Web aplikacija za planiranje osobnih događaja

Balko Macsai, Zoltan

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

Sveučilišni studij

WEB APLIKACIJA ZA PLANIRANJE OSOBNIH DOGAĐAJA

Završni rad

Zoltan Balko Macsai

Osijek, 2022.

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

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 11.07.2022.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Zoltan Balko Macsai
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	R 4314, 22.07.2019.
OIB Pristupnika:	69471815966
Mentor:	Prof. dr. sc. Krešimir Nenadić
Sumentor:	,
Sumentor iz tvrtke:	
Naslov završnog rada:	Web aplikacija za planiranje osobnih događaja
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	Opisati funkcionalnosti osobnog planera. Objasniti kako se navedene funkcionalnosti mogu realizirati pomoću web aplikacije. Za potrebe pohrane podataka modelirati i izraditi bazu podataka. Potrebno je izraditi korisničko sučelje i povezati funkcionalnosti u jednu cjelinu. Tema rezervirana: Zoltan Balko Macsai
Prijedlog ocjene završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	11.07.2022.
Datum potvrde ocjene od strane Odbora:	07.09.2022.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 05.09.2022.

Ime i prezime studenta:	Zoltan Balko Macsai
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R 4314, 22.07.2019.
Turnitin podudaranje [%]:	6

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za planiranje osobnih događaja**

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

i sumentora ,

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

Potpis studenta:

SADRŽAJ

1. UVOD

Ljudski je zaboravljati određene stvari, također ljudski je zapisati prijateljev datum rođendana te imati podsjetnik. Razvojem tehnologije sve manje ljudi danas zapisuje događaje i planove u rokovnike i osobne planere. Zbog toga velike kompanije proizvode web i desktop aplikacije za svoje radnike kako bi im olakšali organizaciju svakodnevnih zadataka. Također, postoje kalendarske aplikacije u kojima je moguće zabilježiti određene dane kako ne koje je važno upamtiti određene i važne događaje. Najjednostavniji način je izrada web aplikacije jer je za njezino pokretanje su potrebni samo web preglednik i Internet veza. Nije potrebno niti preuzimati niti instalirati aplikaciju, već je samo potrebna poveznica na lokaciju web aplikacije te se nakon registracije odmah počinje s radom, kreiraju planovi i zadaju ciljevi. Bilo kad moguće ih uređivati ili obrisati, također aplikacije imaju mogućnost slanja obavijesti.

Za početak, potrebno je odrediti definiciju web aplikacije. To je aplikacija koju korisnik pokreće preko web preglednika i kojom upravlja određenim događajima Nije potrebno preuzimanje aplikacije, nego se sve odvija preko web preglednika odnosno postoji vanjski poslužitelj koji omogućava prikaz informacija aplikacije na web pregledniku. Cilj web aplikacija je velika pristupačnost, odnosno brz način pristupa aplikaciji bez potrebe instaliranja i dugotrajnih dohvaćanja podataka. Iako korisnikova Internet veza može bit sporija, web programeri teže što bržem učitavanju sadržaja aplikacije te se zato koriste naprednim tehnologijama web programiranja.

U ovom završnom radu bit će prikazanje tehnologije potrebne za izradu ovakve web aplikacije. Kombiniranjem tih alata nastaju velike komponente s kojima izrađuju dijelovi aplikacije, kao što su kalendar i komponenta plana. Nadalje prikazano će biti kako ovakvu aplikaciju koristiti, od prijave sve do izrade planova. Sve to praćeno sa slikama i kodom te na kraju zaokruženo s jednostavnim zaključkom.

1.1 Zadatak završnog rada

Opisati funkcionalnosti osobnog planera. Objasniti kako se navedene funkcionalnosti mogu realizirati pomoću web aplikacije. Za potrebe pohrane podataka modelirati i izraditi bazu podataka. Potrebno je izraditi korisničko sučelje i povezati funkcionalnosti u jednu cjelinu.

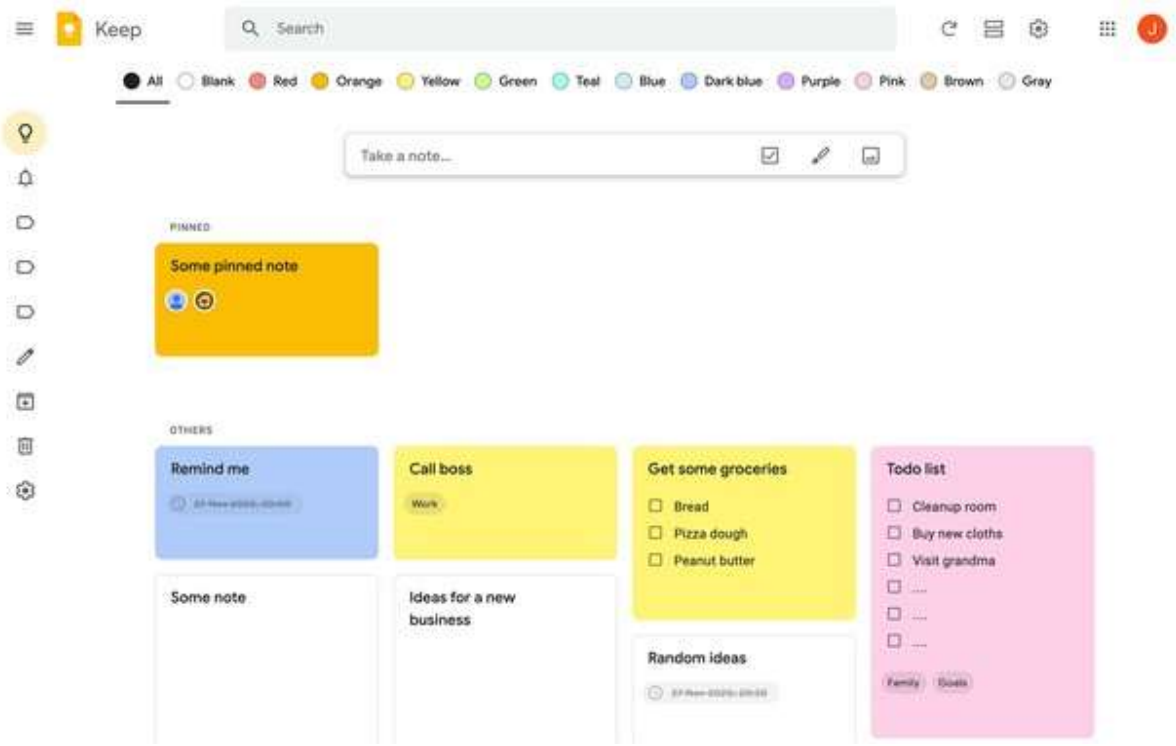
2. PODRUČJE PROBLEMA I SLIČNA RJEŠENJA

Web aplikacija za planiranje događaja ne razlikuje se puno od drugih sličnih rješenja. Zapravo su slična rješenja korištena kao izvor za izradu ove aplikacije. Kombiniranjem rada ovih sličnih rješenja kreirana je aplikacija koja ispunjava osnovne funkcionalnosti spremanja planova za korisnikov određen dan. Također, ova je aplikacija rađena na modelu *slobodno svima*, koji se pokazao kao najbolja opcija za ograničenu funkcionalnost aplikacije.

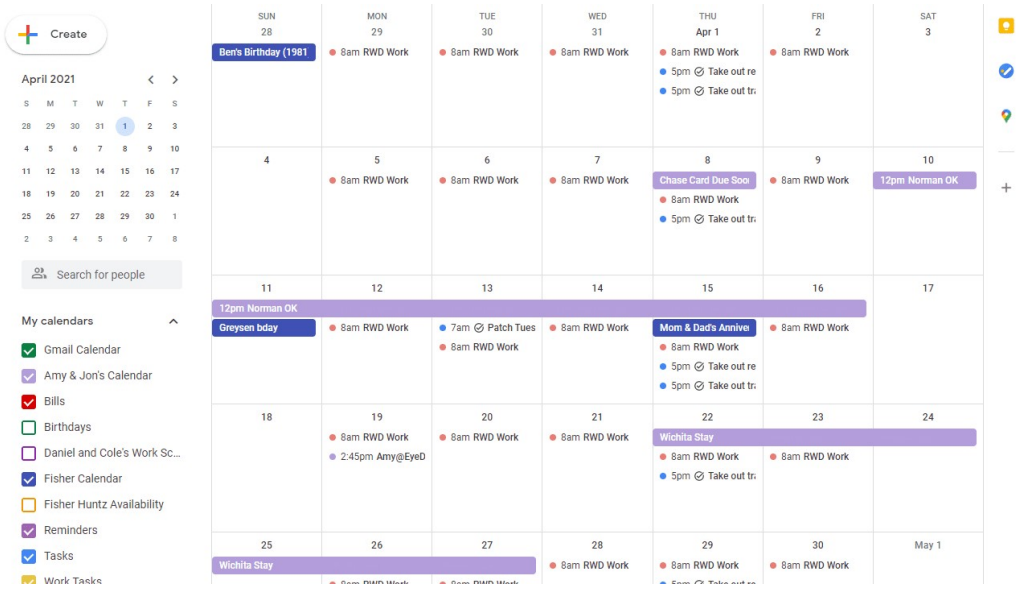
Sva slična rješenja su od velikih kompanija te imaju *zaključanih* funkcionalnosti kako bi kompanije profitirale od svojih korisnika koji imaju potrebu za tim funkcionalnostima. Najčešći način je mjesečno plaćanje u kojem korisnik ulazi u obvezu redovitog plaćanja kako bi nastavio s korištenjem te aplikacije. Ovaj način plaćanja je pogodan za korisnike koji moraju svakodnevno koristiti funkcionalnosti aplikacije. Drugi način je *pay as you use* plaćanje gdje korisnik plaća samo količinu vremena aktivnog korištenja aplikacije. Ovaj model je bolji ako nije potrebna svakodnevna velika upotreba funkcionalnosti aplikacije.

