



УНИВЕРЗИТЕТ У НОВОМ САДУ
**ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ**



Душан Ивић, ПР160/2019

Апликација за доставу хране

ПРОЈЕКАТ

- Примењено софтверско инжењерство (ОАС) -

Нови Сад, септембар 2023.

САДРЖАЈ

1. ОПИС РЕШАВАНОГ ПРОБЛЕМА
2. ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА
3. ОПИС РЕШЕЊА ПРОБЛЕМА
4. ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА
5. ЛИТЕРАТУРА

ОПИС РЕШАВАНОГ ПРОБЛЕМА

Апликације за доставу хране су постале неизоставан део савременог начина исхране. Ове платформе су промениле начин на који наручујемо, припремамо и конзумирамо храну. Корисницима је омогућено да бирају између разноврсних ресторана и јела, без обзира на којој локацији се налазили, а након обављања поруџбине целокупан процес припреме хране и њене евентуалне доставе остаје као обавеза ресторана.

Овакве апликације често нуде и додатан низ функционалности које олакшавају креирање и праћење поруџбина, као што су статус поруџбине, тренутна локација достављача, могућност отказивања поруџбине, обавештења о акцијама и попустима, могућност оцењивања и остављања коментара на страницама “посећених” ресторана, али и безбедан начин онлајн плаћања.

Апликације за доставу хране су у годинама иза нас постале изузетно популарне и широко коришћене у целом свету, а заслуге за раст ових платформи се свакако може приписати и пандемији^[1] која је онемогућила корисницима излазак у ресторане и приморала их да самостално припремају храну или у честима случајевима да се окрену поручивању.

Док је основни концепт поменутих апликација генерално уопштен и врло једноставан, постоји мноштво варијација и решења прилагођених различитим потребама и жељама корисника. У Србији, засигурно најпознатији пружаоци овог типа услуга су Глово и Волт, али свакако не и једини.

Разлика између наведених и мноштва осталих сервиса за поручивање и доставу хране јесте број доступних ресторана и производа. Исто тако, код познатијих платформи, због њихове велике популарности, често и не постоје трошкови доставе, због чега корисници настављају да их користе.

Имплементирано решење представља поједностављени пример апликације за доставу хране и његова главна сврха је разумевање функционалности система. Корисници ове апликације (у овом примеру купци) не морају да воде рачуна о плаћању већ одмах након креирања налога могу да почну са коришћењем платформе.

ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА

Од технологија, за имплементацију серверске апликације (*backend*) коришћен је ASP.NET Core Web API који уз помоћ Entity Framework Core радног оквира комуницира са PostgreSQL базом података. Што се тиче клијентске апликације (*frontend*), као примарна технологија је коришћен ReactJS, уз TypeScript програмски језик. За развој ове full-stack апликације коришћени су алати Visual Studio 2022 и Visual Studio Code, као и алат Insomnia за тестирање API-ја.

ASP.NET Core Web API је радни оквир за развој веб-сервиса и API-ја (*Application Programming Interface*), креиран од стране Microsofta. Овај радни оквир омогућава брз и једноставан развој REST API сервиса, као и аутоматско генерисање документације помоћу алата као што је Swagger. Такође, пружа подршку за комуникацију са великим бројем база података кроз NuGet пакете. Једна од база података подржаних од стране овог радног оквира јесте PostgreSQL, релациона база података која пружа подршку и за NoSQL (не-релационе) функционалности.

ReactJS је једна од најпопуларнијих и најшире коришћених технологија за израду веб апликација. То је JavaScript библиотека (не радни оквир!) развијена од стране Facebooka. Заснива се на компонентној архитектури, што значи да се кориснички интерфејс изграђује од појединачних делова (компонената) што даље олакшава организацију кода и реузабилност самих компонената. Популарност ReactJS-а и његова велика заједница омогућили су стварање великог броја библиотека, разних намена, које знатно олакшавају развој апликације.

У комбинацији са Reactom коришћен је програмски језик TypeScript, надскуп JavaScripta који омогућава статичку типизацију – дефинисање типова променљивих, функција и компонената, типизацију API захтева и одговора и у крајњем случају – боље разумевање кода.

За креирање саме клијентске апликације коришћен је Vite.js, популарни развојни сервер дизајниран тако да олакша и убрза развој веб апликација. Његова најзначајнија предност, поготово у односу на класичан Create React App приступ, јесте брзина. Уместо да компајлира све у један велики фајл као нпр. Webpack, Vite.js користи „esbuild“ да генерише мале и оптимизоване модуле који се могу брже учитати приликом отварања странице^[2].

За управљање глобалним стањем апликације коришћен је Redux, тачније Redux Toolkit. Redux је JavaScript библиотека за управљање стањем апликације и посебно је популарна у апликацијама развијеним помоћу Reacta. Основни појмови везани за Redux су store (вид централног складишта у апликацији), actions (акције које описују догађаје), reducers (функције које обрађују акције и ажурирају стање) и dispatch (методе за покретање акција).

Redux Toolkit је скуп алата који убрзавају и олакшавају рад са Reduxом. Умањује понављање кода аутоматским генерисањем акција и редукторских функција, али и олакшава управљање асинхроним акцијама. Такође, током апликације је у појединим ситуацијама коришћен и Local Storage – мала база података уграђена у веб претраживач која омогућава чување података у текстуалном облику без обзира на трајање сесије.

