječko 0.5l

Staropramen 0.5l

Vino Frankovka 1L

Crno vino 1L

Komponente Proizvoda:

R. broj:	ime:	jed. mje...	Količina:	Ukloni:
1.	Ožujsko Pivo 0.5l	komad	1	⊗

Dodaj Proizvod +

Sl. 5.8. Prikaz stranice za dodavanje proizvoda

Stranica za dodavanje proizvoda sastoji se od sučelja za dodavanje proizvoda i sučelja za odabir sirovina od kojeg se proizvod sastoji. Svakom proizvodu potrebno je dodati sliku, ime, šifru, jedinicu mjere, poreznu stopu i prodajnu cijenu te svaki proizvod mora sadržavati minimalno jednu sirovinu. Ime proizvoda kao i šifra moraju sadržavati minimalno 2, a maksimalno 50 znakova dok se šifra ne smije ponavljati među proizvodima, porezna stopa bira se padajućim izbornikom i može imati vrijednost 0 % ili 25 %, jedinica mjere može biti odabrana jednom od četiri prethodno navedene vrijednosti. Prodajna cijena mora biti brojčana vrijednost s dva decimalna mjesta. Ako neka vrijednost nije ispravno unesena ispod svakog polja ispisuje se odgovarajuća poruka greške.

Da bi se u proizvod dodala neka od sirovina na listi sirovina potrebno je kliknuti sirovinu koja se želi dodati u proizvod. Klikom na sirovinu, sirovina, osim što je dodana u listu komponenti proizvoda, obrisana je iz liste sirovina koje korisnik može dodati u proizvod kako ne bi došlo do dupliciranja sirovina unutar proizvoda. Za brisanje sirovine iz liste komponenti potrebno je kliknuti na ikonu pod stupcem „*Ukloni*“ za odgovarajuću sirovinu te će sirovina biti uklonjena i vraćena u listu sirovina koje se mogu dodati u proizvod.

Prilikom dodavanje sirovina u proizvod moguće ih je pretraživati tako da korisnik unosi ime sirovine koje želi dodati. Prilikom unosa svakog slova lista se smanjuje i unutar nje su prikazane samo one sirovine koje sadrže uzorak unesen u polje za pretraživanje. Kada je sirovina dodana, za svaku sirovinu potrebno je dodijeliti količinu sirovine koju proizvod sadrži. Tako će prilikom prodaje proizvoda doći do umanjena zalihe te sirovine za količinu koja je ovdje navedena. Početna vrijednost postavljena prilikom dodavanje sirovine je nula i klikom na tipku „*dodaj proizvod*“ javlja se greška koja ne dozvoljava da se dodaje proizvod kojem sirovina nema ispravno dodijeljenu količinu od koje se sastoji. Ako su sva polja ispravno ispunjena i proizvod sadrži barem jednu komponentu kojem je količina veća od nula, takav proizvod dodaje se u bazu podataka te se automatski prikazuje kao jedan od mogućih proizvoda u sučelju blagajne. Nakon što je proizvod uspješno dodan pojavljuje se poruka koja potvrđuje uspješno dodavanje proizvoda te aplikacija navodi korisnika na stranicu za pregled svih proizvoda.

Stranici za pregled svih proizvoda korisnik može pristupiti u bilo kojem trenutku tako da klikne na tipku „*Svi Proizvodi*“ koja se nalazi u navigaciji proizvoda. Stranica za pregled svih proizvoda sadrži kartice proizvoda sa svim važnim informacijama o proizvodu kao što je slika, ime, šifra, prodajna cijena, porezna stopa i šifra. Sve kartice proizvoda su interaktivne te klikom na određeni proizvod aplikacija navodi na stranicu za ažuriranje proizvoda koja funkcionira na istome principu kao i stranica za dodavanje proizvoda uz razliku da se određeni proizvod može i obrisati klikom

na tipku „*Obriši*“ te je moguće ažurirati proizvod klikom na tipku „*Ažuriraj*“, čije će se promjene automatski reflektirati na sve dijelove aplikacije koje koriste ažurirani proizvod. Ako je proizvod uspješno ažuriran dolazi do prikaza stranice sa svim proizvodima i porukom o uspješnosti ažuriranja proizvoda. Svi proizvodi koji se nalaze na stranici svih proizvoda su oni koje je moguće koristiti unutar sučelja blagajne kako bi se vršila prodaja istoga.

Kako bi proizvodi bili bolje organizirani, svaki proizvod potrebno je dodati u neku od kategorija proizvoda, gdje kategorija proizvoda predstavlja roditeljski direktoriji u kojem su smješteni svi proizvodi koji su srodni. Koji su to proizvodi srodni te koliko će biti različitih kategorija odlučuje sam korisnik. Kako bi korisnik stvorio kategoriju potrebno je u navigaciji skladišta kliknuti na tipku „*Kategorije Proizvoda*“ nakon kojega se otvara stranica sa sučeljem za stvaranje nove kategorije (Slika 5.9.).