2.1 Google Keep i Google Kalendar

Ova dva rješenja Google-a su poprilično slična. Google Keep(slika 2.1) je web aplikacija za čuvanje bilješki koja omogućava izradu listi namirnica ili planiranje događaja u obliku natuknica koje se mogu prekrižiti, odnosno označiti da su obavljene. Nažalost, Google je Keep-u okončao podršku jer aplikacija nije imala dovoljan doprinos, iako je aplikaciju i dalje moguće koristiti. [1] Dok, Google Kalendar ima opciju odabira dana na kalendaru te na istom kreirati plan. Također, može se odrediti vremensko razdoblje trajanja događaja ili staviti da se radi o cijelom danu, koje će intuitivno se rastegnuti na to određeno razdoblje, kao prikazano na slici 2.2. [2]



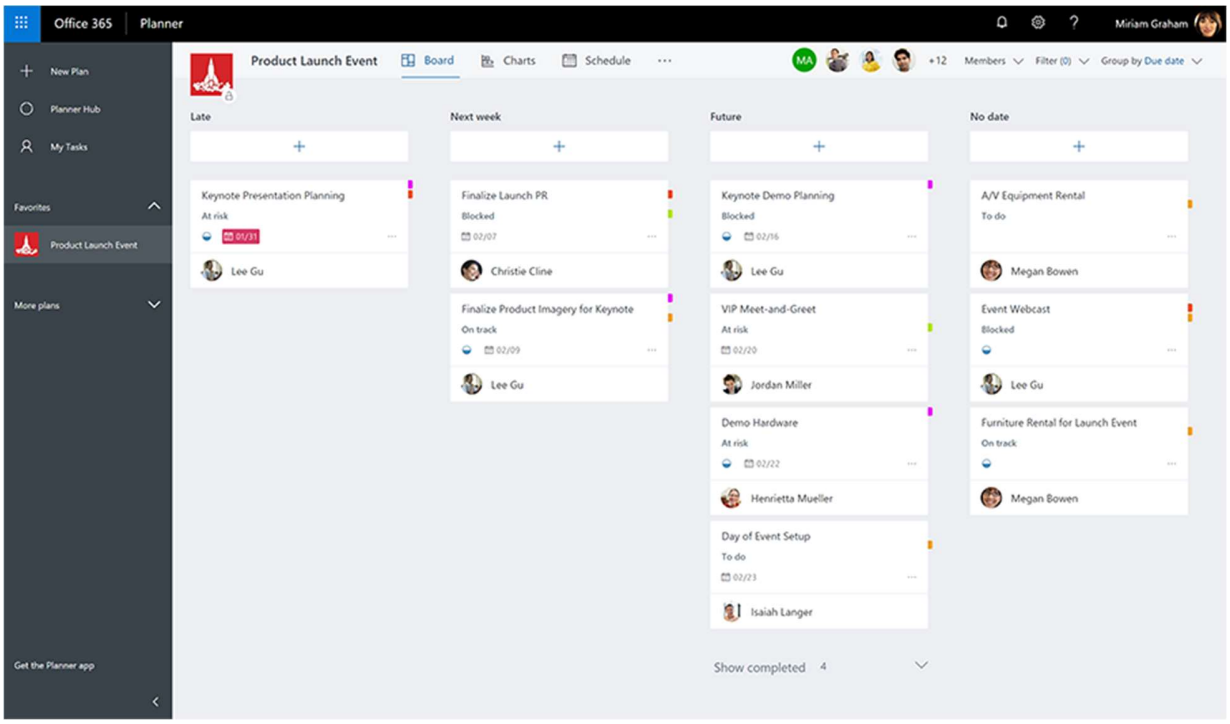
Sl. 2.1 Google Keep



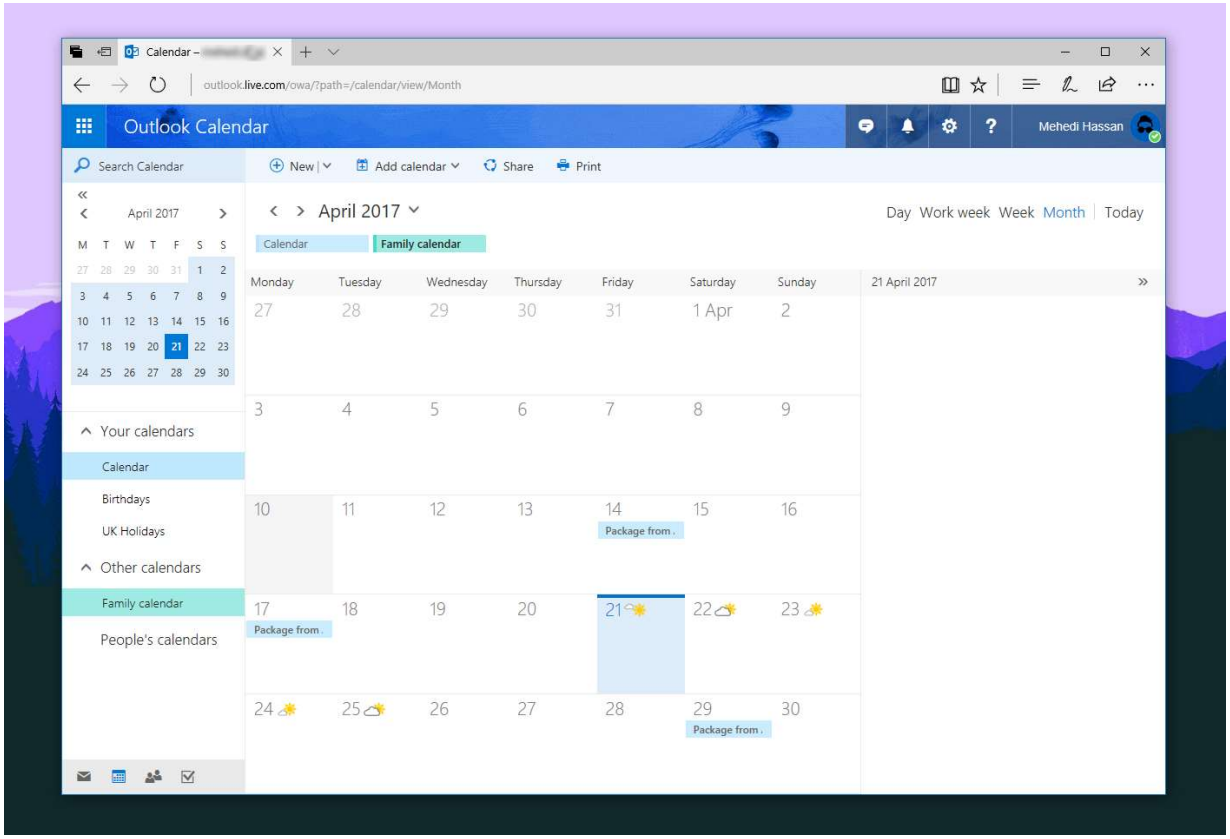
Sl. 2.2 Google Kalendar

2.2. Microsoft Planner i Microsoft Kalendar

Microsoft-ovo rješenje, Microsoft Planner, je planer događaja koji ne izgleda ni kao Google Kalendar, ali ni kao Google Keep, vidljivo na slici 2.3. Njegova funkcionalnost je planiranje projekata i zadataka unutar tima, ima prikaz napretka zadataka i njegovih pod zadataka. Timovi mogu surađivati u izradi planova, ali ima i mogućnost zaključanog rada gdje samo jedan član tima (obično menadžer) ima pravo uređivanja planova. Također, korisnici imaju mogućnost korištenja ovoga alata unutar Microsoft Teams poziva ili za spremanje planova u Microsoft Sharepoint-u. Microsoftov Kalendar je dio Outlook paketa i dolazi s opcijama odabira dana na kalendaru, uređivanja i dodavanja novih događaja za određeni dan, prikazano na slici 2.4. Također, ima opciju sinkronizacije s Microsoft Plannerom. [3]