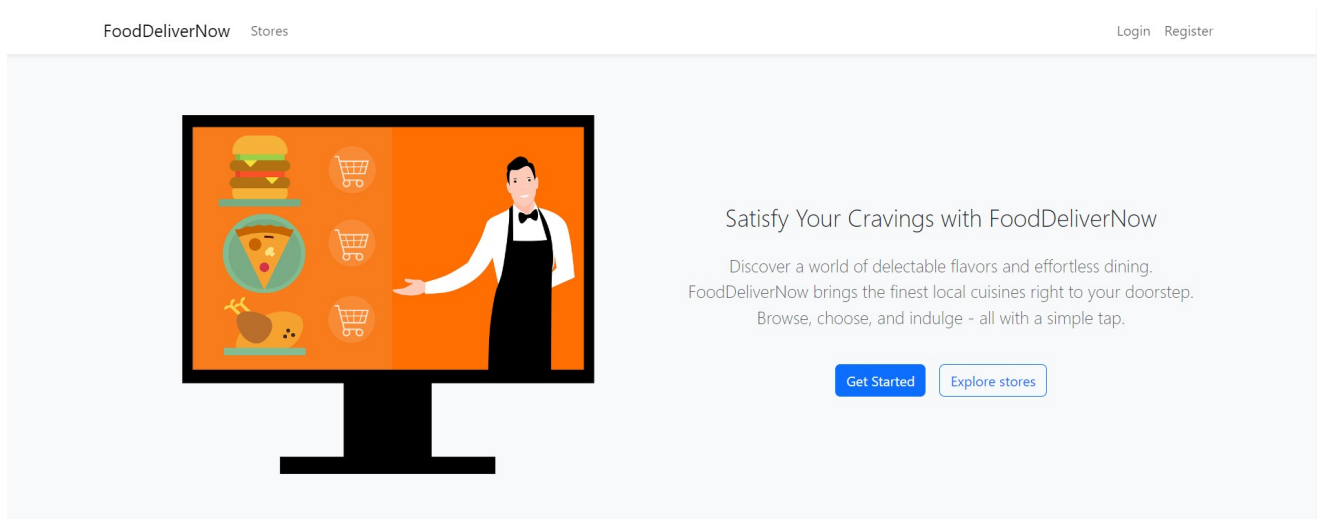
ОПИС РЕШЕЊА ПРОБЛЕМА

Имплементирани систем подржава три типа корисника: купце, партнере и админе. Админи су корисници чије је примарно задужење да верификују новорегистроване партнере, тј. могу да их прихвате или одбију кликом на одређено дугме у табели партнера, а такође имају и преглед свих регистрованих ресторана и креираних поруџбина. Партнери представљају власнике објеката који могу да пријаве, тј. региструју свој ресторан или продавницу у постојећи систем, затим управљају својим ресторанима, прегледају поруџбине и управљају доступним производима. Купци су корисници који могу да поручују производе из доступних ресторана, али и да управљају својим поруџбинама. Основне функционалности апликације су доступни у табели 1.

Ресурс	Акција	Корисник			
		Без налога	Купац	Партнер	Админ
Поруџбина	Преглед	✗	✓	✓	✓
	Креирање	✗	✓	✗	✗
	Отказивање	✗	✓	✗	✗
Ресторан	Преглед	✓	✓	✓	✓
	Додавање	✗	✗	✓	✗
	Модификовање	✗	✗	✓	✗
	Брисање	✗	✗	✗	✓
Производ	Преглед	✓	✓	✓	✓
	Додавање	✗	✗	✓	✗
	Модификовање	✗	✗	✓	✗
	Брисање	✗	✗	✓	✗
Партнер	Верификовање	✗	✗	✗	✓

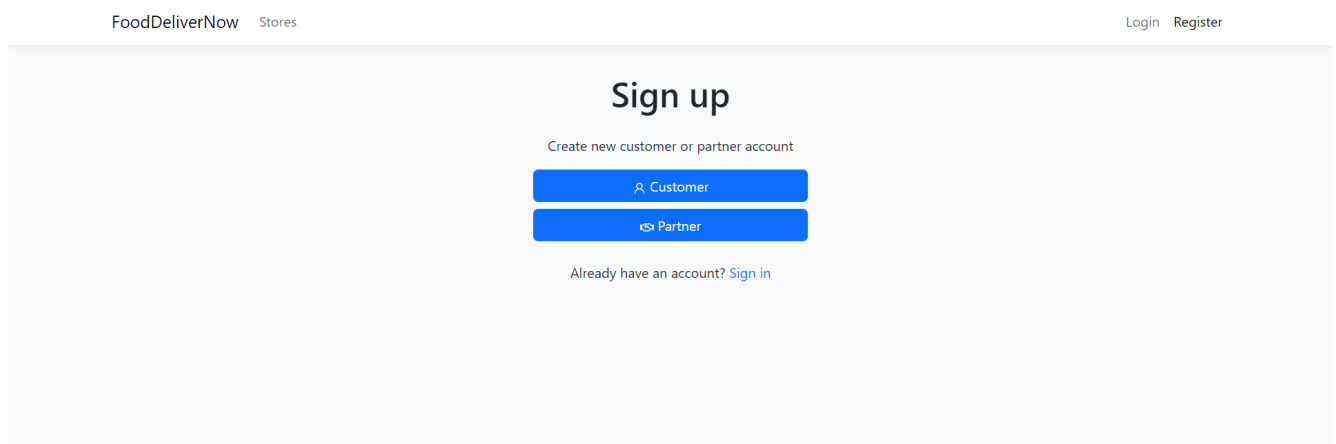
Табела 1 – Функционалности апликације

Почетна страница са којом се корисници сусрећу садржи кратак опис и упутство за коришћење платформе (слика 1). Са ове странице, у зависности од одабране опције, корисник може да пређе на страницу за пријаву на систем, односно на страницу за преглед доступних ресторана.



Слика 1

Регистрација са клијентске стране је омогућена само за прва два типа корисника, купце и партнере. Страница за регистрацију садржи два дугмета којом корисник може да изабере у коју сврху креира налог (слика 2), након чега ће му бити приказана поља за унос.



Слика 2

Овај корак регистрације је имплементиран коришћењем `useState` hook-а у Reactу који чува јединствену ознаку типа корисника и на основу тога рендерује одговарајућу форму за унос података.

