



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



Дејан Курдулија PR 45/2019

**Веб продавница**

**ПРОЈЕКАТ**

- Примењено софтверско инжењерство (ОАС) -

Нови Сад, 06.07.2023

## 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

Пројекат веб продавница има за циљ омогућити корисницима куповину артикала. Продавница се састоји од корисника, артикала и поруџбина. Имамо три типа корисника: администратор, продавац, купац.

Нерегистровани корисници немају могућност коришћења апликације. Корисник типа администратор се уноси директно у базу и не постоји накнадна регистрација за овај тип корисника. Корисник типа продавац се региструје преко стандардног процеса регистрације. Корисник типа купац се може регистровати на два начина: стандардном регистрацијом путем апликације и регистрацијом путем друштвене мреже.

Регистрација се обавља уношењем личних података, постоји могућност додавања слике профила. Поред личних података бира се и тип корисника за регистрацију. Додатна заштита против нежељених корисника јесте да се понови унос лозинке. Сваки регистровани корисник се може пријавити на апликацију уношењем својих креденцијала мејла и лозинке.

Администратори имају могућности:

- Прихватање или одбијање захтјева за верификацију од стране продавца.
- Увид у списак свих поруџбина.
- Увид у списак свих купаца и продавца.

Продавци имају могућности:

- Додавање нових артикала.
- Увид у списак свих својих артикала.
- Измена података постојећег артикла.
- Увид у поруџбине које садрже неки њихов артикал.
- Потврда поруџбине која садржи неки њихов артикал.
- Преглед поруџбина које нису достаљене на мапи.

Купци имају могућности:

- Увид у списак свих артикала.
- Увид у списак свих својих поруџбина осим оних које су отказане.
- Прављење нове поруџбине са доступним артиклима.
- Плаћање поруџбине поузећем, картицом или путем *PayPal-a*[6].
- Праћење одбројавања времена до доставе.
- Отказивање поруџбине.

Сви регистровани корисници имају могућност измене података профила. Тип корисника се не може накнадно мењати. Промена лозинке захтева унос тачне старе лозинке. Сви корисници имају корисничку таблу или ти *dashboard* где имају доступне опције за одређени тип корисника. Корисника типа продавац мора имати одобрену верификацију да би могао да користи апликацију. Продавац резултате процеса верификације добија на мејл.

## ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА

**Visual Studio 2022 Community[1]** - бесплатно интегрисано развојно окружење (IDE) које пружа моћне алатке за програмирање и развој софтвера. Омогућава програмерима да креирају различите врсте апликација, укључујући веб, десктоп и мобилне апликације. Има богат набор функционалности као што су кодни преглед, дебагирање, управљање верзијама, интеграција са различитим програмским језицима и платформама, и много више. У овој апликацији овај алат је кориштен за развој бекенд решења.

**Visual Studio Code[2]** - бесплатни интерфејс за развој који је лак, брз и израђен за програмирање различитих врста апликација. Он нуди богат скуп функција, укључујући кодни преглед, интелигентно навођење, дебагирање, интеграцију са системима контроле верзија и широку подршку за екстензије. Са његовом крос-платформском природом, програмери могу да га користе на *Windows*, *macOS* и *Linux* оперативним системима. *Visual Studio Code* је популаран међу програмерима због своје приступачности, проширивости и брзине, што га чини идеалним избором за развој софтвера. У овој апликацији овај алат је кориштен за развој фронтенд решења.

**SQL Management Studio 19[3]** - алатка која омогућава управљање и администрацију *SQL Server* база података. Ова алатка пружа напредне функционалности за креирање, уређивање и извршавање *SQL* упита, дебагирање и профјалинг базе података, управљање безбедношћу и извршавање административних задатака. Има интуитиван кориснички интерфејс који омогућава лак приступ и манипулацију подацима у бази, чинећи га неопходним алатом за *SQL* програмере и администраторе база података. У овој апликацији ово окружење је кориштено за развој решења базе података.

**React[4]** - *JavaScript* библиотека за развој корисничког интерфејса у веб апликацијама. Он користи компонентну базирану архитектуру, што омогућава лако поновно коришћење и модуларност кода. Са својим виртуелним *DOM (Document Object Model)* приступом, *React* обезбеђује ефикасно ажурирање само промењених делова корисничког интерфејса, што доприноси брзини и перформансама апликације.

