



УНИВЕРЗИТЕТ У НОВОМ САДУ
ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У
НОВОМ САДУ



Немања Малиновић PR108/2019

Веб продавница

ПРОЈЕКАТ

- Примењено софтверско инжењерство (ОАС) -

Нови Сад, 22.08.2023.

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

Овај пројекат има за циљ омогућавање корисницима куповину производа. Продавницу чине корисници, поруџбине и артикли. Постоје три типа корисника: купац, продавац, администратор.

Непријављени и нерегистровани корисници немају могућност употребе апликације. Корисник типа купац може извршити регистрацију на два начина: регистрацијом путем друштвене мреже и путем апликације.

Корисник типа продавац се региструје преко апликације. Корисник типа администратор је унет директно у базу и не постоји регистрација за овај тип корисника.

Процес регистрације се састоји од уношења личних података, уноса слике и одабира типа корисника.

Купци имају могућност:

- Креирање нове поруџбине .
- Отказивање поруџбине.
- Приказ списка свих доступних артикала.
- Приказ списка сопствених поруџбина осим оних које су отказане.
- Плаћање поруџбине поузећем, картицом или преко *PayPal-a*[6].
- Праћење времена доставе

Продавци имају могућност:

- Креирање новог артикла.
- Преглед списка сопствених артикала.
- Измена постојећег артикла.
- Преглед поруџбина које садрже њихов артикал.
- Потврда поруџбине која садржи њихов артикал.
- Преглед недостављених поруџбина на мапи.

Администратори имају могућност:

- Преглед свих поруџбина.
- Преглед списка свих купаца и продаваца.
- Прихватање или одбијање верификације продавца.

Сви регистровани корисници могу извршити измену профила. За промену лозинке потребан је унос тачне старе лозинке. Свим корисницима је доступан *dashboard* где имају опције на основу типа корисника. Тип корисника се не може мењати. Корисник типа продавац мора бити верификован од стране администратора како би могао користити апликацију. Продавац потврду о верификацији добија на мејл.

ОПИС КОРИШЋЕНИХ ТЕХНОЛОГИЈА И АЛАТА

Visual Studio Code[1] - бесплатни интерфејс за развој који је лак, брз и направљен 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.. У овој апликацији овај алат је коришћен за развој фронтенд решења.

Visual Studio 2022 Community[2] - бесплатно интегрисано развојно окружење (IDE) које пружа алате за програмирање и развој софтвера. Омогућава програмерима да креирају различите врсте апликација. Има велик избор функционалности као што су преглед кода, дебаговање, управљање верзијама, интеграција са различитим програмским језицима и платформама. У овој апликацији овај алат је коришћен за развој бекенд решења.

React[3] - *JavaScript* библиотека за развој корисничког интерфејса у веб апликацијама. Он користи архитектуру базирану на компонентама, што омогућава лако 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.

SQL Management Studio 19[4] - алатка која омогућава управљање и администрацију *SQL Server* база података. Ова алатка пружа напредне функционалности за креирање, уређивање и извршавање *SQL* упита, дебаговање и профайлинг базе података, управљање безбедношћу и завршавање административних задатака. Има кориснички интерфејс који омогућава лак приступ и управљање подацима у бази. У овој апликацији ово окружење је коришћено за развој решења базе података.