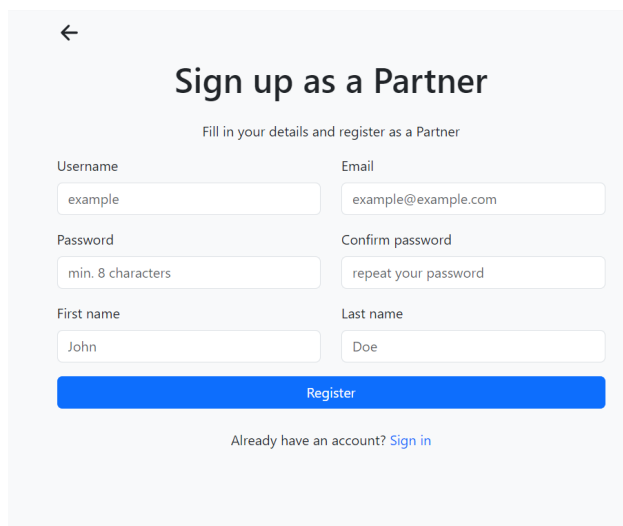
```
enum UserType { Customer = 0, Partner = 1 }
...
const [registerType, setRegisterType] = useState<UserType | null>(null);
...
<Button onClick={() => setRegisterType(UserType.Customer)}>Customer</Button>
<Button onClick={() => setRegisterType(UserType.Partner)}>Partner</Button>
```

Рендеровање форми за унос података се врши креирањем нове компоненте `FormComponent` која ће на основу одабраног типа корисника уметнути одговарајућу форму на страницу за регистрацију.

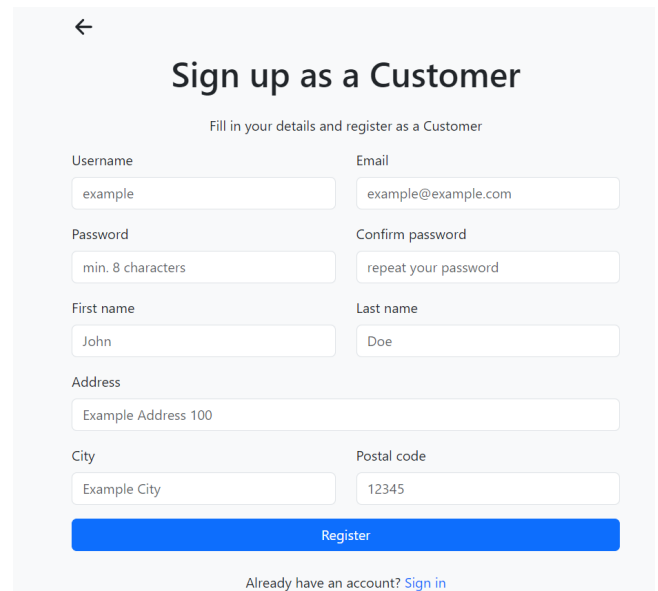
```
const FormComponent = ({ onSubmit }: FormComponentProps) => {
  switch (registerType) {
    case UserType.Customer:
      return <RegisterCustomerForm onSubmit={onSubmit} />
    case UserType.Partner:
      return <RegisterPartnerForm onSubmit={onSubmit} />
  }
}
```

У случају да је корисник одабрао да се региструје као партнер, односно креира налог за управљање ресторанима, на страници ће се појавити форма за унос података релевантних за партнере, а то су корисничко име, емаил адреса, лозинка, као и име и презиме (слика 2.1).

Исто тако, уколико је одабрани тип корисника – купац, приказана форма ће садржати поља за унос корисничког имена, емаил адресе, лозинке, имена и презимена, али ће уз то корисник такође уносити и адресу, град и поштански број како би поруџбине могле да буду достављене на ту локацију (слика 2.2).



Слика 2.1

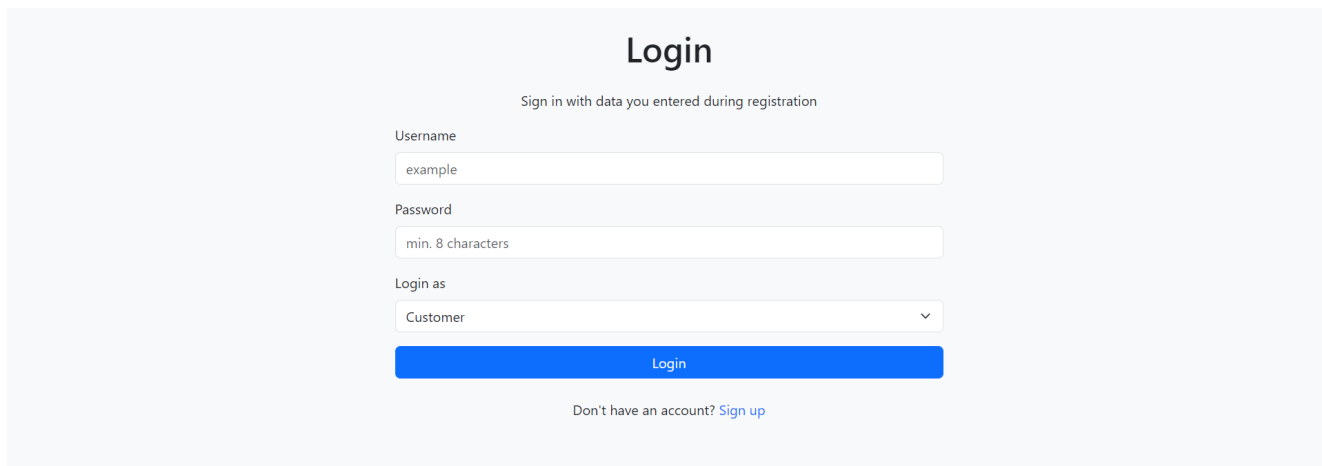


Слика 2.2

Стрелице које се налазе изнад форми за унос података омогућавају кориснику да се врати на страницу за одабир типа налога, тако што ће бити ресетована тренутна вредност стања – `useState` hook, која представља ознаку типа корисника.

```
const [registerType, setRegisterType] = useState<UserType | null>(...);
...
<Button onClick={() => setRegisterType(null)}><IoArrowBack /></Button>
```

Пријава на налог се извршава уношењем корисничког имена и лозинке, али и одабиром типа корисника из падајућег менија (слика 3). Овај корак је потребан како би серверска апликација знала којој бази података да приступи, с обзиром на то да се за сваки од три типа корисника користи засебна табела.