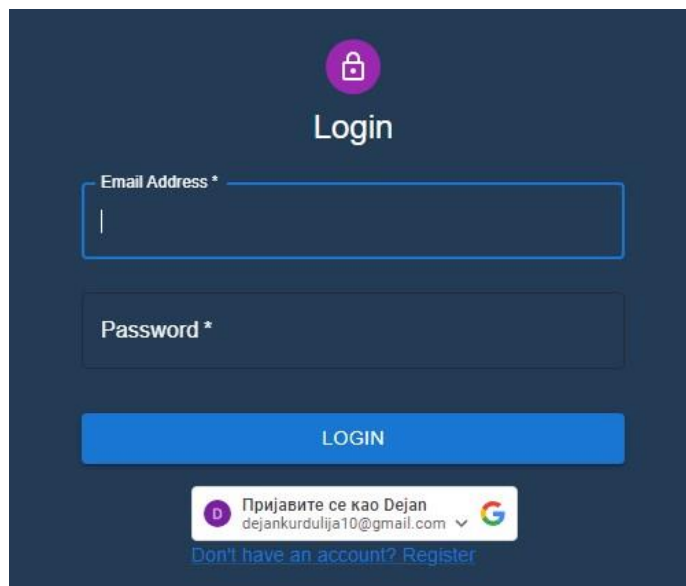
**ASP .NET Core 6.0 Web API C#[5]** - моћан *framework* за изградњу веб апликација у *C#* језику. Пружа ефикасан и скалабилан начин за развој *API* -ја за модерне апликације. Са богатим сетом алата и подршком за аутентификацију, ауторизацију и управљање токовима података, олакшава имплементацију сигурности. У апликацији коришћени су следећи пакети:

- *Automapper* - обезбеђује мапирање својстава између објеката класа.
- *BCrypt.Net* – енкрипција и поређење хешираних ресурса.
- *Google.Apis.Auth* - обезбеђује аутентификацију и регистрацију преко друштвене мреже *Google*.
- *Microsoft.AspNetCore.EntityFrameworkCore* - обезбеђује ресурсе из базе података.
- *Microsoft.AspNetCore.Authentication.JwtBearer* - производња и валидација токена за препознавање корисника.

## ОПИС РЕШЕЊА ПРОБЛЕМА

### Непријављен корисник

Приликом отварања апликација почетна страница је уједно и почетна страница непријављеног корисника. На почетној страни се налази форма за пријаву гдје корисник уноси своје креденцијале.



Слика 3.1.1. Форма за пријаву корисника

Корисник може да се пријави путем *Google* налога. Непријављеним корисницима је такође понуђен линк уколико немају регистрован налог, линк води ка форми за регистрацију.

```
public async Task<string> Login(LoginDto loginDto)
{
    var users = await _repository.GetAll();
    User? user = users.Where(u => u.Email == loginDto.Email).FirstOrDefault();
    if (user == null)
        throw new Exception($"User with {loginDto.Email} doesn't exist! Try again.");
    if (!BCrypt.Net.BCrypt.Verify(loginDto.Password, user.Password))
        throw new Exception($"Password is incorrect! Try again.");

    var claims = new[] {
        new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
        new Claim(JwtRegisteredClaimNames.Iat, DateTime.UtcNow.ToString()),
        new Claim("UserId", user.Id.ToString()),
        new Claim("Email", user.Email!),
        new Claim(ClaimTypes.Role, user.Type.ToString()),
        new Claim("Verification", user.Verification.ToString())
    };

    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_configuration["Jwt:Key"] ?? "default"));
    var signIn = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);
    var token = new JwtSecurityToken(
        _configuration["Jwt:Issuer"],
        _configuration["Jwt:Audience"],
        claims,
        expires: DateTime.UtcNow.AddDays(1),
        signingCredentials: signIn);

    return new JwtSecurityTokenHandler().WriteToken(token);
}
```

Слика 3.1.2. Имплементација кода пријаве на апликацију.