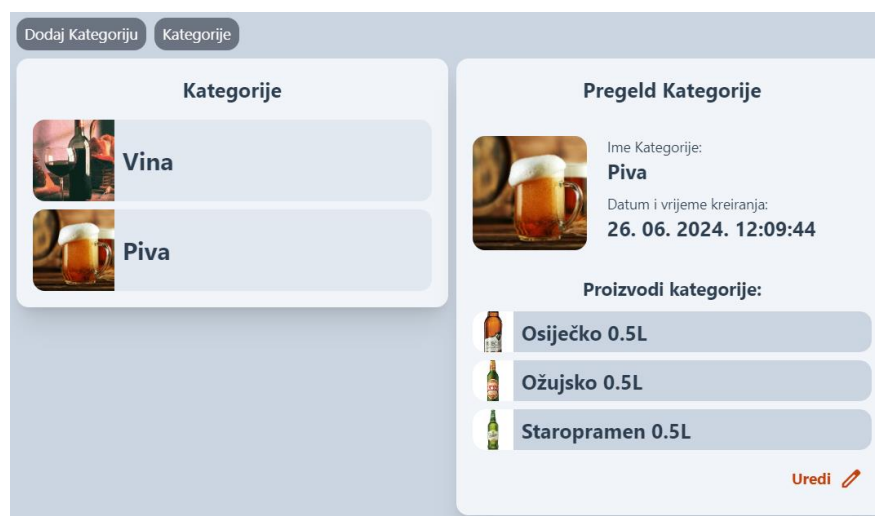


Sl. 5.9. Prikaz stvaranja kategorije „Piva“

Stranica za dodavanje kategorije sastoji se od sučelja za dodavanje kategorije i sučelja sa svim proizvodima koji nisu dio niti jedne kategorije. Za stvaranje nove kategorije korisnik treba odabrati sliku koja najbolje opisuje kategoriju, upisati odgovarajuće ime koje sadrži minimalno 2, a maksimalno 50 znakova te mora sadržavati minimalno jedan proizvod u listi dodanih proizvoda. Ako neke od navedenih stavki nisu ispravno popunjene ispod svake stavke javlja se poruka s pripadajućom greškom. Za dodavanje proizvoda u listu dodanih proizvoda kategorije potrebno je u sučelju za proizvode kliknuti na proizvod koji se želi dodati u listu. Prilikom dodavanja proizvoda u listu dodanih proizvoda kategorije, proizvod se automatski briše iz sučelja s listom svih proizvoda koji nisu dio niti jedne kategorije. Sučelje sa svim proizvodima može se i pretraživati prema imenu proizvoda. Svaki od proizvoda dodan u listu dodanih proizvoda kategorije sadrži pripadajuću sličicu i naziv proizvoda te je pozadina zelene boje (Slika 5.9.).

Prelaskom kursora preko određenog proizvoda pozadina se mijenja u crvenu boju kako bi indicirala na mogućnost brisanja proizvoda iz kategorije. Klikom na proizvod koji se želi ukloniti, proizvod se briše s liste dodanih proizvoda i vraća se u listu svih proizvoda koji još ne pripadaju niti jednoj kategoriji. Klikom na tipku „Dodaj“ kategorija proizvoda se dodaje u bazu ukoliko su sva polja ispravno ispunjena i javlja se obavijest s porukom uspješnosti dodavanja kategorije te se korisniku prikazuje stranica sa svim kategorijama. Stranici s kategorijama moguće je pristupiti i klikom na tipku „Kategorije“ koja se nalazi u navigaciji kategorija proizvoda.

Na stranici svih kategorija proizvoda nalaze se dva sučelja. Sučelje za prikaz svih dodanih kategorija s lijeve strane i pregled odabrane kategorije s desne strane (Slika 5.10.).