The image shows a login form titled "Login". Below the title is a subtitle: "Sign in with data you entered during registration". The form contains three input fields: "Username" with the placeholder text "example", "Password" with the placeholder text "min. 8 characters", and a "Login as" dropdown menu currently set to "Customer". Below these fields is a blue "Login" button. At the bottom of the form, there is a link that says "Don't have an account? Sign up".

Username

example

Password

min. 8 characters

Login as

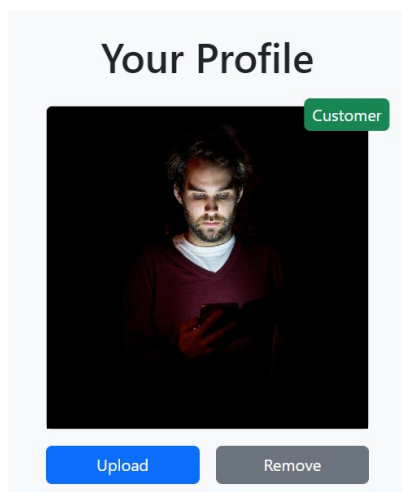
Customer

Login

Don't have an account? [Sign up](#)

Слика 3

Регистровани корисници могу да прегледају и измењују своје податке на страници профила. Поред промене података који су унети приликом регистрације (слика 4.2), укључујући и промену лозинке (слика 4.3), корисници такође могу да додају или бришу своју профилну слику (слика 4.1). На овој страници корисник може да види и коју улогу у систему поседује.

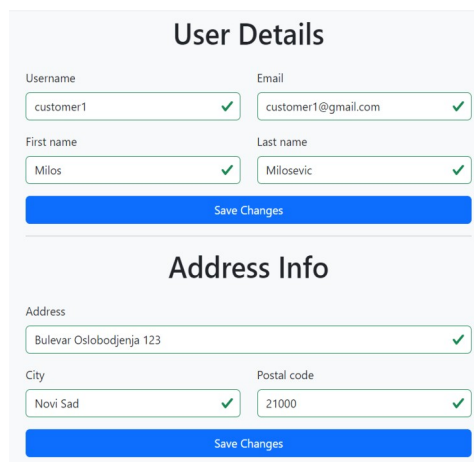
The image shows a user profile page titled "Your Profile". It features a profile picture of a man with a beard looking at a phone. To the right of the picture is a green badge that says "Customer". Below the picture are two buttons: "Upload" (blue) and "Remove" (grey).

Your Profile

Customer

Upload Remove

Слика 4.1

The image shows two forms. The top form is titled "User Details" and contains fields for Username (customer1), Email (customer1@gmail.com), First name (Milos), and Last name (Milosevic). Each field has a green checkmark icon to its right. Below these fields is a blue "Save Changes" button. The bottom form is titled "Address Info" and contains fields for Address (Bulevar Oslobođenja 123), City (Novi Sad), and Postal code (21000). Each field has a green checkmark icon to its right. Below these fields is a blue "Save Changes" button.

User Details

Username customer1 ✓

Email customer1@gmail.com ✓

First name Milos ✓

Last name Milosevic ✓

Save Changes

Address Info

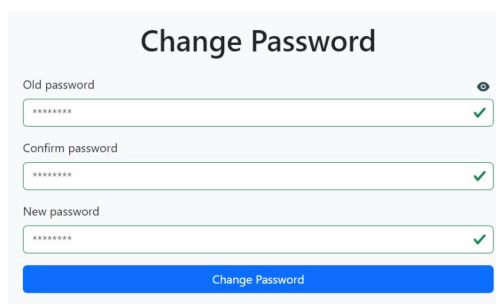
Address Bulevar Oslobođenja 123 ✓

City Novi Sad ✓

Postal code 21000 ✓

Save Changes

Слика 4.2

The image shows a "Change Password" form. It contains three input fields: "Old password", "Confirm password", and "New password". Each field has a green checkmark icon to its right. Below these fields is a blue "Change Password" button.

Change Password

Old password ✓

Confirm password ✓





New password ✓

Change Password

Слика 4.3






За разлику од купаца, који ће након пријављивања на налог бити пребачени на профилну страницу, партнер и админ ће бити пребачени на страницу која представља контролну таблу. Контролна табла омогућава овим корисницима да прегледају ресурсе који су релевантни за њихову улогу.

На контролној табли админа биће приказани сви регистровани партнери ради верификације (слика 5.1), затим сви ресторани у систему, као и све постојеће поруџбине постоје. Процес верификације је такав да када админ притисне једно од два доступна дугмета (прихвати и одбиј), апликација шаље захтев серверу са идентификационим бројем партнера и статусом верификације, који је енумерација и његове вредности су познате и клијентској и серверској апликацији.