Валидација података се ради и на серверској и на клијентској страни. Уколико је корисник унио погрешне креденцијале добиће поруку о грешци, уколико је корисник унио валидне податке, ти подаци се провјеравају тако што се тражи подударење мејла, а лозинка се криптује са *Bcrypt.Net* и пореди са криптованом вредношћу сачуваном у бази података. Уколико је пријава успешна функција враћа токен, који је генерисан са *JwtBearer*[7] и који садржи *claim*-ове који су

потребни за ауторизацију и траје 24 сата. Уколико се корисник региструје преко *Google-a* аутоматски се пријављује на систем. Имплементација регистравања путем *Google* налога је приказана на сликама испод.

```
public async Task<string> GoogleLogin(string token)
{
    GoogleUserDto externalUser = await VerifyGoogleToken(token);
    if (externalUser == null) { throw new ConflictException("Invalid user google token."); }

    List<User> users = await _repository.GetAll();
    User user = users.Find(u => u.Email.Equals(externalUser.Email));

    if (user == null)
    {
        user = new User()
        {
            FirstName = externalUser.FirstName,
            LastName = externalUser.LastName,
            Username = externalUser.Username,
            Email = externalUser.Email,
            Image = new byte[0],
            Password = "",
            Address = "",
            BirthDate = DateTime.Now,
            Type = EUserType.CUSTOMER,
            Verification = EVerificationStatus.ACCEPTED
        };

        await _repository.Register(user);
    }

    var claims = new[] {
        new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString()),
        new Claim(JwtRegisteredClaimNames.Iat, DateTime.UtcNow.ToString()),
        new Claim("UserId", user.Id.ToString()),
        new Claim("Email", user.Email!),
        new Claim(ClaimTypes.Role, user.Type.ToString()),
        new Claim("Verification", user.Verification.ToString())
    };

    var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_configuration["Jwt:Key"] ?? "default"));
    var signIn = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);
    var tokenString = new JwtSecurityToken(
        _configuration["Jwt:Issuer"],
        _configuration["Jwt:Audience"],
        claims,
        expires: DateTime.UtcNow.AddDays(1),
        signingCredentials: signIn);
    return new JwtSecurityTokenHandler().WriteToken(tokenString);
}
```

Слика 3.1.3. Регистравање корисника путем *Google-a*

```
private async Task<GoogleUserDto> VerifyGoogleToken(string externalLoginToken)
{
    try
    {
        var validationSettings = new GoogleJsonWebSignature.ValidationSettings()
        {
            Audience = new List<string>() { _googleClientId.Value }
        };

        var googleUserInfo = await GoogleJsonWebSignature.ValidateAsync(externalLoginToken, validationSettings);

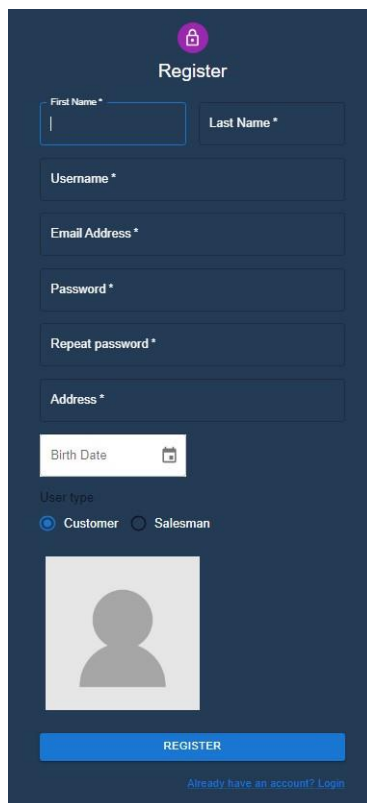
        GoogleUserDto externalUser = new GoogleUserDto()
        {
            Username = googleUserInfo.Email.Split("@")[0],
            FirstName = googleUserInfo.GivenName,
            LastName = googleUserInfo.FamilyName,
            Email = googleUserInfo.Email
        };

        return externalUser;
    }
    catch
    {
        return null;
    }
}
```

Слика 3.1.4. Регистравање корисника путем *Google-a*

Уколико корисник кликне на линк отвара му се страница за регистравање гдје може унијети своје личне податке, додати профилну слику и изабрати који тип корисника жели да буде.

Корисничко име и мејл морају да буду јединствени, поред тога да датум буде валидно унесен корисник такође мора да има преко 18 година да би могао да се региструје на апликацију. Да би се додатно смањили нежељени корисници постоји поље за понављање лозинке, за успјешну регистрацију је потребно да се лозинке поклапају.