Sl. 5.10. Prikaz stranice svih kategorija

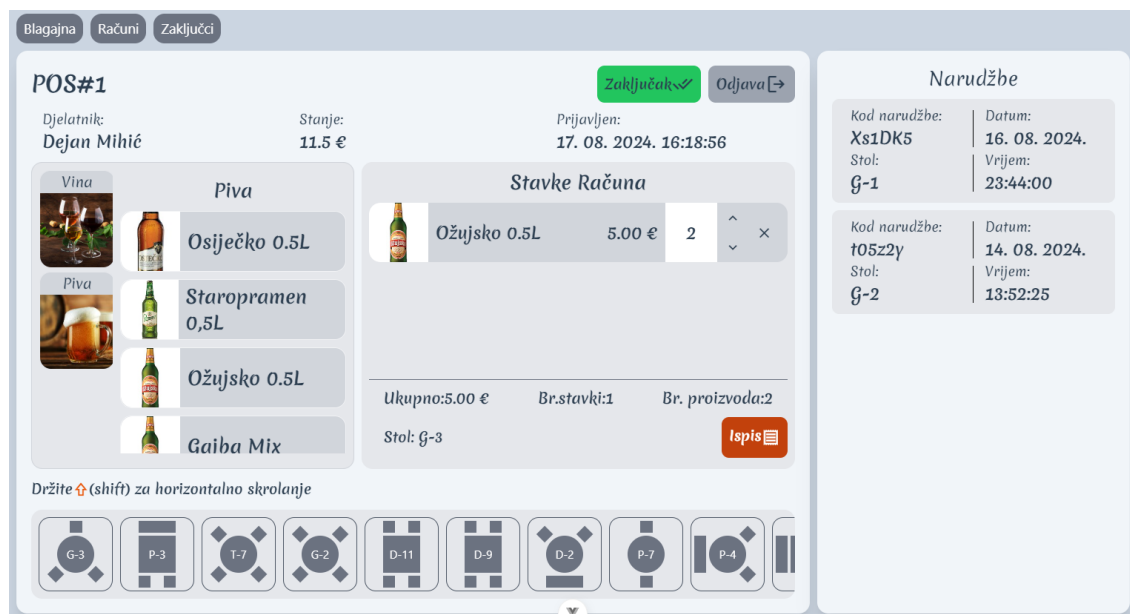
Ako nije označena niti jedna kategorija u sučelju „Kategorije“, u sučelju „Pregled Kategorije“ bit će prikazana poruka kako nije označena niti jedna kategorija. Ako je potrebno određenu kategoriju izmijeniti, potrebno je odabrati odgovarajuću kategoriju te kliknuti na tipku „Uredi“ sučelja za pregled kategorije, nakon čeka se otvara stranica za ažuriranje kategorije.

Stranica za ažuriranje kategorije radi na istom principu kao i stranica za dodavanje kategorije uz tu razliku što su ovdje dvije tipke „Obriši“ i „Ažuriraj“, a tipka „Dodaj“ kako je to u sučelju za dodavanje kategorije. Klikom na tipku „Obriši“ kategorija proizvoda bit će obrisana iz baze te će svi proizvodi koji su bili dio te kategorije postati proizvodi koji nisu dodijeljeni niti jednoj kategoriji te će se naći u sučelju svih proizvoda. Klikom na tipku „Ažuriraj“ kategorija će biti ažurirana novim izmjenama ako su zadovoljena sva pravila jednaka onima za dodavanje nove kategorije.

Skladište je jedan od temeljnih dijelova aplikacije kako bi se omogućio rad aplikacije u punom obujmu. Bez dodanih sirovina, kategorija i proizvoda te uvoza sirovina, nemoguće je obavljati prodaju putem sučelja blagajne.

## 5.4. Blagajna

Blagajna je, uz skladište jedan od temeljnih dijelova aplikacije. Stranici blagajne mogu pristupiti samo djelatnici restorana. Početna stranica blagajne je sučelje za stvaranje računa i pregled narudžbi (Slika 5.11.).



Sl. 5.11. Prikaz sučelja blagajne

Za prikaz sučelja blagajne djelatnik prvotno mora obaviti prijavu na odgovarajuću blagajnu te zatim dolazi do prikaza sučelja blagajne. Sučelje blagajne sastoji se od nekoliko dijelova.

U gornjem dijelu nalaze se osnovne informacije blagajne kao što su njen naziv, ime prijavljenog djelatnika i dr. te tipke za odjavu i zaključak. Ako djelatnik nije izdao niti jedan račun onemogućena je tipka „Zaključak“ blagajne, tek nakon što je djelatnik izdao barem jedan račun moguće je provesti zaključak blagajne. Ako je izdan barem jedan račun, sve dok se ne obavi zaključak nemoguće je odjaviti se iz blagajne klikom na tipku „Odjava“.