Partners						
ID	Image	Username	First name	Last name	Status	Verify
17		partner2	Jovan	Jovanovic		✓ ✗
16		partner1	Marko	Markovic		✓ ✗

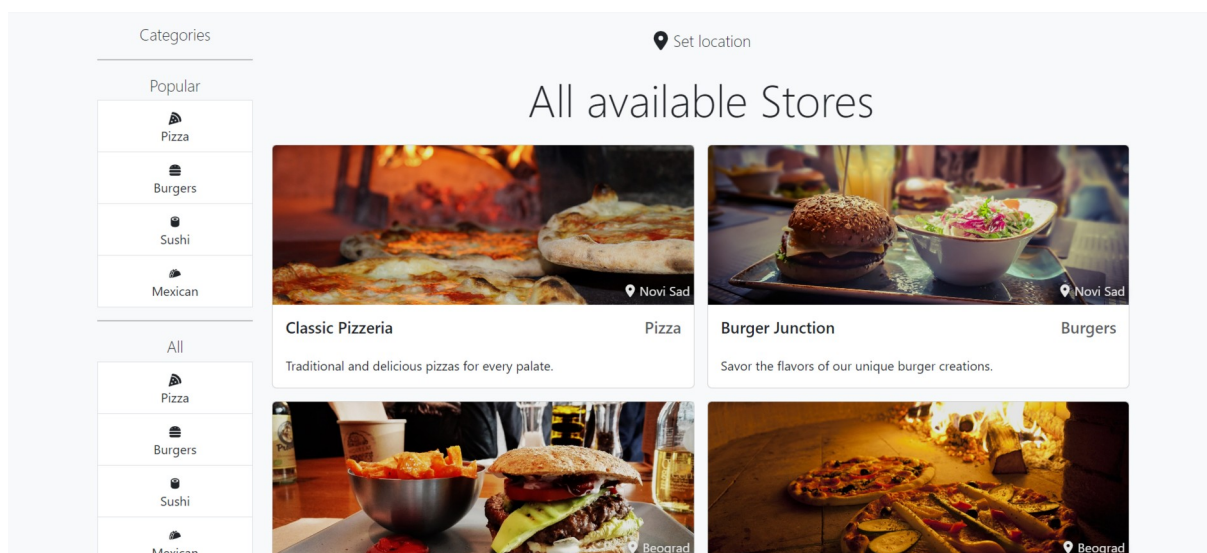
Слика 5.1

Што се тиче партнера, они на контролној табли могу да прегледају постојеће ресторане, додају нове и да прегледају постојеће поруџбине у тим ресторанима, али само под условом да су верификовани, тј. прихваћени од стране админа. Исто тако, партнер ће на основу обавештења на врху странице моћи да сазна свој статус верификације (слика 5.2).

Your current status is: Accepted. You can now perform all store and product related actions.							
Your Stores							
ID	Image	Name	Category	Address	City	Partner	Visit
39		Classic Pizzeria	Pizza	123 Pizza Street	Novi Sad	16	Visit
41		Burger Junction	Burgers	11 Burger Boulevard	Novi Sad	16	Visit
45		Taco Time	Mexican	123 Taco Street	Novi Sad	16	Visit
47		Groceries Galore	Groceries	789 Fresh Street	Novi Sad	16	Visit

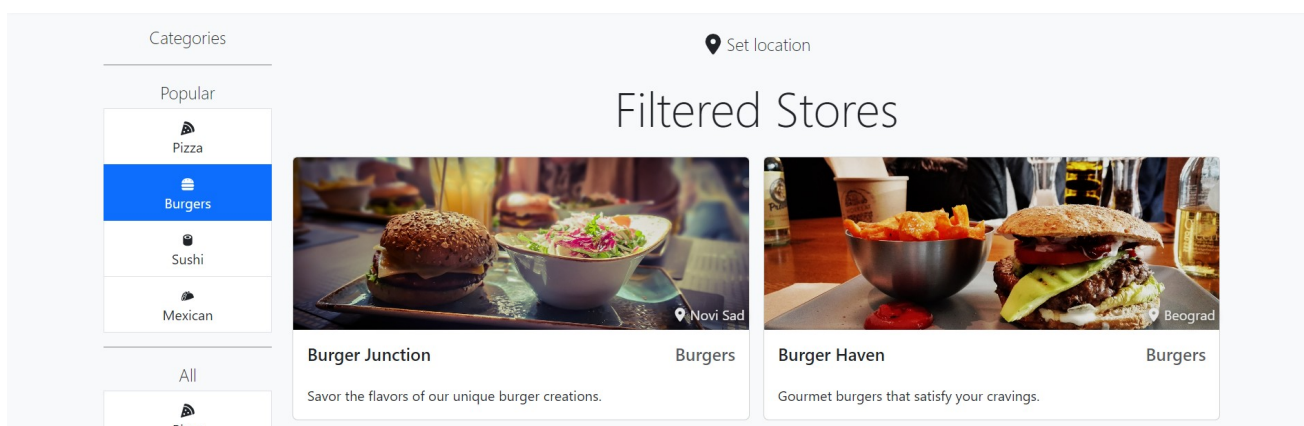
Слика 5.2

На страници за приказ свих доступних ресторана корисник може да види основне информације свих доступних ресторана у систему (слика 6.1), затим да пређе на страницу за приказ појединачног ресторана, али и да претражује ресторане по категорији. Уколико је корисник одабрао адресу за доставу, биће му приказани само ресторани који достављају на ту локацију (слика 6.2).