Слика 3.1.5. Страница за регистрацију корисника.

Приликом уноса сви подаци морају бити попуњени. Лозинка се не чува у бази у свом оригиналном формату већ се криптује са *Bcrypt.Net*[8] алгоритмом и чува се њена хеширана вредност. Имплементација регистровања корисника на апликацију је приказана на слици дела кода испод.

```
public async Task<UserDto> Register(RegisterDto registerDto)
{
    List<User> users = await _repository.GetAll();

    if (String.IsNullOrEmpty(registerDto.FirstName) || String.IsNullOrEmpty(registerDto.LastName)
        || String.IsNullOrEmpty(registerDto.Username) ||
        String.IsNullOrEmpty(registerDto.Email) || String.IsNullOrEmpty(registerDto.Address) ||
        String.IsNullOrEmpty(registerDto.Password) || String.IsNullOrEmpty(registerDto.RepeatPassword)
        || String.IsNullOrEmpty(registerDto.Type.ToString()))
    {
        throw new BadRequestException($"You must fill in all fields for registration!");
    }

    if (users.Any(u => u.Username == registerDto.Username))
        throw new ConflictException("Username already in use. Try again!");

    if (users.Any(u => u.Email == registerDto.Email))
        throw new ConflictException("Email already in use. Try again!");

    if (registerDto.Password != registerDto.RepeatPassword)
        throw new BadRequestException("Passwords do not match. Try again!");

    User newUser = _mapper.Map<RegisterDto, User>(registerDto);
    if (registerDto.ImageForm != null)
    {
        using (var memoryStream = new MemoryStream())
        {
            registerDto.ImageForm.CopyTo(memoryStream);
            var imageBytes = memoryStream.ToArray();
            newUser.Image = imageBytes;
        }
    }
    newUser.Password = BCrypt.Net.BCrypt.HashPassword(newUser.Password);
    newUser.Type = (EUserType)Enum.Parse(typeof(EUserType), registerDto.Type.ToUpper());

    if (newUser.Type == Common.EUserType.SALESMAN)
        newUser.Verification = Common.EVerificationStatus.INPROGRESS;
    else
        newUser.Verification = Common.EVerificationStatus.ACCEPTED;

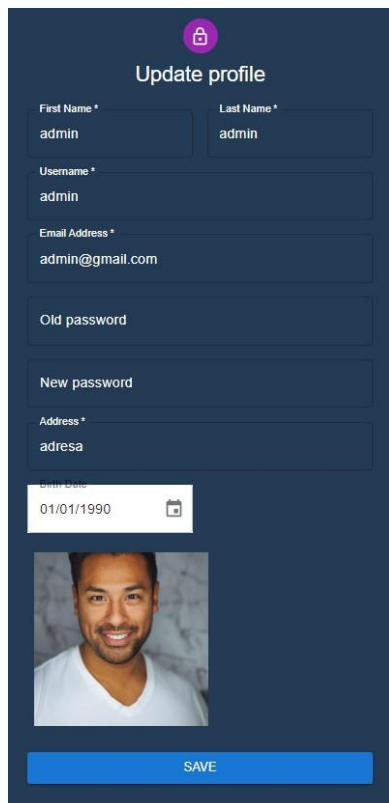
    UserDto dto = _mapper.Map<User, UserDto>(await _repository.Register(newUser));
    return dto;
}
```

Слика 3.1.6. Имплементација кода регистрације корисника.



## Пријављени корисник

Корисник може да измени своје личне податке као и слику профила. Уколико жели да измени лозинку потребно је да унесе исправну стару лозинку а затим и нову. Уколико је неко поље празно или невалидно унесено кориснику ће се показати порука о грешци.



Update profile

First Name \*  
admin

Last Name \*  
admin

Username \*  
admin


Email Address \*  
admin@gmail.com

Old password

New password

Address \*  
adresa

Birth Date  
01/01/1990



SAVE

Слика 3.2.1. Измена података профила корисника.