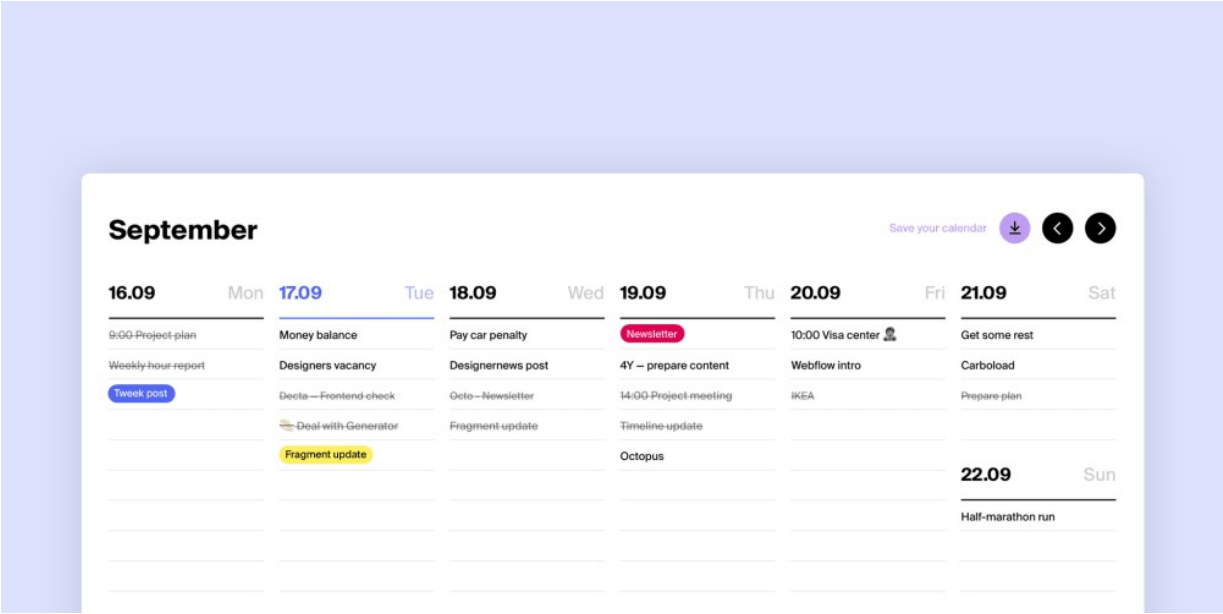
Sl. 2.3 Microsoft Planner



Sl. 2.4 Microsoft Kalendar

2.3 Tweek Calendar

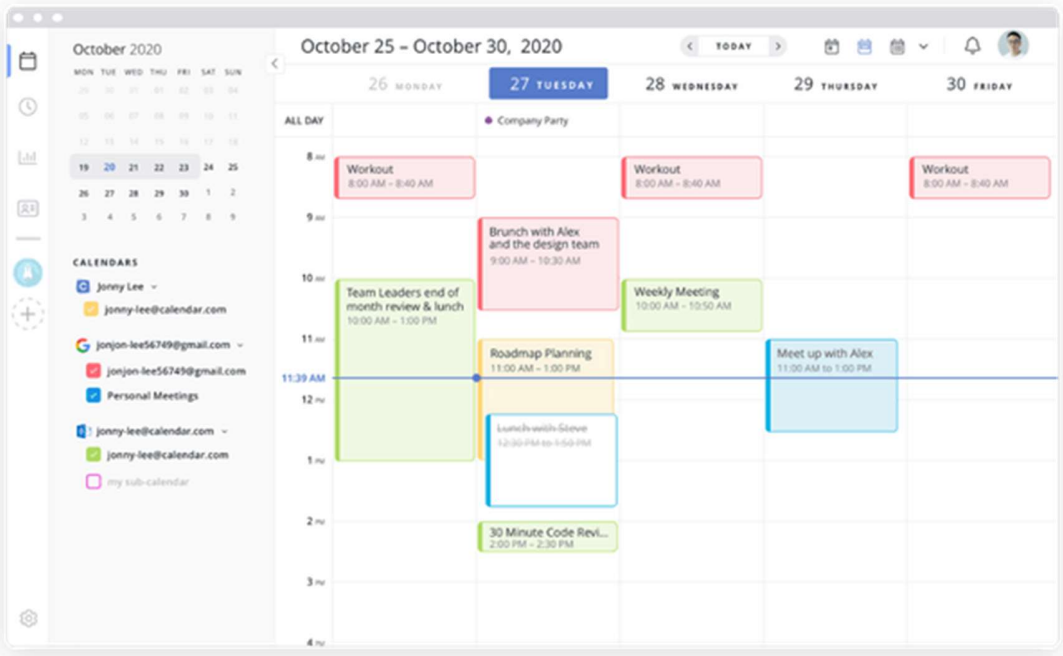
Ovaj jednostavan kalendar(slika 2.5) ima prikaz obaveza na tjednoj bazi. Osim mogućnosti spremanja bilješki, ima mogućnost offline rada korištenjem priručne memorije uređaja te je u mogućnosti ažurirati planove i držati korisnikove događaje unutar rasporeda. Prilikom ponovnog povezivanja na Internet, priručna memorija se spaja s online bazom gdje se onda i ažurira online baza planova. Također, koristi Google-ov Firebase Realtime Database noSQL bazu podataka. U besplatnoj verziji korisnik može imati dva zasebna kalendara. Plaćenu verziju najviše koriste tvrtke kako bi svakom od svojih zaposlenika razdvojili osobni kalendar od poslovnog kalendara te na taj način zaposlenik ima veću koncentraciju na zadatke dodijeljene na poslu. [4]



Sl. 2.5 Tweek Calendar

2.4 Calendar

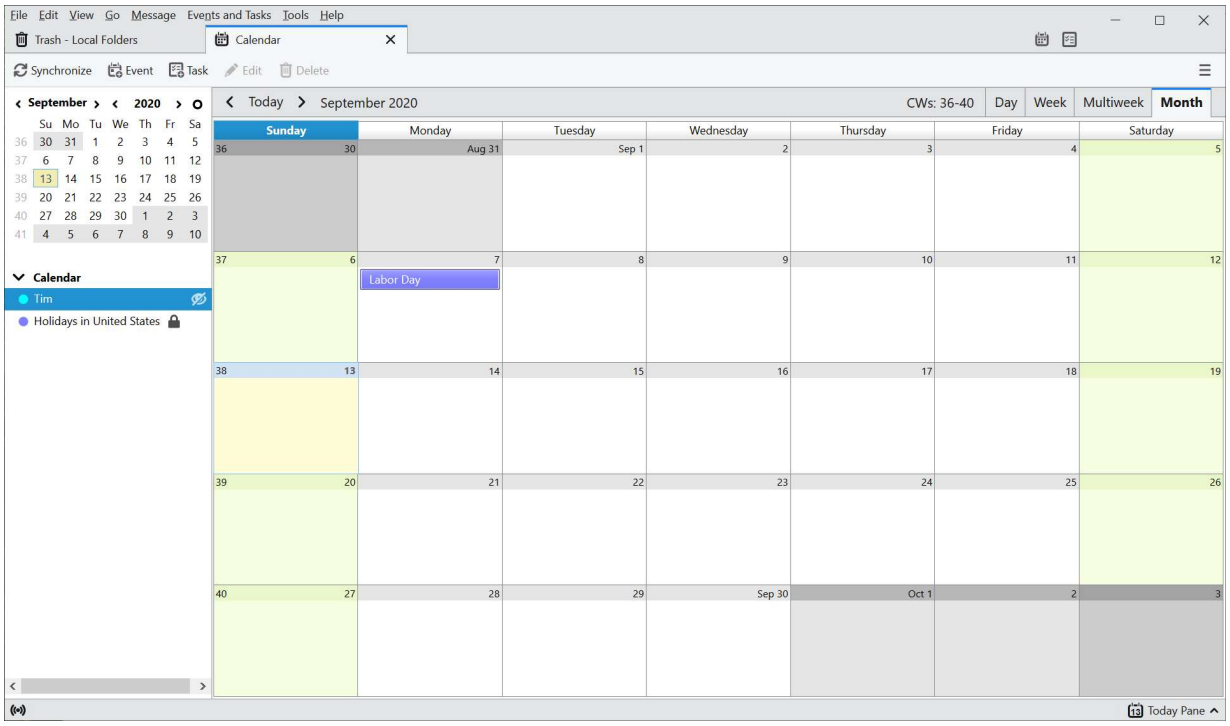
Kako i ime govori, ovo rješenje je kalendar koji kao i Google Calendar i Tweek Calendar ima opciju pisanja bilješki i planiranja događaja na kalendaru, prikazano na slici 2.6. Vrlo je jednostavan i s puno mogućnosti i vrlo prilagođeno korisniku. Isto tako ima mogućnost sinkronizacije s Google Kalendarom i Microsoft Kalendarom. Iako odvojen od tih velikih korporacija, ima njihovu podršku i zato je bio nominiran kao jedan od najboljih web kalendara na Internetu. [5]



Sl. 2.6 Tweek Calendar

2.5 Thunderbird Calendar

Posljednje rješenje nije aplikacija koja je specifična za planiranje događaja. Thunderbird je besplatni i *open-source* servis e-pošte koji osim elektroničkog poštanskog sandučića sadrži web-chat i klijentski kalendar. Ovaj kalendar je jednostavna verzija svih ovih gore rješenja, vidljivo na slici 2.7, jer je najjednostavniji, ali obavlja svoj posao brzo i jednostavno bez dodatnih plaćanja.[6]



Sl. 2.6 Thunderbird Calendar

3. TEHNOLOGIJE IZRADE RADA I RAZVOJNI ALATI

3.1. HTML

Hyper Text Markup Language (HTML) je strukturni jezik pomoću kojega je većina web stranica i web aplikacija kreirano. HTML-om se strukturira izgled stranice, odnosno sadržaj stranice. Iako većina novih web aplikacija danas ne koristi direktno pisanje HTML koda, odlično je poznavati kako HTML funkcionira i kako se strukturira HTML stranica. [7] Tijekom izrade ovog završnog rada, HTML nije ručno pisan, nego je automatsko generiran, kao što je objašnjeno kasnije.

3.2. CSS i Sass

CSS je stilski jezik i kratica za Cascading Style Sheets, CSS-om se opisuje izgled i sadržaj HTML dokumenta, odnosno CSS-om se izrađuje, korisnički primamljiv, izgled web aplikacije. CSS datoteka se dodaje u glavi HTML dokumenta pomoću `<link>` tagova. Nakon toga, pravilnim opisom u CSS datoteci može se izmjenjivati cijeli izgled HTML dokumenta. [8]

Sass je pred procesorski stilski jezik koji se kompilira u CSS datoteku pri učitavanju HTML dokumenta na server. U web zajednici Sass je prozvan *CSS-om na steroidima* zbog svojih mogućnosti korištenja varijabli i ugnježđivanja HTML elemenata jednih u druge bez nastanka.

Sass pomaže programeru u stiliziranju web aplikacije bez ponavljanja istih linija koda. Osim toga, Sass pruža izradu parcijalnih CSS datoteka koje u proizvodnji neće biti vidljive, a one mogu sadržavati varijable i funkcije korištene u ostatku web aplikacije. [9]

U ovom završnom radu korišten je Sass, on je olakšao pristup dizajniranju izgleda komponenti te je bilo lakše fokusirati se na funkcionalnosti, a ne na sami izgled.

3.3 JavaScript

JavaScript je objektno-orijentirani programski jezik koji koristi JIT (just-in-time) kompilaciju, ali je na internetu poznat kao skriptni jezik za web stranice. JavaScript je jezgra ovog završnog rada, te se sve svodilo na izradu komponenti korištenjem JavaScript biblioteka.

Mogućnost korištenja ogromne biblioteke modula je jedan od glavnih razloga zašto JavaScript ima veliku popularnost. Javascript biblioteka omogućava ponovno korištenje već kreiranih elemenata i modula bez ponovnog stvaranja istih.[10]

3.4 NPM

Kada nastane potreba za vanjskim JavaScript paketima, odnosno modulima koji nisu iz osnovne biblioteke JavaScript-a, tada se koristi Node package manager, odnosno *npm.* Node package manager se poziva u konzoli sustava ili terminalu code editora. Sastoji se od tri dijela: web mjesto, Command Line Interface i registar. Web mjesto je lokacija gdje se nalaze svi paketi i moduli dostupni kroz *npm*. Command Line Interface je sučelje koje se pokreće putem konzole te registar iz kojeg CLI dobavlja pakete s web mjesta. [15]

Tijekom izrade ovog završnog rada, *npm* je korišten u više situacija, Prvo korišten za kreiranje projekta, a onda za dodavanje kalendara i drugih korisnih biblioteka.