Na lijevom djelu sučelja blagajne nalaze se kategorije proizvoda i pripadajući proizvodi kategorije. Za dodavanje proizvoda kao stavke računa potrebno je kliknuti na određenu kategoriju gdje se proizvod nalazi, a zatim i na sam proizvod. Svaka dodana stavka računa sastoji se od slike proizvoda koji je dodan, naziva proizvoda, cijene proizvoda u ovisnost i količini, količine koja se prodaje, dvije ikone u obliku strelice za uvećanje i umanjenje količine te ikone za brisanje stavke,

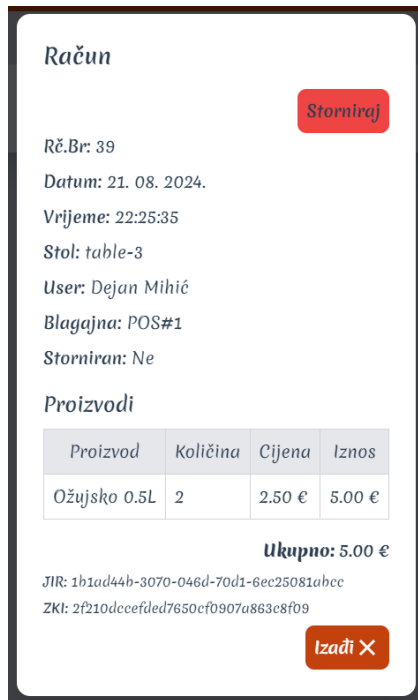
odnosno proizvoda. Početna vrijednost količine za svaki dodani proizvod je jedan te vrijednost količine ne može biti manja od jedan. Klikom na ikonu uvećanja *strelice prema gore* dolazi do uvećanja količine za jedan, u jednakom iznosu umanjuje se količina ako korisnik klikne na ikonu umanjenja *strelice prema dolje*. Uvećanjem i smanjenjem količine automatski se mijenja i cijena stavke. Ako korisnik želi obrisati stavku potrebno je kliknuti na ikonu u obliku „X“. U donjem dijelu sučelja za stavke računa nalaze se tri polja koje prikazuju ukupnu cijenu svih stavki računa, količinu dodanih stavki te ukupan broj proizvoda koji predstavlja broj stavki umnožen s količinom za svaki proizvod zasebno te njihov međusoban zbroj. Prilikom promjene količine pojedine stavke i/ili brisanja pojedine stavke sve navedene vrijednosti automatski se mijenjaju prema novo utvrđenom stanju.

U donjem dijelu nalazi se sučelje za odabir stola, prije ispisa računa potrebno je odabrati stol za koji se račun izdaje kako bi se moglo pratiti kada je neki od stolova posljednje korišten te kako bi svi korisnici koje posjete stranicu imali uvid u posjećenost restorana na početnoj stranici aplikacije.

Kada su sve stavke računa dodane i stol je odabran, korisnik blagajne treba kliknuti na tipku „*Ispis*“ kako bi došlo do ispisa računa, zapisa u bazu te prodaju samih proizvoda. Ispisom računa količina zaliha smanjuje se za iznos količine prodanih proizvoda.

Na desnom dijelu sučelja blagajne nalazi se lista narudžbi. U listi narudžbi nalaze se sve narudžbe nastale od strane prijavljenih korisnika. Klikom djelatnika na neku od narudžbi sve stavke narudžbe učitavaju se u sučelje blagajne i djelatnik obavlja ispis računa. Lista narudžbi nastaje od strane prijavljenih korisnika tako što svaki prijavljeni korisnik na početnoj stranici može kliknuti na odgovarajući stol gdje mu se otvara dijalog za narudžbu koji je gotovo istovjetan sučelju blagajne uz izuzetak što umjesto ispisa dolazi do pohrane narudžbe u bazu gdje se zatim sve narudžbe prikazuju u listi narudžbi blagajne.

U navigaciji blagajne nalaze se još i tipke za navigaciju na stranicu za prikaz računa kao i prikaz zaključaka. Na stranici za prikaz računa nalaze se svi računi koji su izdani od trenutno prijavljenog djelatnika, na trenutno prijavljenoj blagajni. Prikazane su samo osnovne informacije za pojedini račun, a klikom na određeni račun otvara se dijalog sa svim informacijama o računu (Slika 5.12.) te se na vrhu nalazi tipka za storniranje računa „*Storniraj*“, a na dnu tipka za izlazak iz dijaloga „*Izadi*“. Storniranjem računa uvećava se stanje zaliha za količinu sirovina koja je izdavanjem istoga računa prethodno bila oduzeta. Svaki od izdanih računa kao i stornirani računi imaju svoje „*ZKI*“ i „*JIR*“ identifikatore računa te su oni generirani unutar aplikacije i predstavljaju samo pokazne vrijednosti koje se ne mogu koristiti u praksi.



Sl. 5.12. Prikaz računa

Stranica za prikaz zaključaka omogućava odabir raspona datuma za koji se žele dohvatiti zaključci, a početno je postavljen na trenutni dan. Slično kao i za račune na stranici je prikazana lista zaključaka za zadani raspon s osnovnim informacijama, klikom na neki od zaključaka otvara se dijalog s svim informacijama o odabranom zaključku.