## Администратор

Када се корисник пријави на систем има приступ својим опцијама. Сваки корисник има навигациони бар са дугметом *Home* које води на почетну страницу пријављеног корисника и *Logout* које служи за одјаву са апликације. Почетна страница односно *dashboard* зависи од врсте пријављеног корисника. Сваки корисник има доступну опцију за измену профила а администратор поред тога има опцију да види списак свих продаваца и списак свих поруџбина.



Слика 3.3.1. Почетна страница администратора

Администратор може да види листу свих поруџбина са свим детаљима везаних за њих.

▼	6	komentar	Dr. Sime Miloševića br. 10	20	2023-06-29T07:38:13		DENIED
▼	1002	neka bude brzo	Dr. Sime Miloševića br. 10	20	2023-07-06T22:30:13	0:54:00	INPROGRESS
▼	1003	komentar	Novosadskog sajma 36	20	2023-07-06T22:30:35		INPROGRESS

Слика 3.3.2. Листа свих поруџбина.



Приликом регистрација новог продавца на апликацију његова регистрација треба да буде одобрена тј. верификована од стране администратора. Администратор може да прихвати или одбије захтев за верификацију од стране продавца.

HOME						
ID	First name	Last name	Username	Email	Verification	Verify
3	salesman	salesman	salesman	salesman@gmail.com	ACCEPTED	
4	Dejan	Kurdulija	Dejan	dejan@gmail.com	ACCEPTED	
1,...	test	test	test	test@gmail.com	INPROGRESS	ACCEPT DENY

Слика 3.2.4. Верификација продавца од стране администратора.

Након процеса верификације продавац преко мејла буде обавештен о резултатима.

Мејл се шаље уз помоћ пакета *MimeKit*. Имплементација слања мејла продавцу након процеса верификације је приказана на слици испод.

```
public async Task SendEmail(string email, string verification)
{
    string text = $"Your verification request for Online Shop is {verification}";
    var mail = new MimeMessage
    {
        Subject = "Verification",
        Body = new TextPart(MimeKit.Text.TextFormat.Plain) { Text = text }
    };

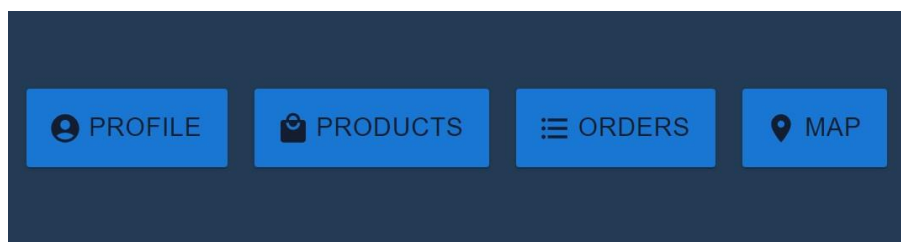
    mail.From.Add(new MailboxAddress(_configuration["MailSettings:DisplayName"],
        _configuration["MailSettings:From"]));
    mail.To.Add(MailboxAddress.Parse(email));

    SmtpClient smtp = new SmtpClient();
    await smtp.ConnectAsync(_configuration["MailSettings:Host"],
        int.Parse(_configuration["MailSettings:Port"]!),
        SecureSocketOptions.Auto);
    string s = _configuration["MailSettings:From"] + " " + _configuration["MailSettings:Password"];
    await smtp.AuthenticateAsync(_configuration["MailSettings:From"],
        _configuration["MailSettings:Password"]);
    await smtp.SendAsync(mail);
    await smtp.DisconnectAsync(true);
}
```

Слика 3.2.5. Слање мејла продавцу о резултатима процеса верификације.



## Продавац

Улога продавца на овој апликацији јесте креирање нових производа.



Слика 3.3.1. Почетна страница корисника типа продавац.

Продавац може да види листу свих својих производа и може да управља њима.

HOME						LOGOUT
ID	Image	Name	Description	Amount	Price	ADD NEW PRODUCT
1		banana	banana	10	10	CHANGE DELETE
2		tresnja	svjeza	20	10	CHANGE DELETE

Слика 3.3.2. Листа производа продавца.

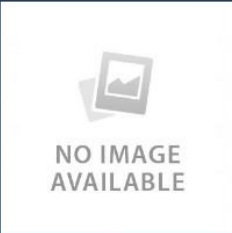
Кликом на дугме *Add new product* кориснику се отвара нови искачући прозорчић где може унијети све потребне податке и креирати нови производ. За сваки производ је могуће додати и слику. Сва поља осим слике су обавезна за унијети и уколико неко није попуњено корисник ће добити поруку о грешци.