3.5 ReactJS

ReactJS je JavaScript biblioteka za kreiranje korisničkih sučelja, odnosno kreiranje komponenti koje mogu biti više puta upotrjebljene. ReactJS istovremeno kreira i HTML i JavaScript kod [11] tako što koristi funkcijski stil pisanja komponente i korištenjem JSX-a. JSX je proširenje sintakse JavaScript-a kojim je omogućeno pisanje koda sličnom HTML kodu unutar JavaScript datoteke. Na taj se način izbjegava pisanje običnog HTML-a te se taj kratak JSX kod može rekreirati pomoću rekreiranja komponente ReactJS biblioteke. [13]

Cijelo korisničko sučelje aplikacije ovog završnog rada je kreirano pomoću ReactJS-a, Kreiranje ReactJS projekta je opisano u idućem poglavlju.

3.6 Firebase

Firebase je skup alata za izgradnju, poboljšavanje i razvoj aplikacija i pokriva veliki dio usluga koje bi programeri inače morali sami izraditi. Na primjer, to uključuje analitiku, autentifikaciju, baze podataka, pohranu datoteka, konfiguracija aplikacije i slično. Sve ove usluge su smještene na oblaku i skaliraju se uz malo ili nimalo truda programera.

Firebase Authentication vodi računa o autentifikaciji korisnika te pruža više načina komunikacije između korisnika i servera, gdje svaki registrirani korisnik vidi samo svoje podatke spremljene unutar oblaka.

Firebase Realtime Database je NoSQL baza podataka na oblaku, koja kao što joj i ime kaže, nema tabličnu niti relacijsku strukturu. Ovakve baze koriste ključ-sadržaj relaciju, kojom server ili

backend može intuitivno obavljati procese. Iz tog razloga, Firebase omogućava programeru brzo i jednostavno rješenje za kreiranje i korištenje baza podataka. [14]

Za izradu aplikacije ovog završnog rada, postojala je potreba za bazom podataka i autentifikacijom korisnika, za ovaj rad su zato korišteni ovi Firebase alati

3.7 Visual Studio Code

Visual studio code (VSC) je Microsoftovo razvojno okruženje koje je dostupno na svim velikim računalnim sustavima, kao što su Windows, Linux ili MacOS. VSC ima mnoštvo proširenja kojima olakšava razvoj web aplikacija, ima unutarnju konzolu kojom se mogu pozivati skripte ili dodavati git održavanje aplikacije. Najčešći je izbor za razvoj web aplikacija i te je zato i odabran za razvoj ove web aplikacije za planiranje događaja. [15]

Cijela aplikacija ovog završnog rada kreirana je unutar Visual Studio Code-a, te su njegova proširenja jako pomogla pri izradi aplikacije. Proširenja kao što je, Prettier, ovo proširenje omogućava bolje formatiranje JavaScript koda te je programeru tada jasnija struktura neke ReactJS komponente.[21] Proširenje ESLint, programeru pomaže pri provjeri sintakse JavaScript-a datoteka.[22]

4. RAZVOJ DIZAJNA I FUNKCIONALNOSTI APLIKACIJE

Prije početka izrade web aplikacije, potrebno je odabrati programski jezik i razvojno okruženje za izradu aplikacije. Za izradu web aplikacije, najbolji programski jezik je JavaScript odnosno razvojno okruženje Visual Studio Code, kasnije u tekstu pisano kraticom VSC.

4.1 Kreiranje ReactJS aplikacije

Kreiranje ReactJS web aplikacije ima par bitnih koraka. Kreiranje projekta započinje node installerom *npx*. Za početak, određivanje imena projekta mora biti podržano u node installeru npx. Node installer ne dozvoljava razmake i velika slova, na primjer ime projekta: my-app. Zatim, u željenoj datoteci na sustavu unutar konzole sustava poziva se komandna linija 1 sa slike 4.1. Nakon nje ulazak u projektnu mapu obavlja se linijom 2 sa slike 4.1 te se ReactJS aplikacija pokreće komandnom linijom 4 sa slike 1.1.

Linija	***Kod***
1:	`npx create-react-app my-app`
2:	`cd my-app`
3:	`code .`
4:	`npm start`

Sl. 4.1. Kod za kreiranje ReactJS aplikacije

Kako bi uređivanje izgleda i funkcionalnosti ReactJS web aplikacije bilo moguće, otvaranje projekta u Visual Studio Code-u se izvršava komandnom linijom 3 sa slike 4.1. [17] U izradi ovog završnog rada korišten je kod sa slike 4.1 i jedina izmjena je bila ime projekta koji je bio *zavrsniv1.0.* Nakon prvog kreiranja potreba za ovim linijama nije bila potrebna, jedino što je bilo potrebno su komandne linije 2 i 3 sa slike 4.1. Linijom 4 sa slike 4.1 se pokreće projekt na pregledniku.

U terminalu VSC-a nakon pokretanja projekta potrebno je dodati pakete koji će biti od pomoći u dizajnu i funkcionalnosti projekta. Dodavanjem paketa *react-router-dom* omogućena je funkcija izmjena putanja aplikacije. Dodavanje ovih paketa je prikazano prvim retkom slike 4.2. ReactJS aplikacija ima jednu stranicu, paket *react-router-dom*, otvara mogućnost dodavanja više stranica na ReactJS aplikaciju.[18] Drugo dodavanje potrebno za projekt je bio izvor ikonica za gumbove

unutar aplikacije. Stoga dodavanje *bootstrap-icons* paketa, kao što je prikazano na drugoj liniji sa slike 4.2, omogućuje izbor preko 1600 različitih ikonica za korištenje unutar aplikacije. [19]

Linija	***Kod***
1:	`npm i react-router-dom`
2:	`npm i bootstrap-icons`

Sl. 4.2. Kod za dodavanje dodatnih paketa u projekt

4.2 Izrada strukture projekta

Kreiranjem ReactJS projekta i dodavanjem dodatnih paketa, započinje izrada strukture projekta. Struktura projekta sadrži navigacijsku traku i sadržaj trenutne putanje dane putem *react-routera,* prikazan kodom sa slike 4.3. Glavna putanja označava se sa znakom `/`, odnosno praznom putanjom. Ova aplikacija sadrži planove na glavnoj putanji, a druga putanja dostupna na navigacijskoj traci je izmjenjiva. Ako korisnik nije trenutno prijavljen, onda ta putanja vodi na registracijski obrazac, koji se nalazi na putanji `/sign-in`. Na ovoj stranici korisnik ima jednostavan obrazac za prijavu korisnika. U suprotnom, ako je korisnik prijavljen, putanja je `/profile` te vodi na njegov profil gdje može vidjet svoju e-mail adresu i ima opciju odjave iz aplikacije.

Linija	***Kod***
1:	`<Routes>`
2:	`<Route path="/" element={<MyNotes/>}/>`
3:	`<Route path="/sign-in" element={<SignIn/>}/>`
4:	`<Route path="/profile" element={<Profile/>}/>`
5:	`</Routes>`

Sl. 4.3. Kod za odabir putanja

Navigacija ima svoju zasebnu komponentu i treba biti prikazana na svim putanjama aplikacije. Kako bi se to omogućilo kreira se `Layout` komponenta čiji je dio koda prikazan na slici 4.4. Ova komponenta sadrži komponentu `MainNavigation` jer je potrebno da navigacija bude uvijek vidljiva te unutar `<main>` taga sadrži sve ostale komponente stranice, kombinirane u jedan element zvan `children`.

Props je atribut koji sadrži sve atribute koje komponenta može primiti pri kreiranju. Kako bi props sadržavao sve putanje, potrebno je `Layout` omotati oko `Routes` kao što je prikazano na slici 4.5. Sada će sav sadržaj određene rute biti vidljiv kao sadržaj `Layout` komponente te će istovremeno biti vidljiva i navigacijska traka.

```
Linija   Kod
1:      function Layout(props){
2:        return(
3:          <div>
4:            <MainNavigation/>
5:            <main>{props.children}</main>
6:          </div>
7:        );
8:      }
```

Sl. 4.4. Kod za Layout komponentu

```
Linija   Kod
1.      <Layout>
2.        <Routes>
3.          <Route path="/" element={<MyNotes/>}/>
4.          <Route path="/sign-in" element={<SignIn/>}/>
5.          <Route path="/profile" element={<Profile/>}/>
6.        </Routes>
7.      </Layout>
```

Sl. 4.5. Kod omotavanja Layout komponente na Routes komponentu

Izradu glavne stranice aplikacije započinjem dodavanjem elemenata unutar `MyNotes` komponente. Ova komponenta sadržavat će kalendar i planove odabranog dana. Pri prvom učitavanju bit će prikazani planova za trenutni dan. Kalendar neće biti vidljiv sve dok se ne klikne gumb za kalendar. Način prikaza bilješki opisan je u slijedećem poglavlju.

4.3 Kreiranje komponente bilješke

Kreiranje bilješke kao komponente ima dva koraka. Kreiranje liste bilješki kako bi u unutar komponente `MyNotes` bile prikazane pravilno. Ovu komponenta će se zvati `NoteList` i ona će primati, kroz props, popis bilješki za određeni datum iz baze podataka. Također sadržavat će gumb za kreiranje nove bilješke. Kreiranje nove bilješke odvija se unutar `Popup` komponente. `Popup`

komponenta se pomoću CSS-a stvara preko sadržaja cijele aplikacije i unutar nje sadržaj može biti naknadno zadan. To znači da je `Popup` komponentu moguće na više mjesta iskoristiti s različitim sadržajima. Unutar ove `Popup` komponente je jednostavan obrazac za unos podataka popunjavaju se podatci o novoj bilješci te se dodaju na popis bilješki za taj dan. Drugi korak je kreirati komponentu `Note`. Ova komponenta će imati svojstvo ponovnog korištenja te će dodavanje više istih komponenti u listu bilješki biti lagano. Bilješka ima naslov i opis bilješke koji su tipa `string` te prioritet odrade koji ima brojčani tip. Izgled komponente Note je kao kvadratni oblak koji sadrži naslov bilješke. Prilikom prelaska miša preko bilješke vidljiv je i njezin opis.