Blagajna je ključni dio aplikacije, jer omogućava prodaju proizvoda koju jedan restoran ima u ponudi što je i svrha restorana kao ugostiteljskog objekta, uz ugodnu ambijent, kvalitetu proizvoda i drugo.

## 5.5. Upravljanje

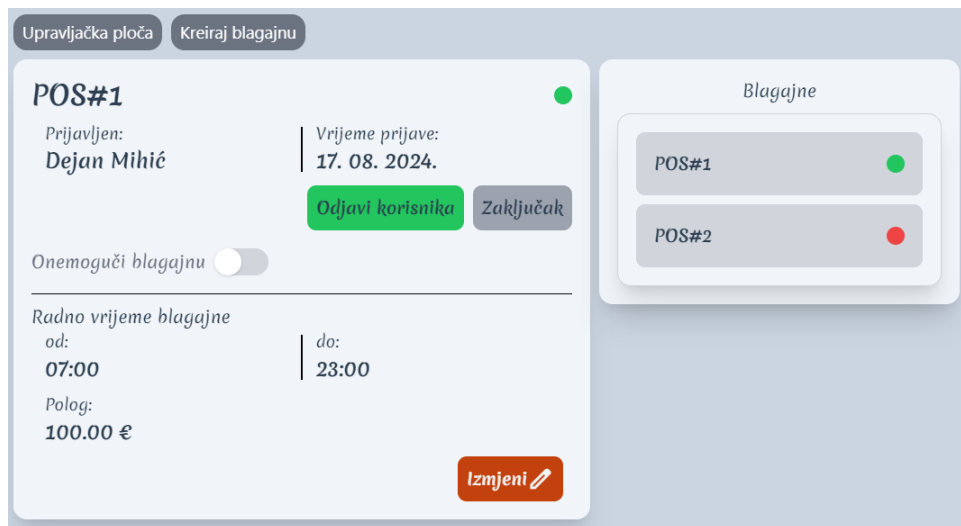
Upravljanje je dio aplikacije kojemu ima pristup samo administrator aplikacije te služi za upravljanje pojedinim dijelovima aplikacije. Stranica za upravljanje ima vlastitu navigaciju koja se sastoji od sljedećih dijelova:

- Blagajna
- Stolovi
- Zaposlenici

Prilikom otvaranja stranice za upravljanje prikazuje se stranica *Blagajna*. Stranica blagajne sastoji se od sučelja za upravljanje blagajnom te sučelja za prikaz svih blagajni i njihov odabir (Slika 5.13.). U sučelju za upravljanje blagajnom nalaze se sve informacije o samoj blagajni, je li blagajna



slobodna ili je djelatnik prijavljen u nju te njegovi podaci. Sučelje omogućuje onemogućavanje blagajne ako se javi potreba da se trenutno prijavljenom djelatniku onemogući rad te omogućava da sam administrator obavi zaključak i odjavu trenutno prijavljenog djelatnika. Također omogućena je izmjena blagajne gdje se klikom na tipku „Izmjeni“ otvara sučelje za izmjenu osnovnih parametara blagajne. Kako bi se prikazale informacije o drugoj blagajni i pripadajuće kontrole potrebno je u listi blagajni kliknuti na željenu blagajnu.



Sl. 5.13. Prikaz stranice za upravljanje blagajnama

U navigaciji za upravljanje blagajne nalazi se i tipka „*Kreiraj blagajnu*“ koja omogućuje kreiranje blagajne tako što se unosi vrijeme rada blagajne, polog i ime blagajne, tek nakon što je blagajna kreirana bit će dodana u bazu podataka i bit će prikazana u listi blagajni.

Stranica *Stolovi* omogućava upravljanje sa svim stolovima objekta. Na stranici se nalazi topografski prikaz restorana s rasporedom sjedećih mjesta u obliku praznih *slotova*, gdje svaki slot predstavlja predefinirano mjesto gdje administrator može dodati stol. Svaki od praznih mjesta četvrtastog je oblika sa pripadajućim identifikatorom. Da bi voditelj dodao stol potrebno je kliknuti na neki od praznih mjesta gdje se želi dodati stol. Klikom na prazno mjesto stranica će se automatski spustiti na mjesto sučelja za dodavanje stola s pripadajućim identifikatorom (Slika 5.14.). Svaki stol mora imati svoj jedinstveni identifikator koji se automatski ispunjava klikom na željeno prazno mjesto, mora sadržavati oblik stola te ime stola. Za oblik stola korisnik može odabrati jedu od ponuđenih oblika. Oblici su podijeljeni na okrugle i četvrtaste stolove s dva, tri ili maksimalno četiri sjedeća mjesta i različitim rasporedima sjedenja. Ime mora sadržavati minimalno 1 do maksimalno 6 znakova te klikom treba odabrati jedan od ponuđenih stolova.