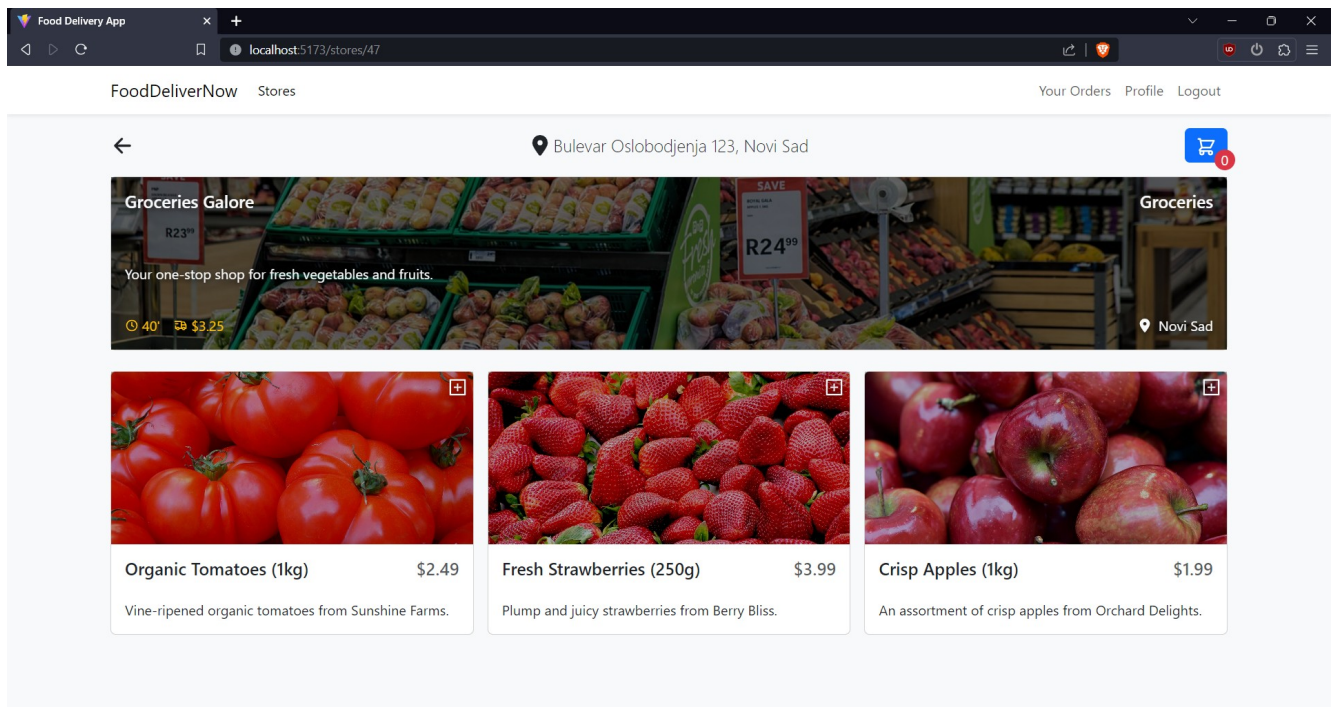
Слика 6.1

На бочној траци понуђен је списак свих доступних категорија, као и списак најпопуларнијих категорија, односно 3-4 категорије које су виђене у највише ресторана. Примери категорија доступних у систему, током писања пројектне документације, су Пиза, Бургери, Суши, мексичка храна и намирнице.



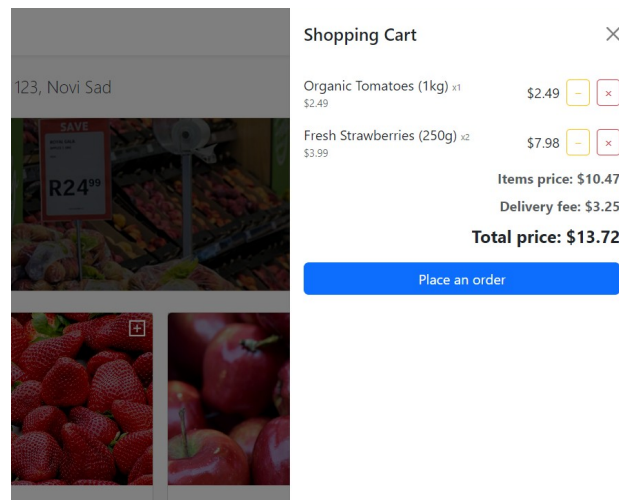
Слика 6.2

Кликом на једну од доступних картица корисник прелази на страницу одабраног ресторана (рутирање ка /stores/id) где су поред основних информације доступне и информације везане за доставу, као што су трошкови доставе и процењено време доставе (слика 7). Учитавање података о ресторану је имплементирано тако да се из УРЛ-а извлачи идентификациони број ресторана, врши провера да ли је валидан, а затим се из глобалног стања апликације преузима одабрани ресторан, уколико постоји. Непостојање ресторана са наведеним бројем или невалидност броја ће вратити корисника назад на страницу са свим доступним ресторанима.



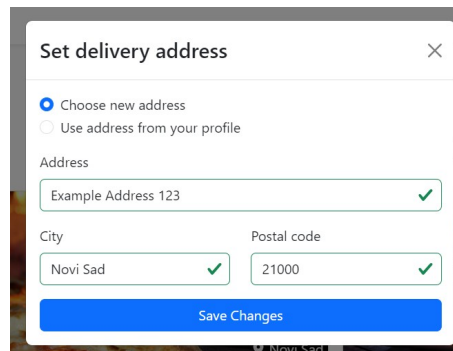
Слика 7

Такође, на овој страници су приказани и свих производи које нуди одабрани ресторан, а уколико је улоговани корисник – купац, кликом на одређено дугме може да додаје те артикле у своју корпу. Корпа (слика 8) садржи основне информације о артиклима, као што су име, цена и количина, а у укупну цену укључује и трошкове доставе. Уколико је додат неки производ у корпу, купац може да умањи изабрану количину, али и да у потпуности избаци производ из корпе. Уколико су све провере успешно извршене, поруџбина ће бити креирана, а корпа враћена на иницијално стање.



Слика 8

Како би купац уопште могао да изврши поруџбину, потребно је да одабере адресу за доставу (слика 9). Та опција се налази на страницама за приказ свих и појединачних ресторана, а иницијално ће бити искоришћена адреса са профила купца. Кликом на дугме за одабир локације, купац може да унесе и нову адресу, након чега ће му бити приказани ресторани који су доступни за ту локацију. Оваквом имплементацијом, купац није везан са само једну адресу, већ може да поручује са било које локације.



Слика 9

Након креирања поруџбине купац може да прегледа претходне и активне поруџбине на посебној страници, коју може да посети кликом на одговарајуће дугме у навигацији на самом врху. Ова страница садржи табелу са свим поруџбинама и приказује основне информације, као што су назив ресторана, датум и време, статус и укупну цену поруџбине (слика 10).