Prikazivanje dodatnih podataka o bilješci obavljat će se klikom na bilješku. Ovdje je također upotrjebljena `Popup` komponenta, ali ovaj put sadržaj komponente su naslov i opis bilješke te prioritet bilješke. Osim toga, dodani su gumbovi za brisanje i uređivanje bilješke a kod je prikazan na slici 4.6. Brisanjem bilješke, ona se briše s popisa bilješki te ju više nije moguće vratiti. Klikom na uređivanje bilješke prikazan je sličan obrazac kao pri kreiranju bilješke, samo je ovaj put već popunjen i slobodno je uređivanje polja. Potvrđivanjem uređivanja ažurira se bilješka te je odmah vidljiva promjena na `NoteList` komponenti.

```
Linija   Kod
1:      <Popup trigger={buttonPopup} setTrigger={setButtonPopup}>
2:        <div>
3:          <h2>{title}</h2>
4:          <p>Priority: {priority}</p>
5:          <p>{description}</p>
6:          <div className={classes.buttonHolder}>
7:            <button
8:               className={classes.button}
9:               onClick={() => {
10:              setUpdatePopup(true);
11:              setButtonPopup(false);
12:              }}
13:           >
14:           Edit
15:           </button>
16:           <button className={classes.button} onClick={onDeleteHandler}>
17:           Delete
18:           </button>
19:       </div>
20:       <br />
21:     </div>
22:   </Popup>
```

Sl. 4.6. Kod za prikaz informacija o bilješki

4.4 Dodavanje komponente kalendara

Komponenta kalendara nije stvorena od nule. Nego ju ReactJS zajednica ima dostupnu javno preko *npm-a.* Dodavanje ovog kalendara te njegovo korištenje se izvodi kao na slici 4.7. Prve dvije linije slike 4.7 prikazuju instaliranje modula, a druge dvije linije korištenje tog kalendara unutar projekta. Klikom na bilo koji datum na kalendaru, poziva se `onChange` metoda, ova metoda promjeni `value` na odabrani datum. S ovim kodom omogućen je unos novih bilješki na odabrani dan. [20]

```
Linija   Kod
1.      //pozivanje u terminal
2.      npm i react-calendar
3.      //unutar MyNotes.js datoteke
4.      <Calendar onChange={onChangeDate} value={value}/>
```

Sl. 4.7. Kod za instaliranje i korištenje ReactJS Calendar-a

Komponenta kalendara ima varijablu `value` koja je tipa `Date` iz tog razloga ima funkciju `toString()` kojom datum iz tipa `Date` pretvaramo u tip `string`. String verziju ove varijable možemo koristiti za prikaz datuma na stranici te u budućnosti za spremanje bilješke na datum određen u bazi podataka.

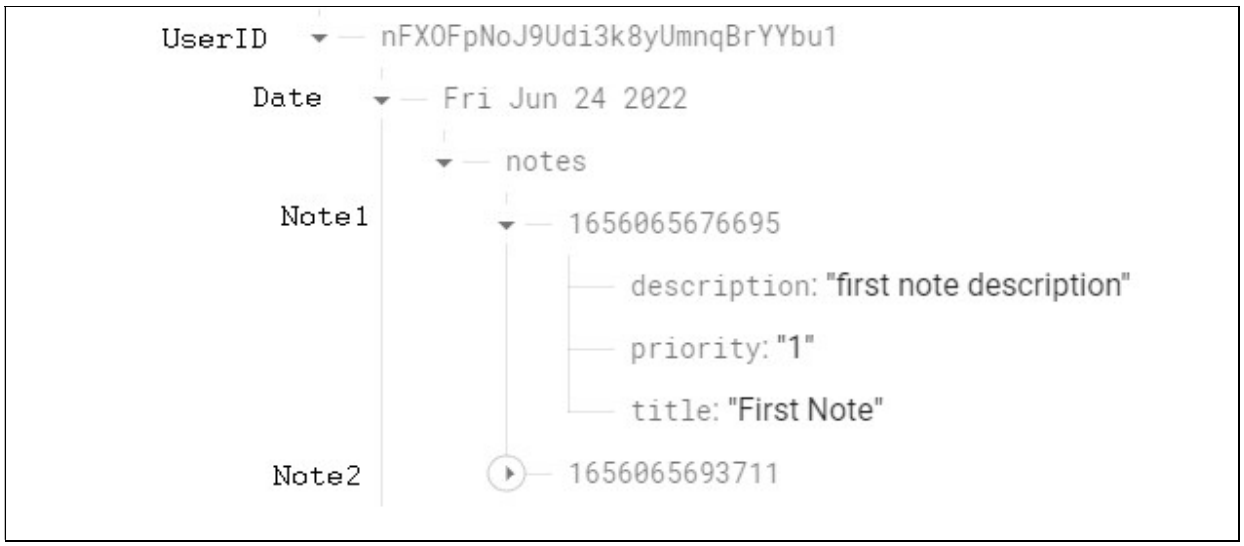
4.5 Firebase baza podataka i autentifikacija

Firebase omogućava kreiranje NoSQL baze podataka koja je pogodna za spremanje podataka ključ-vrijednost, ali prvo je potrebno kreirati autentifikaciju korisnika. Firebase ima mogućnost autentifikacije korisnika metodom `signInWithEmailAndPassword`, ako korisnik postoji onda je prijavljen. Ako korisnik ne postoji, onda postoji metoda `createUserWithEmailAndPassword`. Ova metoda kao parametre e-mail i zaporku korisnika te ih dodaje u popis registriranih korisnika, koji nakon što je dodan može korištenjem `signInWithEmailAndPassword` metode se prijaviti. Prijavljen korisnik se sprema u `AuthContext` koji se dohvaća funkcijom `useAuthValue()` prikazano kodom sa slike 4.8.

Linija	*Kod*
1.	`const AuthContext = createContext();`
2.	`export function useAuthContext(){`
3.	`  return useContext(AuthContext);`
4.	`}`

Sl. 4.9. Spremljene informacije o prijavljenom korisniku

Baza podataka ima strukturu kao na slici 4.8. `UserID` je jedinstveni ključ koji dobiva svaki registrirani korisnik te u bazi podataka svaki korisnik ima svoj odjeljak koji je pod tim ključem, pod svakim ključem su datumi koji sadržavaju barem jednu bilješku. Pri kreiranju nove bilješke dodaje se bilješka s njezinim naslovom, opisom i prioritetom na datum te u listu notes kao na slici 4.8. Dodaje se funkcijom `set`, koja prima referencu na lokaciju spremanja i što treba spremiti, u ovom slučaju odabrani datum i novu bilješku. Ako ne postoji bilješka na datumu koji je odabran, kreirat će se grana tog datuma, a ako ipak postoji već neka bilješka na odabranom datumu, ona će se dodati na isti datum u obliku liste. Uređivanje bilješke se isto obavlja funkcijom `set`, samo se pri uređivanju mijenjaju samo polja koja se izmjenjuju. Također, brisanjem bilješki u aplikaciji brišu se bilješke u bazi podataka. To je odrađeno funkcijom `remove`. Funkcija `remove` prima referencu na lokaciju bilješke koju treba obrisati iz baze podataka.



Sl. 4.9. Struktura NoSQL baze podataka na Firebase-u

5. OPIS RADA WEB APLIKACIJE

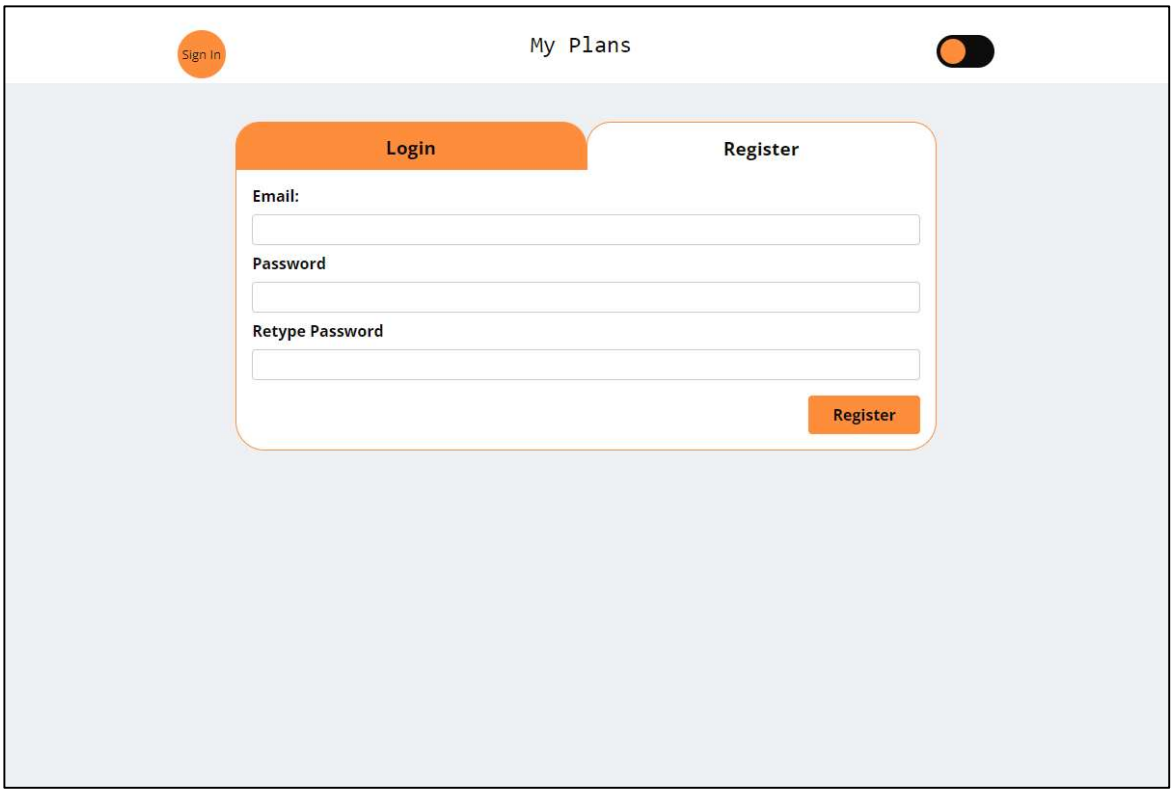
Korištenje funkcionalnosti razrađenih u prijašnjim poglavljima, u ovom poglavlju će biti objašnjene. Prikaz rada web aplikacije će biti objašnjen korak po korak od registracije do izrade novog plana.