Name \*

Description \*

Amount \*

Price \*



ADD CANCEL

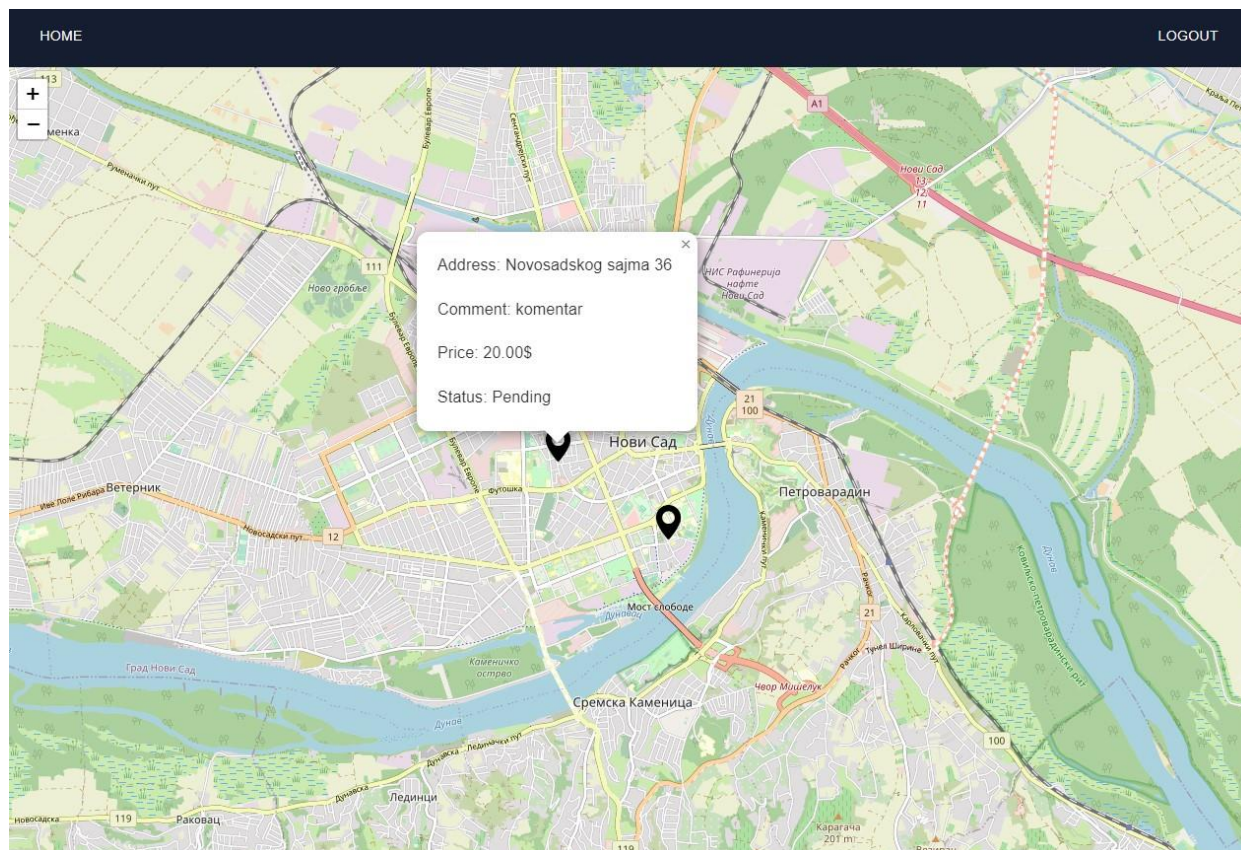
Слика 3.3.3. Креирање новог производа.

Поред креирања новог производа продавац такође може са њима да управља. Може да брише производе кликом на дугме *Delete* или да измени податке неког производа кликом на дугме *Change*. Продавац може да види листу свих поруџбина на којима се налази барем један његов производ. Кликом на стрелицу свака поруџбина се прошири и буду доступни сви њени детаљи. Да би достава почела поруџбина мора бити одобрена од стране продавца. Када продавац одобри поруџбину тада креће одбројавање до доставе које се аутоматски приказује и купцу који је и креирао ту поруџбину. Уколико је поруџбина отказана продавац неће моћи да је види.