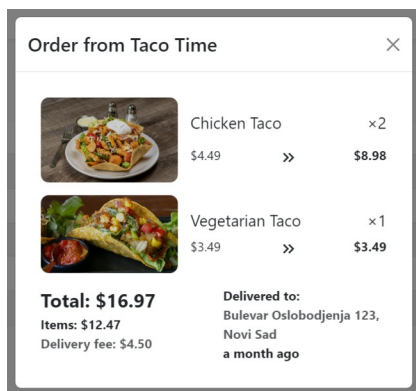
FoodDeliverNow Stores Your Orders Profile Logout

Your Order History					
Order ID	Store	Creation date	Creation time	Status	Total amount
75	Classic Pizzeria	August 11, 2023	7:30 PM	✓	\$45.72
76	Taco Time	August 11, 2023	7:30 PM	✓	\$16.97
77	Burger Junction	August 11, 2023	7:30 PM	✗	\$24.22
88	Classic Pizzeria	August 23, 2023	7:10 PM	✓	\$19.74
89	Groceries Galore	August 31, 2023	11:39 AM	✓	\$11.72
90	Classic Pizzeria	September 8, 2023	5:16 PM	✗	\$30.73
91	Groceries Galore	September 8, 2023	5:17 PM	...	\$13.22

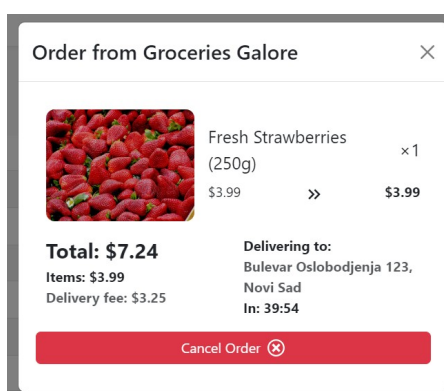
Слика 10

Кликом на једну од постојећих поруџбина купац може да прегледа и остале детаље, као што су појединачни производи, количину поручених производа, трошкове доставе, адресу доставе, а у зависности од стања поруџбине може да види и време када је достава извршена, односно тајмер који представља процењено време доставе.

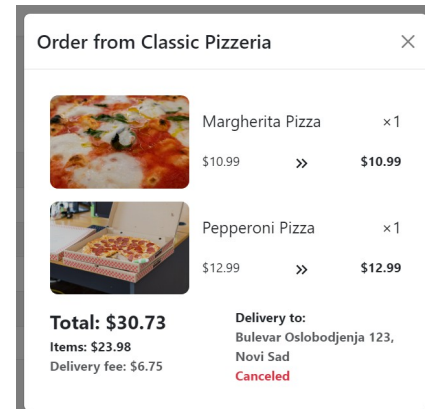
У зависности од стања, поруџбина може бити извршена (слика 10.1), активна (слика 10.2) или отказана (слика 10.3). Купац има опцију и да кликом на дугме откаже поруџбину уколико већ није извршена и у том случају ресторанима ће бити враћени одабрани производи (количине ће бити враћене на претходно стање).



Слика 10.1



Слика 10.2



Слика 10.3

Одбројавање до доставе је имплементирано тако што се на страници за поруџбине чува податак о тренутном времену и информација се ажурира сваке секунде коришћењем `useEffect` hooka, док се коришћењем „`moment.js`“ библиотеке рачуна разлика између тренутног времена и времена када би достава требало да буде достављена (у будућности). Та разлика, у милсекундама, се форматира у облик „`mm:ss`“ и приказује у компоненти.

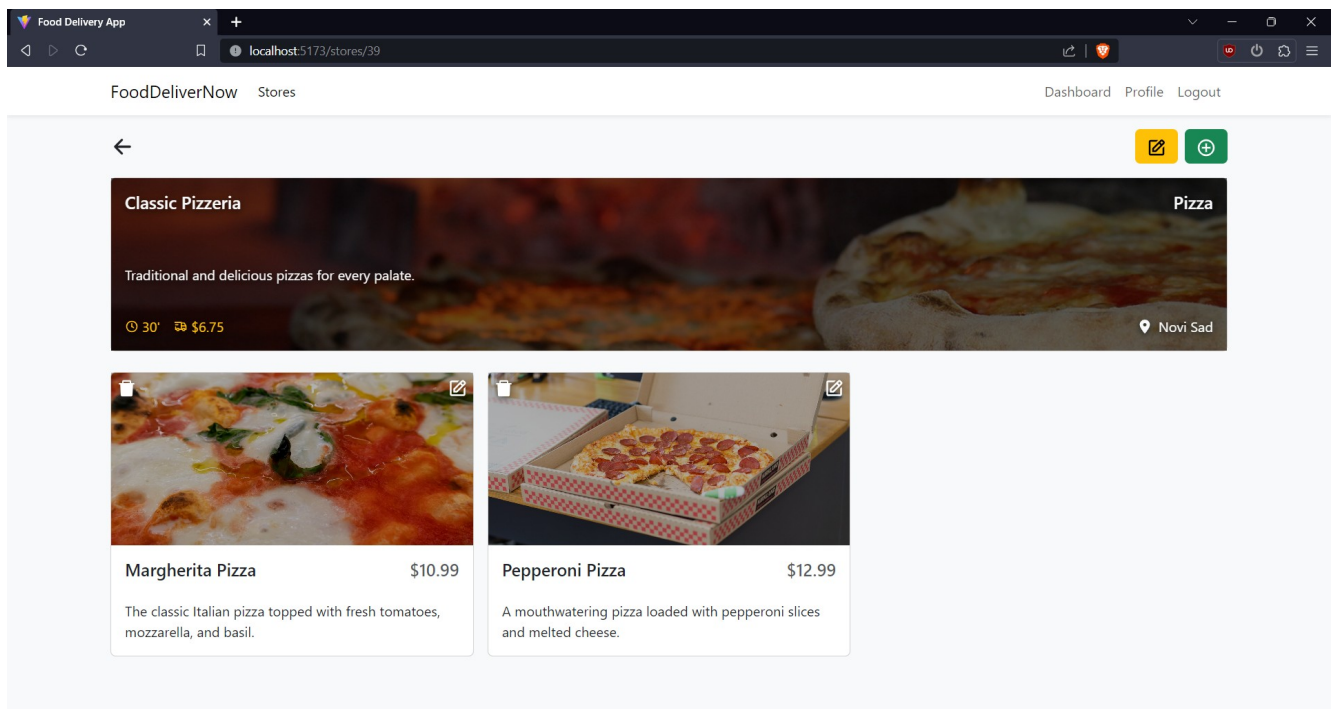
```
function getFormattedDeliveryTime(order: Order) {
  const { deliveryTime } = order.store;
  const deliveryAt = moment(order.createdAt).add(deliveryTime, "minutes");

  const difference = moment.duration(deliveryAt.diff(currentTime), "milliseconds");

  return moment.utc(difference.asMilliseconds()).format("mm:ss");
}
```