5.1 Registracija korisnika

Učitavanjem web aplikacije, bilo kroz `npm start` ili kroz poveznicu u prilogu, korisnik ima priliku vidjet početnu stranicu aplikacije. Korisnik nije prijavljen te na početnoj stranici aplikacije bit će poruka koja obavještava korisnika da za korištenje aplikacije treba biti prijavljen, vidljivo na slici 5.1. Prvi korak za prijavu odnosno registraciju je kliknuti gumb na kojem piše *Sign In.*

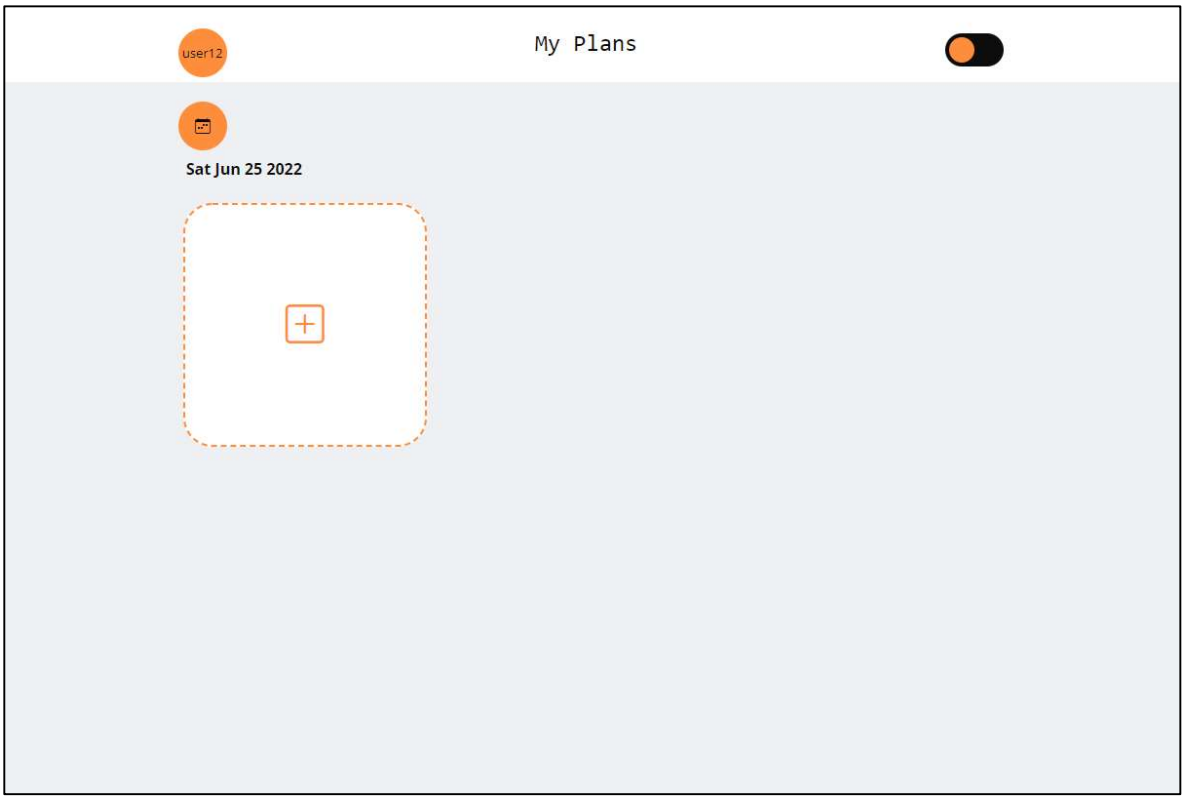
Sl. 5.1 Početni zaslon aplikacije kada korisnik nije prijavljen

Na novo učitanom prozoru, kao na slici 5.2, pojavit će se izbor između prijave postojećeg korisnika (Login) i prijave novog korisnika (Register).



Sl. 5.2 Registracijski zaslon aplikacije

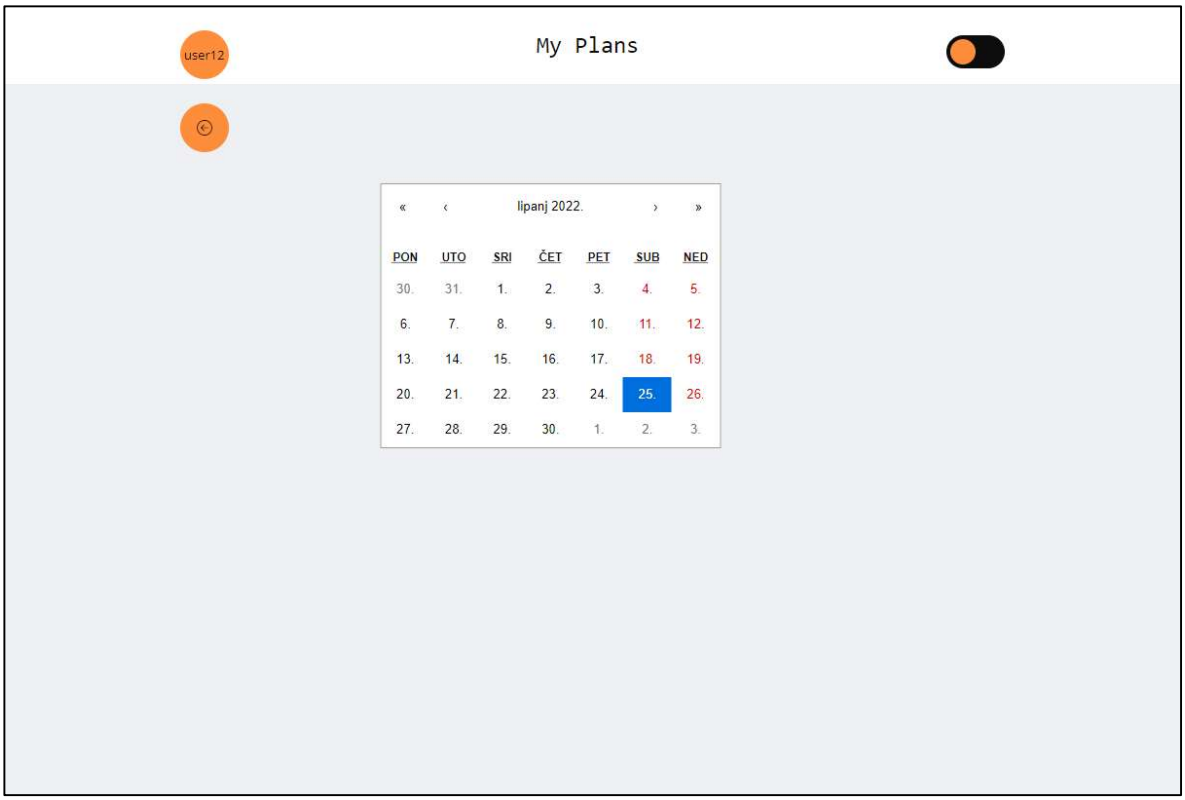
Unošenjem e-mail-a i zaporke korisnik ispunjava obrazac za prijavu, korisnik treba pritisnuti gumb za prijavu i ako je prijava uspjela, korisnik će biti proslijeđen na novi zaslon gdje će mu biti učitana lista današnjih planova (primjer na slici 5.3.). Ako prijava nije uspjela korisnik će dobiti kratku poruku s informacijom da mu prijava nije uspjela te uz to razlog zašto nije prijava uspjela. Jedan od razloga neuspjele prijave je da nije u listi korisnika ta kombinacija za prijavu, a za registraciju neuspjela prijava je da uneseni e-mail već postoji u listi korisnika ili da je lozinka prekratka.



Sl. 5.3 Početni zaslon aplikacije nakon prijave korisnika

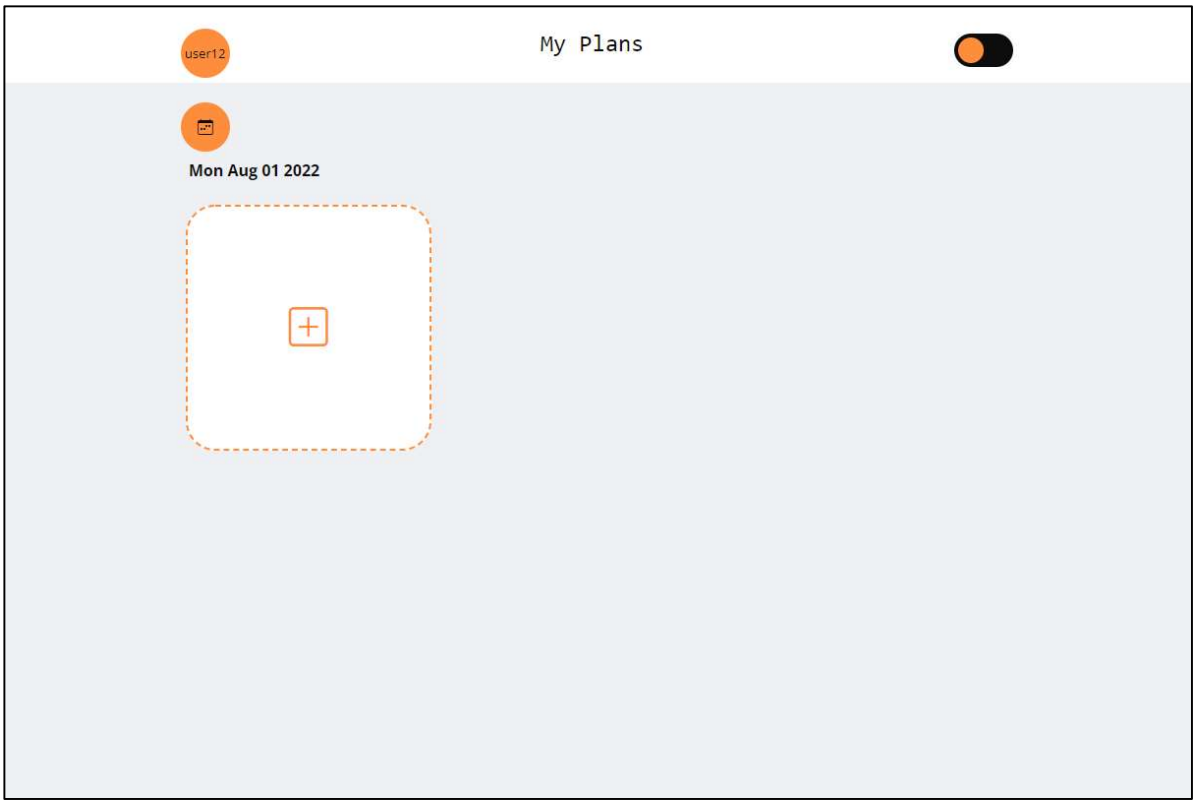
5.2 Odabir dana na kalendaru

Funkcionalnost odabira dana se obavlja klikom na ikonu kalendara koja se nalazi na početnom zaslonu aplikacije, vidljivo na slici 5.3. Nakon klika otvara se zaslon s kalendarom, kao na slici 5.4.