HOME							LOGOUT
	Id	Comment	Address	Price	Order Time	Delivery Time	Status
▼	1002	neka bude brzo	Dr. Sime Miloševića br. 10	20	2023-07-06T22:30:13	0:57:20	INPROGRESS
▼	1003	komentar	Novosadskog sajma 36	20	2023-07-06T22:30:35		INPROGRESS
							APPROVE

Слика 3.3.4. Листа поруџбина продавца.

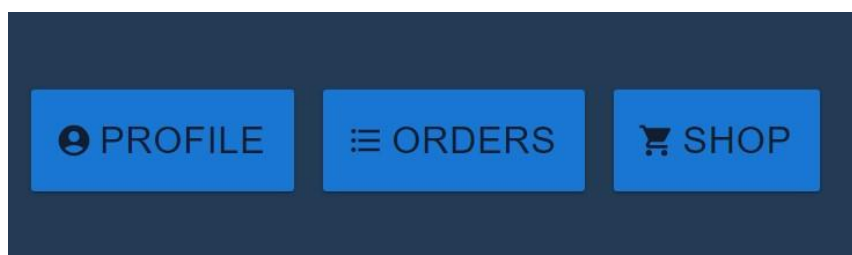
Сваки продавац може да отвори мапу и види своје поруџбине са тачном локацијом доставе. Уколико кликне на неку поруџбину на мапи приказће му се сви детаљи те поруџбине.



Слика 3.3.5. Мапа поруџбина.

## Купац

Корисници типа купац представљају крајњу тачку овог система. Главна карактеристика купаца јесте куповина артикала и прављење поруџбина.



Слика 3.4.1. Почетна страна корисника типа купац.

Купац може да види све своје поруџбине, у које спадају и поруџбине у току и претходно достављене поруџбине.

HOME

LOGOUT

Id	Comment	Address	Price	Order Time	Delivery Time	Status	
1002	neka bude brzo	Dr. Sime Miloševića br. 10	20	2023-07-06T22:30:13		INPROGRESS	DENY
1003	komentar	Novosadskog sajma 36	20	2023-07-06T22:30:35		INPROGRESS	DENY



Details

Name	Description	Price	Amount	Total price (\$)
tresnja	svjeza	10 \$	x2	20 \$

Слика 3.4.2. Поруџбине купца.

Кликом на стрелицу се свака поруџбина проширује и онда су доступни сви њени детаљи. Купац може да откаже поруџбину уколико је она у току или није још одобрена од стране продавца. Када продавац одобри поруџбину купцу почиње одбројавање времена до доставе.

Главна карактеристика купца јесте куповина производа и прављење самих поруџбина. Кликом на дугме *Shop* са почетне странице кориснику се отвара нови прозор гдје је приказана листа свих доступних производа у веб продавници. Поред сваког производа се налази дугме *Add to cart* чијим кликом се додавају производи у корпу. Уколико кликнемо дугме више пута, толико пута ће се одређени производ додати у корпу.

ID	Image	Name	Description	Amount	Price	
1		banana	banana	8	10	<a href="#">ADD TO CART</a>
2		tresnja	svjeza	18	10	<a href="#">ADD TO CART</a>

Слика 3.4.3. Продавница купца.

Кликом на дугме *My cart* кориснику се отвара нови искачући прозорчић који представља његову корпу. У корпи корисник може да види детаље сваког производа и укупну цену поруџбине. Корисник може да повећава или смањује количину сваког производа помоћу дугмади „+“ и „-“. У зависности од количине производа ажурира се и укупна цена поруџбине. Цена доставе поруџбине се рачуна у зависности од броја различитих продаваца чији се производи наручују, цена доставе за једног продавца јесте 20\$.

### My cart

Product	Price	Amount	
banana	10 RSD	x2	<a href="#">+</a> <a href="#">-</a>

Price: 20\$  
Delivery price: 20\$

[ORDER](#)
[CLOSE](#)

Слика 3.4.4. Корпа.



Кликом на дугме *Order* кориснику се отвара нови искачући прозорчић који представља наплату поруџбине. Да би извршио коначну потврду поруџбине корисник треба да унесе адресу доставе и уколико жели може да остави неки коментар.