Sl. 5.14. Sučelje za dodavanje stolova

Ako su sva polja ispravno popunjena stol se automatski dodaje na prazno mjesto koje je prethodno odabrano klikom na tipku „Dodaj“, a ako to nije slučaj može se pojaviti jedna grešaka u obliku poruke greške ispod odgovarajućeg polja.

Svaki stol ima svoje ime, oblik, raspored sjedenja i identifikator koji ovisno o boji prikazuje koje je vremensko razdoblje prošlo kada je stol posljednje korišten. Tri su moguće boje:

- Zelena – zadnja narudžba obavljena prije maksimalno 20 minuta
- Narančasta – zadnja narudžba obavljena između 20 i 40 minuta
- Crvena – zadnja narudžba obavljena prije više od 40 minuta

Svaki dodani stol je interaktivan te se klikom na željeni stol otvara dijalog (Slika 5.15.) koji omogućava izmjenu stola s novim imenom i oblikom, ali ne i izmjenu identifikatora, jer je usko povezan s praznim mjestom gdje je stol dodan. Da bi korisnik promijenio oblik stola potrebno je kliknuti na trenutni oblik stola te će se sadržaj dijaloga promijeniti te omogućiti odabir novog oblika stola. Trenutni oblik stola označen je svjetlo sivom bojom pozadine te će se klikom na novi oblik promijeniti boja pozadine kako bi se naznačilo da je novi oblik označen.



Sl. 5.15. Dijalog za ažuriranje stola

Nakon označavanja potrebno je kliknuti na tipku za povratak u obliku strelice u lijevo. Za promjenu imena stola u dijalogu za upravljanje stolom potrebno je kliknuti na ime ispod oblika stola te se mijenja sadržaj dijaloga koji korisniku omogućuje da promjeni ime gdje vrijede pravila da ime mora sadržavati minimalno 1, a maksimalno 6 znakova. Nakon promjene imena potrebno je kliknuti na tipku za povratak u obliku strelice u lijevo. Da bi se ažurirale unesene promjene potrebno je kliknuti na tipku „Ažuriraj“. Ako korisnik želi osloboditi mjesto i na njemu više ne želi imati stol u dijalogu za upravljanje stolom klikom na tipku „Izbriši“, stol će se izbrisati iz baze i više se za taj stol neće moći izdavati računi te on neće biti vidljiv niti jednom korisniku.

Ako ne postoji niti jedan kreirani stol ne može se izdati niti jedan račun, jer je svaki račun povezan s određenim stolom te niti jedan korisnik nema uvid u zauzetost stolova, što je jedna od svrha ove aplikacije.

Prilikom otvaranja stranice *Zaposlenici* prikazuje se sučelje za dodavanje novih zaposlenika. Dodavanje novog zaposlenika zahtjeva od administratora unos imena, prezimena zaposlenika te lozinku ulogu i elektroničku poštu, dok ostala polja nisu obavezna. Nakon što je novi korisnik dodan u bazu administrator više ne može pristupiti izmjeni korisničkih podatak već to može obaviti jedino sam korisnik koji je kreiran. Administrator jedino može izmijeniti ulogu korisnika i to tak da u navigaciji stranice *Zaposlenik* klikne na „Ažuriraj ulogu“ pri čemu se otvara stranica sa sučeljem koje prikazuje sve korisnik (Slika 5.16.).



Sl. 5.16. Sučelje za ažuriranje uloge korisnika i zaposlenika

Sučelje omogućava pretraživanje gdje se prilikom unosa pretražuju korisnička imena, prezimena i naziv elektroničke pošte te će u listi biti sadržani samo korisnici koji zadovoljavaju uneseni tekst. Također sučelje nudi prikaz samo onih korisnika koji su zaposlenici, samo onih koji nisu

zaposlenici već su samo korisnici te prikaz i korisnika i zaposlenika. Za svakog korisnika prikazana je korisnička slika ako je postavljena, ime korisnika te uloga koju korisnik ima. Također uz svakog korisnika nalazi se i tipka za prikaz više opcija gdje administrator može promijeniti i ažurirati ulogu bilo kojeg korisnika ili zaposlenika.