Што се тиче осталих типова корисника, на страници ресторана админи немају никакву контролу, док партнери (само уколико су власници изабраног ресторана) могу да измене основне информације ресторана, додају и измене производе, али и да обришу производе из система, након чега они више неће бити доступни за поручивање (слика 11).

С обзиром на то да је имплементирано само логичко брисање, старе поруџбине неће изгубити било какве информације везане за производ. Брисањем производа, вредност атрибута `IsCanceled` у моделу производа биће постављена на „`true`“.



Слика 11

Форме за додавање и модификовање ресторана, односно за додавање и модификовање производа су имплементирани на исти начин, с тим што се код модификовања за почетне вредности узимају постојећи подаци.

```
const initialValues = {
  name: data?.name || "",
  description: data?.description || "",
  category: data?.category || "",
  ...
}
```

За разлику од форми везаних за производе, форме за додавање и модификовање ресторана су због обима података и нешто дуже, те су поља за унос раздвојена у више корака – основне информације, контакт информације и информације везане за доставу. Структуре форми су приказане на сликама 12.1 и 12.2.

Слика 12.1

Слика 12.2

ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА

Обележавање подручја у којем се врши достава

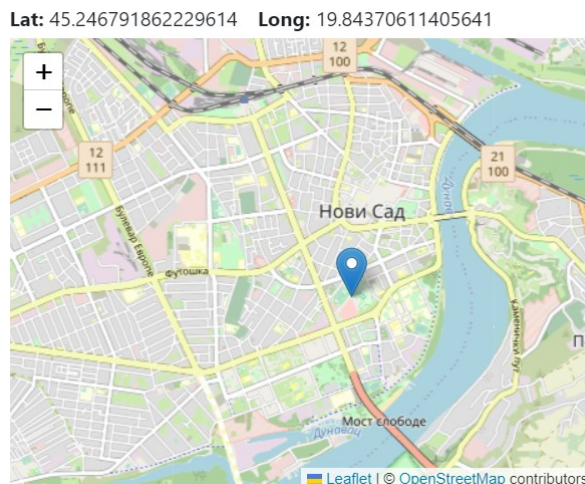
Апликација захтева од купца (клијента) ручно уношење адресе, како при регистрацији тако и при одабиру локације за доставу, док од партнера захтева да унесе адресу ресторана, односно продавнице. Због начина на који је апликација имплементирана, клијент може да поручује само из ресторана или продавнице која се налази у граду у којем клијент захтева доставу, односно ни на један начин није дефинисана територија за коју ресторан или продавница врши доставу.

Овај проблем би могао да се отклони имплементацијом мапе на којој би партнер (власник ресторана или продавнице) цртао полигон чије тачке које представљају координате и тиме дефинисао област покривености (слика 13.1). Са друге стране, након што клијент унесе адресу за доставу био би коришћен један од доступних Geolocation API-ја ради добијања координата те локације.

Друга опција је да на истој мапи клијент постави тачку на тренутну локацију (слика 13.2), што би одмах дало координате, али би слање захтева ка Geolocation API-ју било потребно и у овом случају – ради добијања адресе и броја.



Слика 13.1



Слика 13.2

Оваквом имплементацијом би клијенту били представљени сви ресторани који достављају на одабрану локацију, али то не би био случај и са клијентом који се налази изван подешене територије, без обзира да ли се налазе у истом граду или не.

Stripe интеграција

Stripe је ирско-америчка компанија која пружа услуге за online плаћања. Платформа намењена програмерима даје могућност прихватања кредитних картица и других начина плаћања. Постоји библиотека намењена React програмерима за лакшу интеграцију ове платформе у разне веб апликације.

Ова технологија би могла да буде искоришћена приликом поручивања производа у неком од доступних ресторана у апликацији, а „test mode“ намењен програмерима би пружио доживљај да је трансакција успешно обрађена.

Сервис за обавештавање корисника путем електронских порука

Због начина на који је систем имплементиран, корисници – купци могу да прате стање својих поруџбина само приступом апликацији. Овај „проблем“ би могао да буде решен имплементацијом сервиса који би обавештавао кориснике о креираним и отказаним поруџбинама путем електронских порука. Исти случај би био и са партнерима, који би на овај начин могли да буду обавештени о промени њиховог статуса верификације, о новим поруџбинама у ресторанима, итд. Ипак, код клијената би ова функционалност имала и значај у смислу сигурности, јер би врло брзо били информисани о акцијама које су извршене од стране трећег лица.

ЛИТЕРАТУРА

- [1] *Uspon mobilnih aplikacija za dostavu hrane i prevoz putnika (Slučaj Srbije) - Centar za izraživanje javnih politika, decembar 2020.*
- [2] *Why Vite?* - <https://vitejs.dev/guide/why.html>