Кориснику су понуђена три начина плаћања: поузећем, картицом или путем *PayPal-a*[6].

Уколико корисник изабере плаћање поузећем довољно је да кликне на *Order*, у супротном бира неке од понуђених опција. Уколико изабере плаћање путем *PayPal-a*[6] кориснику се отвара нови прозор који је заправо пријава на кориснички систем *PayPal-a*[6]. Корисник у сваком тренутку може да одустане од наплате или да промјени неке податке прије саме потврде поруџбине.

Слика 3.4.5. Наплата поруџбине.

Уколико су сва поља валидно попуњена креира се поруџбина. Статус поруџбине се ставља „У току“ а поље *Approved* на *false* зато што поруџбина треба бити одобрена од стране продавца. Уколико је количина наручених производа валидна у односу на доступну количину креира се поруџбина. Имплементација кода креирања поруџбине је приказана на слици испод.

```
public async Task<OrderDto> CreateOrder(int userId, CreateOrderDto orderDto)
{
    if (String.IsNullOrEmpty(orderDto.Address))
        throw new Exception($"You must fill field for address!");

    Order newOrder = _mapper.Map<CreateOrderDto, Order>(orderDto);
    newOrder.UserId = userId;
    newOrder.OrderTime = DateTime.Now;
    newOrder.Status = Common.EOrderStatus.INPROGRESS;
    newOrder.Approved = false;

    foreach (OrderProduct op in newOrder.OrderProducts)
    {
        Product p = await _productRepository.GetProductById(op.ProductId);

        if (op.Amount > p.Amount)
            throw new BadRequestException("There is not enough amount of this product.");

        op.ProductId = p.Id;
        p.Amount -= op.Amount;
        newOrder.Price += op.Amount * p.Price;
    }

    OrderDto dto = _mapper.Map<Order, OrderDto>(await _orderRepository.CreateOrder(newOrder));
    return dto;
}
```

Слика 3.4.6. Имплементација кода креирања поруџбине.

## ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА

- У циљу побољшања перформанси апликације, специфично фронтенд дела апликације може се користити други алат за креирање и оптимизацију кода (*Vite* уместо *create-react-app*).
- У циљу побољшања перформанси апликације, специфично фронтенд дела апликације може се користити *GraphQL* уместо традиционалног *Rest API*, јер гарантује већу ефикасност преноса података, флексибилност и смањење броја захтева.
- У циљу побољшања перформанси апликације, специфично бекенд дела апликације може се имплементирати микросервисна архитектура и тиме се додатно убрза пренос и обрада података као и побољша скалабилност система.
- По угледу на све модерне апликације и сајтове требало би омогућити и непријављеним корисницима увид у сам изглед апликације и одређене функционалности.
- Имплементирати систем поврата заборављене лозинке које би функционисало преко мејла корисника.
- У циљу побољшања задовољства купаца и продаваца требало би омогућити одређени временски период за који купац треба да одобри поруџбину, односно одређени временски за који купац може да поништи поруџбину.
- Функционалност мапе и праћења поруџбине је потребна и купцу колико и продавцу. Уколико су поруџбине међународне то би додатно допринело задовољству купаца.
- Имплементирати рецензије и задовољство купаца на одређене производе или продавце.
- Усавршавању апликације би додатно помогло и увођење више језика на којима апликација може да ради, као и могућност корисника да одабере њему одговарајући језик. У тренутном стању, сви линкови, дугмад и информације писани су на енглеском језику.

## ЛИТЕРАТУРА

- [1] <https://visualstudio.microsoft.com/vs/getting-started/>
- [2] <https://code.visualstudio.com/docs>
- [3] <https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16>
- [4] *Alex Banks. Schmidt, Learning React: Functional Web Development with React and Redux, 2017*
- [5] <https://learn.microsoft.com/en-us/aspnet/core/web-api>
- [6] <https://www.npmjs.com/package/@paypal/react-paypal-js/>
- [7] <https://www.c-sharpcorner.com/article/how-to-implement-jwt-authentication-in-web-api-using-net-6-0-asp-net-core/>
- [8] <https://learn.microsoft.com/en-us/answers/questions/830417/verify-passwords-with-bcrypt-net>