## 5.6. Narudžbe i računi

Stranica *Narudžbe i računi* vidljiva je svim prijavljenim korisnicima te njoj svaki korisnik može vidjeti sve svoje obavljene narudžbe i račune. Stranica se sastoji od sučelja s tri stupca (Slika 5.17.). Prvi stupac prikazuje listu svih narudžbi koje je korisnik izvršio. Za svaku narudžbu prikazane su samo osnovne informacije kao što su datum i vrijeme izdavanja narudžbe, naziv stola za koji je izdana te indikator koji prikazuje u kojem je procesu narudžba. Ako je narudžba tek zaprimljena indikator će prikazivati samo crvenu točku crvene boje pozadine i postotak od 0 %. Ako je djelatnik koji upravlja blagajnom prihvatio narudžbu indikator će promijeniti boju u narančastu i postotak od 50 %. Kada djelatnik pregleda stavke narudžbe i izda račun indikator će pokazivati 100 % zelenom bojom. Klikom na jednu od narudžbi u trećem stupcu sučelja prikazuju se detalji narudžbe, gdje korisnik može vidjeti sve informacije o narudžbi. U drugome stupcu nalazi se lista svih računa koji su izdani putem narudžbe tog korisnika, lista sadrži samo osnovne informacije o svakom pojedinom računu, ako korisnik želi vidjeti sve informacije o pojedinom računu, potrebno je kliknuti na neki od račune te će se u trećem stupcu sučelja prikazati svi detalji za traženi račun.

The screenshot displays a user interface for managing orders and invoices. It is divided into three main sections: 'Vaše Narudžbe' (Your Orders), 'Vaši Računi' (Your Invoices), and 'Narudžba' (Order Details).

**Vaše Narudžbe:** This section lists five orders with their respective dates, times, codes, and table numbers. Each order has a progress indicator (a bar chart) showing its completion status:
 

- Order 1: 14. 08. 2024, 13:03:31, Code: jfQkwq, Table: G-1. Progress: 100% (green bar).
- Order 2: 17. 08. 2024, 16:20:14, Code: jhvRMD, Table: G-4. Progress: 100% (green bar).
- Order 3: 13. 08. 2024, 18:32:13, Code: AFmzXQ, Table: G-2. Progress: 100% (green bar).
- Order 4: 21. 08. 2024, 21:42:58, Code: 5nkG3U, Table: G-1. Progress: 0% (red dot).
- Order 5: 16. 08. 2024, 11:35:18, Code: (empty), Table: (empty). Progress: 0% (red dot).

**Vaši Računi:** This section lists five invoices with their respective dates, times, codes, and amounts:
 

- Invoice 1: 13. 08. 2024, 15:04:05, Code: 13 POS#1, Amount: 2.5 €. Table: G-1.
- Invoice 2: 17. 08. 2024, 16:14:02, Code: 34 POS#1, Amount: 5 €. Table: D-2.
- Invoice 3: 17. 08. 2024, 16:13:31, Code: 32 POS#1, Amount: 8.5 €. Table: G-1.
- Invoice 4: 13. 08. 2024, 18:33:13, Code: 14 POS#1, Amount: 27 €. Table: G-2.
- Invoice 5: 14. 08. 2024, 13:04:17, Code: (empty), Amount: (empty). Table: (empty).

**Narudžba:** This section shows details for a specific order:
 

- Code: jfQkwq, Blagajna: POS#1
- Datum: 14. 08. 2024., Vrijeme: 13:03:31
- Stol: G-1
- Narudžba zaprimljena: Da, Narudžba izvršena: Da
- Proizvodi: A table listing items: Staropramen 0,5L (1 unit, 3.00 € each, total 3.00 €).

Sl. 5.17. Sučelje za prikaz narudžbi i računa prijavljenog korisnika

Svrha stranice za narudžbe i račune je mogućnost da korisnik prati proces u kojem mu se narudžba nalazi, da u konačnici može pregledati koje je to i kakve narudžbe imao te kako bi se povećala transparentnost

## 6. ZAKLJUČAK

U današnje vrijeme živi se ubrzanim životom te su velika očekivanja za brzom i kvalitetnom uslugom. Iako restorani često nude vrlo visoku kvalitetu proizvoda, ne nude nužno i veliku kvalitetu kompletne usluge, a samim time to se reflektira i na brzinu. Jedan od načina za povećanje kvalitete usluge je da se korisnika uključi u sam rad restorana, tako da se korisniku putem web aplikacije omogući veći utjecaj na same usluge koje restoran nudi. U ovome radu objašnjeno je kako izraditi web aplikaciju koja nudi sve funkcionalnosti potrebne da se korisniku približe sve usluge i informacije koje restoran nudi.

Web aplikacija za vođenje restorana ostvarena je korištenjem *Vue* razvojnog okvira koji omogućava izradu fleksibilnih, reaktivnih i modernih web stranica. Aplikacija se sastoji od četiri razine te ovisno o razini pristupa korisnik ima uvid u zauzetost sjedećih mjesta putem topografskog pregleda restorana, može vršiti narudžbe proizvoda te ima uvid u cijene proizvoda i/ili usluga. Samo korisnici koji su zaposlenici restorana imaju mogućnost rada s blagajnom, izdavanja računa, rad sa skladištem koje uključuje stvaranje sirovina, uvoz sirovina, pregled zaliha, stvaranje proizvoda te stvaranja kategorija kojim proizvodi pripadaju. Korisnici s najvećom razinom pristupa su voditelji koji mogu upravljati računima zaposlenika te upravljati s rasporedom i brojem stolova.