Sl. 5.4 Zaslon s kalendarom

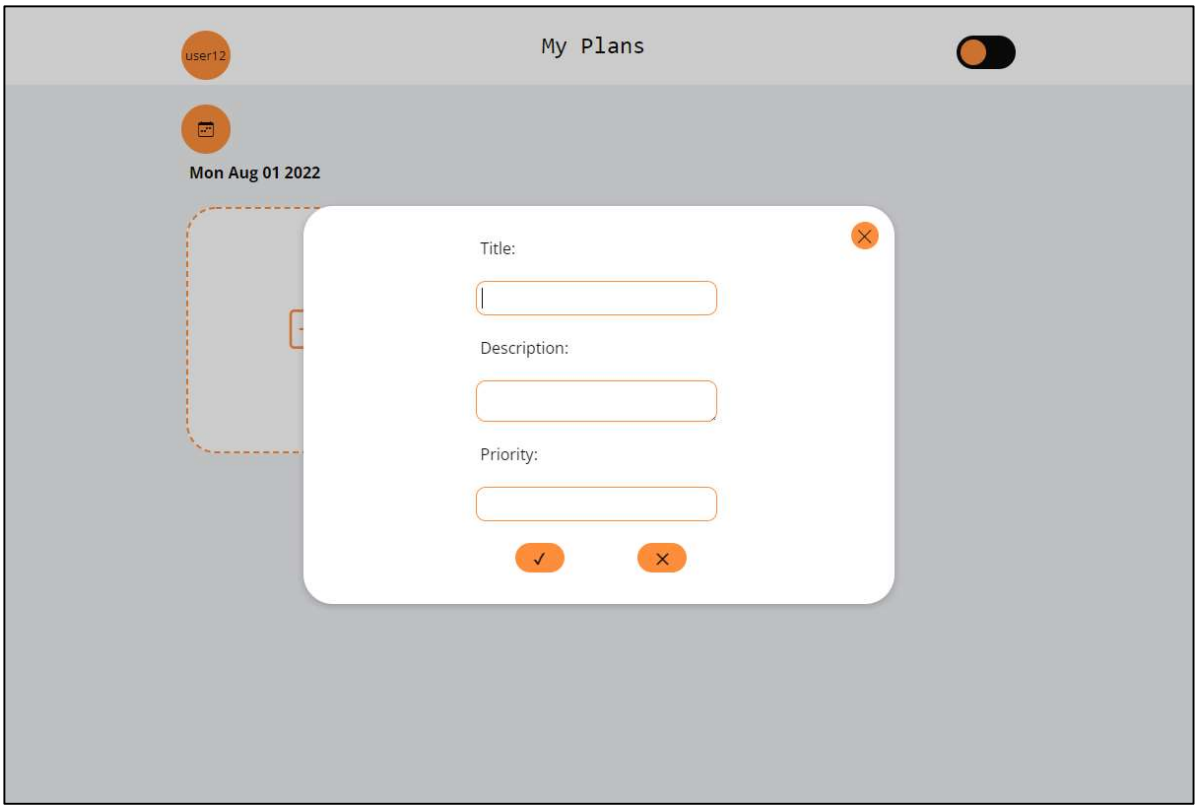
Odabirom željenog dana zaslon se odmah prebacuje na početni zaslon, gdje je sada vidljiv odabrani datum i zabilježeni planovi za taj datum. U primjeru na slici 5.5. odabran je datum 1. kolovoza 2022.



Sl. 5.5 Početni zaslon za dan 1. kolovoza 2022.

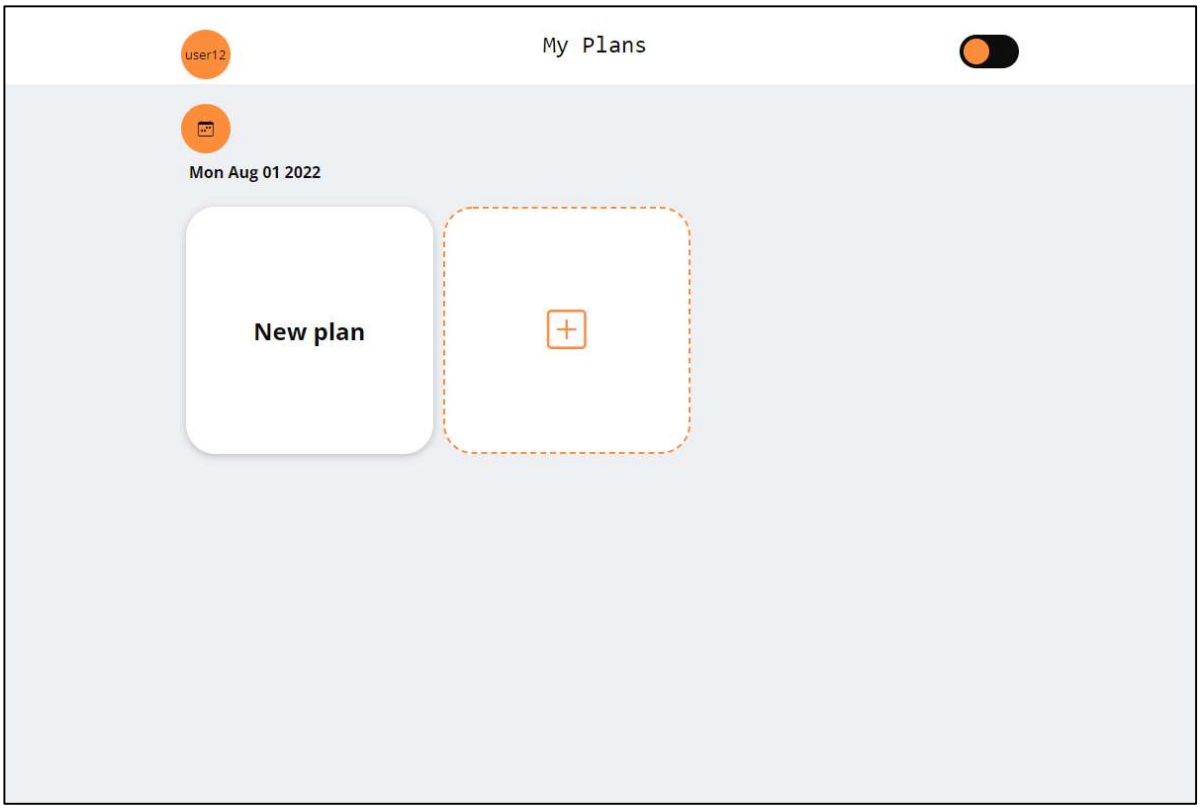
5.3 Dodavanje novog plana

Nakon odabira željenog dana, sada korisnik može, klikom na gumb sa znakom +, dodati novi plan. Klikom na ovaj gumb, korisnik na ekranu dobiva obrazac, kao na slici 5.6, koji je potrebno ispuniti kako bi plan bio kreiran. Korisnik može odustati od kreiranja plana klikom na X ili bilo gdje izvan ekrana obrasca.



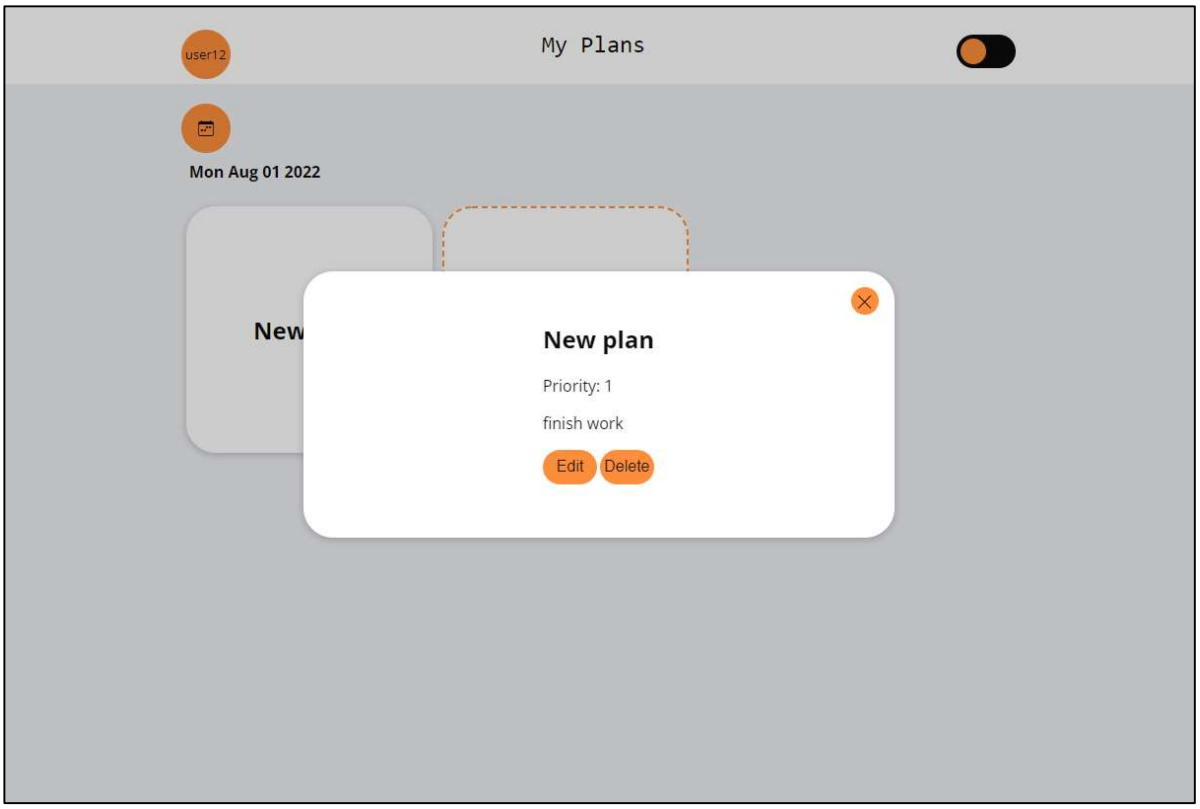
Sl. 5.6 Obrazac za dodavanje nove bilješke.