Svrha ovoga rada ponuditi konceptualno rješenje kako povećati kvalitetu usluge u restoranima, kako na jednostavan način voditi restoran te kroz izradu web aplikacije naučiti kako se koriste pojedine tehnologije kako bi se povećalo znanje i konkurentnost programera na tržištu.

## LITERATURA

- [1] »Zoho Inventory,« Zoho, [Mrežno]. Available: <https://www.zoho.com/us/inventory/>. [Pokušaj pristupa 16. 6. 2024.].
- [2] »Unleashed,« Unleashed software, [Mrežno]. Available: <https://www.unleashedsoftware.com/>. [Pokušaj pristupa 16. 6. 2024.].
- [3] »About Node.js,« nodejs, [Mrežno]. Available: <https://nodejs.org/en/about>. [Pokušaj pristupa 16. 6. 2024.].
- [4] »A point of sale for however you sell,« Square, [Mrežno]. Available: <https://squareup.com/us/en/point-of-sale>. [Pokušaj pristupa 16. 6. 2024.].
- [5] »Revel System,« Revel A Shift4 Company, [Mrežno]. Available: <https://revelsystems.com/>. [Pokušaj pristupa 16. 6. 2024.].
- [6] »Frequently Asked Questions,« vuejs, [Mrežno]. Available: <https://vuejs.org/about/faq.html>. [Pokušaj pristupa 16. 6. 2024.].
- [7] »HTML: HyperText Markup Language,« [Mrežno]. Available: <https://developer.mozilla.org/en-US/docs/Web/HTML>. [Pokušaj pristupa 16. 6. 2024.].
- [8] »What is CSS, and why is it important?,« Bigcommerce, [Mrežno]. Available: <https://www.bigcommerce.com/glossary/css/>. [Pokušaj pristupa 16. 6. 2024.].
- [9] »Get started with Tailwind CSS,« tailwindcss, [Mrežno]. Available: <https://tailwindcss.com/docs/installation>. [Pokušaj pristupa 16. 6. 2024.].
- [10] »What is TypeScript? Definition, History, Features and Uses,« Invedus, [Mrežno]. Available: <https://invedus.com/blog/what-is-typescript-definition-history-features-and-uses-of-typescript/>. [Pokušaj pristupa 17. 6. 2024.].
- [11] »What is Firebase?,« Educative, [Mrežno]. Available: <https://www.educative.io/answers/what-is-firebase>. [Pokušaj pristupa 17. 6. 2024.].

## SAŽETAK

Osnovna ideja završnog rada je stvoriti web aplikaciju koja omogućava vođenje svih dijelova restorana na jednostavan i intuitivan način. Približiti korisnika samome restoranu, tako da bude uključen u rad restorana kako bi se povećala kvaliteta i brzina usluge koji nudi te svim korisnicima omogućiti jedinstven pristup putem web sučelja jednak za sve korisnike. Ovaj rad pokazuje kako primjenom tehnologija kao što su Vue.js, Node.js i Firebase izraditi web aplikaciju za vođenje restorana, ali i druge srodne aplikacije.

Ključne riječi: Firebase, Node.js, restoran, Vue.js, vođenje, web



## **ABSTRACT**

Web application for restaurant management

The main idea of the final project is to make a web application that helps manage all parts of a restaurant in an easy and clear way. The goal is to connect the user with the restaurant, getting them involved in its work to improve the quality and speed of service, and to give all users the same access through a web interface. This project shows how to use technologies like Vue.js, Node.js, and Firebase to make a web application for running a restaurant, and also other similar applications.

Key words: Firebase, management, Node.js, restaurant , Vue.js, web

## ŽIVOTOPIS

Dejan Mihić rođen je 26. 01. 1990. godine u Našicama, Hrvatska. Osnovnoškolsko obrazovanje završava u „*Osnovna škola Hinka Juhna, Podgorač*“, a srednjoškolsko obrazovanje završava u „*Srednja škola Isidora Kršnjavoga, Našice*“ sa zanimanjem „*tehničar za elektroniku*“ 2008. godine. Nakon završetka srednje škole upisuje „*Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku*“ smjer „*informatika*“ kojeg ne uspijeva završiti. Nekoliko godina obavlja građevinske poslove u Njemačkoj, nakon čega se vraća u Hrvatsku i obavlja konobarske poslove. 2020. godine odlučuje ponovo upisati isti fakultet smjera „*računarstvo*“ te u tu svrhu uspješno polaže državnu maturu, nakon čega uspješno upisuje fakultet.

---

Potpis autora