Nakon unosa podataka o planu, kao što su naslov i opis, korisnik, klikom na kvačicu, potvrđuje unos plana te isti odmah bude vidljiv na početnom zaslonu aplikacije, kao na slici 5.6. Nakon toga, korisnik može preći mišem preko plana kako bi vidio više informacija o planu ili kliknuti na plan.



Sl. 5.6 Početni zaslon aplikacije nakon dodavanja novog plana

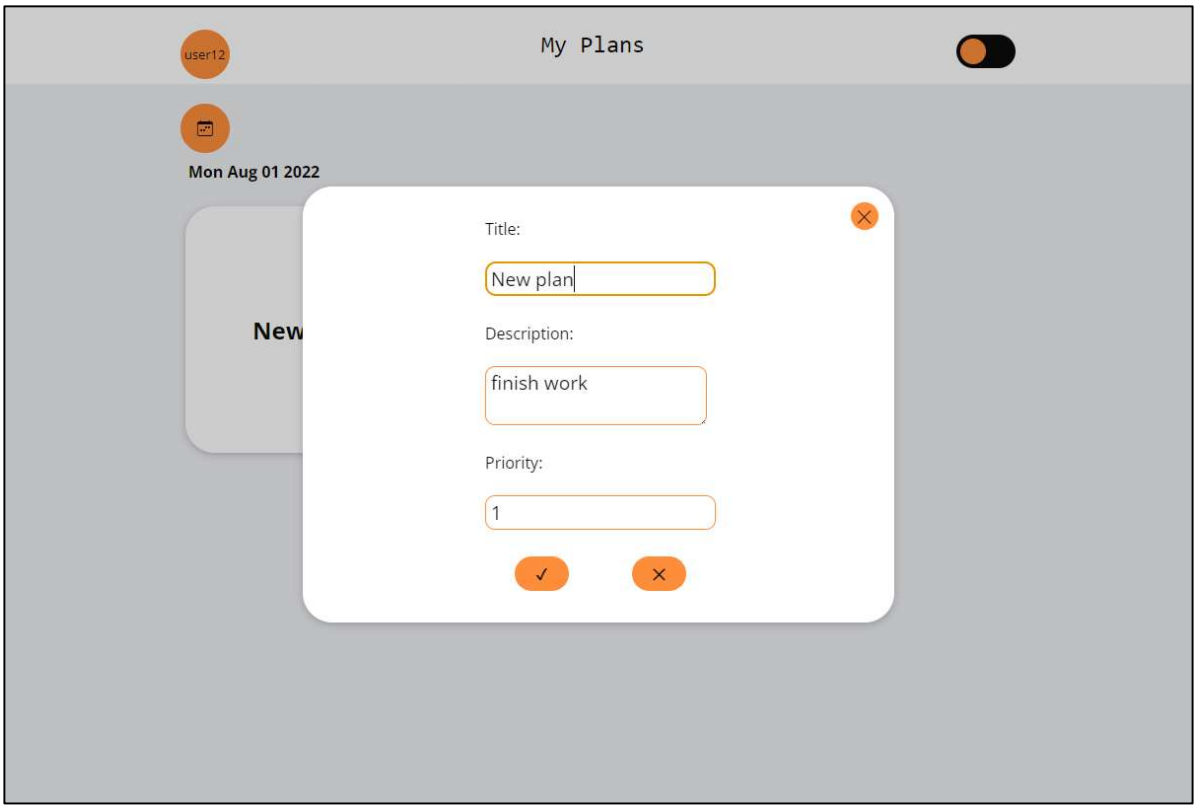
Klikom na plan, korisnik može vidjeti sve informacije o planu i vidjeti opcije uređivanja i brisanja plana, kao što je prikazano na slici 5.7.



Sl. 5.7 Popup ekran s informacijama plana

5.4 Uređivanje postojećeg plana

Uređivanje plana obavlja se klikom na gumb *Edit*, vidljiv na slici 5.7. Nakon klika na gumb Edit, otvara se novi prozor s informacijama plana, ali sada informacije imaju omogućenu izmjenu, kao na slici 5.8. Uređivanje se potvrđuje klikom kvačicu ili ako korisnik ne želi obaviti izmjene može klikom na X odustati od promjena. Promjene se obavljaju odmah nakon klika na kvačicu.



Sl. 5.8 Popup ekran s informacijama plana koje je moguće uređivati

Nakon klika na kvačicu, ekran se vraća na početni ekran te je plan vidljivo promijenjen. Nekada nije dovoljno urediti plan, nego ga je potrebno obrisati kako bi se rekreirao od nule.

5.5 Brisanje postojećih planova

Plan je moguće obrisati kroz ekran gdje su vidljive sve informacije o planu, kao na slici 5.7. Brisanje plana je nepovratno te se plan iste sekunde briše i lokalno i na serveru, odnosno u bazi podataka.

6. ZAKLJUČAK

Kreiranje web aplikacije, uz trenutni napredak tehnologije, vrlo je jednostavno. Uz poznavanje područja rada i programskog jezika, programer vrlo brzo dolazi do finalnog produkta kao što je aplikacija ovog završnog rada. Ono što zadaje probleme je popravljanje pogrešaka i takozvanih *bug-ova.* Svaki objekt i svaki tekst mora proći kroz testiranje kako pri objavljivanju aplikacije na tržište korisnici ne bi imali problema.

Kako bi izrada web aplikacije bila moguća potrebno je proučiti novu tehnologiju kao što su NPM i ReactJS. Node package manager programeru daje na izbor veliku biblioteku JavaScript paketa te istovremeno brzo preuzimanje istih. Jedan od tih paketa je ReactJS koji korištenjem funkcijskih komponenti kreira cijelu strukturu web aplikacije. ReactJS komponente olakšavaju kreiranje interaktivnog i korisniku intuitivnog sučelja koje je vrlo lagano rekreirati te ponovno iskoristiti na idućem projektu.

Osim učenja novih tehnologija, poznavanje već poznatih tehnologija pomaže programeru pri izradi urednog koda koristeći Visual Studio Code koji omogućava održavanje koda s mnoštvom izbornika i integriranom konzolom.

LITERATURA

[1] Funkcionalnosti Google Calendar-a, dostupne na:
https://support.google.com/a/users/answer/9247501, [21.6.2022.]

[2] Funkcionalnosti Google Keep-a, dostupne na:

https://www.google.com/keep/, [9.7.2022.]

[3]Funkcionalnosti Microsoft Planner-a, dostupne na:

https://support.microsoft.com/en-us/office/when-to-use-microsoft-project-planner-to-do-or-the-tasks-app-in-teams-8f950d32-d5f4-40db-a8b7-4d1b82b55e17 , [21.6.2022.]

[4] Funkcionalnosti Tweek Calendar-a, dostupne na:

https://tweek.so/calendar/help, [21.6.2022.]

[5] Funkcionalnosti Calendar-a, dostupne na:

https://www.calendar.com/product/, [21.6.2022.]

[6] Funkcionalnosti Thunderbird-a, dostupne na:
https://www.thunderbird.net/en-US/about/, [22.6.2022.]

[7] HTML, dostupno na:
https://html.com/, [23.6.2022.]

[8] Dokumentacija CSS-a, dostupna na:

https://developer.mozilla.org/en-US/docs/Web/CSS, [23.6.2022.]

[9] Dokumentacija Sass-a, dostupna na:

https://sass-lang.com/documentation/, [23.6.2022.]

[10] Dokumentacija JavaScript-a, dostupna na:

https://developer.mozilla.org/en-US/docs/Web/JavaScript, [23.6.2022.]

[11] K. Peterson, „React js: The Ultimate Beginner's Guide to Learn React js Programming Step by Step – 2020“, mEm lnc

[12] Dokumentacija ReactJS-a, dostupna na:

https://reactjs.org/, [23.6.2022.]

[13] Korištenje JSX-a, dostupno na:

https://beta.reactjs.org/learn/writing-markup-with-jsx, [23.6.2022.]

[14] What is Firebase?, dostupno na:

https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0, [23.6.2022.]

[15] Visual Studio Code, dostupno na:

https://code.visualstudio.com/docs, [23.6.2022.]

[16] NPM registar, dostupno na:

https://docs.npmjs.com/about-npm, [24.6.2022.]

[17] Create a New React App, dostupno na:

https://reactjs.org/docs/create-a-new-react-app.html, [24.6.2022.]

[18] react-router-dom, dostupno na:

https://www.npmjs.com/package/react-router-dom, [24.6.2022.]

[19] Bootstrap icons, dostupne na:

https://icons.getbootstrap.com/, [24.6.2022.]

[20] React-calendar, dostupan na:

https://www.npmjs.com/package/react-calendar, [24.6.2022.]

[21] Prettier, dostupan na:

https://marketplace.visualstudio.com/items?itemName=esbenp.prettier-vscode, [9.7.2022.]

[22] ESLint, dostupan na:

https://eslint.org/ , [9.7.2022.]

SAŽETAK

U ovom radu opisan je tijek izrade web aplikacije za planiranje događaja te su opisane i potrebne funkcionalnosti takve aplikacije. Korišteni su opisni, stilski i objektno orijentirani jezici. Prikazana je usporedba s već postojećim rješenjima te kako su ona utjecala na izradu ovog rada. Objašnjen je tijek izrade web aplikacije od nule do finalnog produkta u svim potrebnim koracima. Prikazane su funkcionalnosti aplikacije korak po korak od registracije do izrade plana.

Ključne riječi: kalendar, organizacija, plan, web aplikacija

ABSTRACT

Web application for planning personal events

This final thesis describes the development of a web application for event planning, in addition, the required functionalities of such an application are described. Descriptive, stylistic, and object-oriented programing languages were used in the creation of this application. A comparison of already known solutions are shown, and these had an impact in development. The flow of development of the web application from scratch to final product is shown in steps. Created a walkthrough of all functionalities from the registration to creation of an event.

Key words: calendar, organization , event , web application

PRILOZI

Prilog 1. Pismena verzija završnog rada u .docx formatu

Prilog 2. Pismena verzija završnog rada u .pdf

Prilog 3. Cijela se aplikacija nalazi na github-u, koji je na poveznici:

https://github.com/Zokky2e/zavrsniv1.0

Prilog 4. Otvorena verzija aplikacije na poveznici:

https://zavrsni-3652b.web.app/