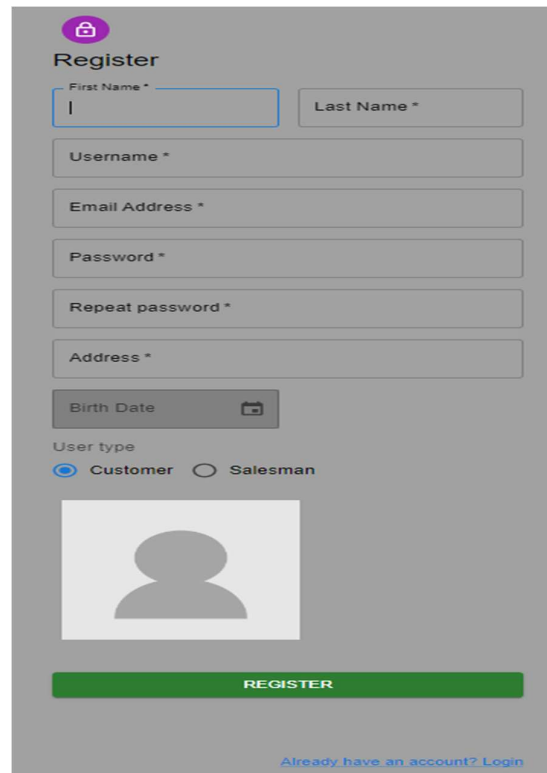
ASP .NET Core 6.0 Web API C#[5] - *framework* за изградњу веб апликација у *C#* језику. Пружа ефикасан начин за развој *API* -ја за модерне апликације. Са богатим сетом алата и подршком за аутентификацију, ауторизацију и управљање токовима података, олакшава имплементацију сигурности. У апликацији коришћени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

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

ОПИС РЕШЕЊА ПРОБЛЕМА

Непријављени корисник

Ови корисници имају могућност регистрације и могућност пријаве у апликацију. Приликом регистрације попуњава се форма за регистрацију.



Слика 3.1.1. Форма за регистрацију

Сви подаци морају бити валидно унети. Лозинка се у бази чува у хешираном формату помоћу *Bcrypt.Net*[8] алгоритма. Имплементација регистрације корисника на апликацију је приказана на слици испод.

```
public async Task<UserDto> Register(RegisterDto registerDto)
{
    List<User> users = await userRepo.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 = IMapper.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 == EUserType.SALESMAN)
        newUser.Verification = EVerificationStatus.INPROGRESS;
    else
        newUser.Verification = EVerificationStatus.ACCEPTED;
    UserDto dto = IMapper.Map<User, UserDto>(await userRepo.Register(newUser));
    return dto;
}
```

Слика 3.1.2. Имплементација кода регистрације корисника.

Приликом пријаве у апликацију попуњава се форма за пријаву у коју корисник уноси важећи мејл и лозинку

Слика 3.1.3. Форма за пријаву

Корисник може да се пријави преко *Google mail* налога или директним уносом мејла и лозинке. Уколико корисник не поседује налог, понуђен је линк за редирекцију на страницу регистрације.

```
2 references
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 3.1.4. Имплементација кода пријаве на апликацију

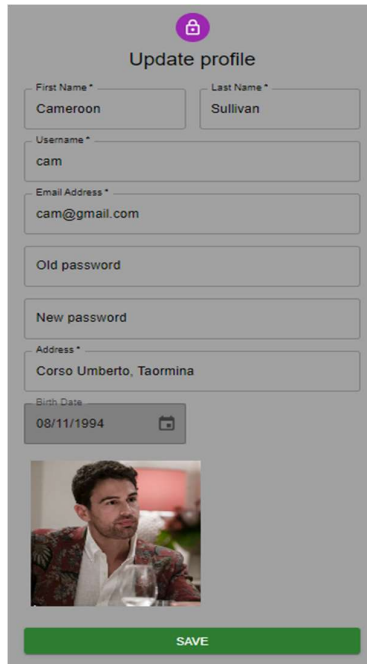
Валидација података се ради на серверској и на клијентској страни. Уколико је корисник унео погрешне податке добиће поруку о грешци, уколико је корисник унео валидне податке, ти подаци се проверавају тако што се тражи подударање мејла, а лозинка се криптује са *Bcrypt.Net*, пореди са криптованом вредношћу 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 mail-a* аутоматски се пријављује на систем. Имплементација регистравања путем *Google* налога је приказана на слици испод.

```
2 references
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 3.1.5. Регистровање корисника преко *Google mail-a*.

Пријављени корисник

Свим пријављеним корисницима је дозвољена измена профила. Може се извршити измена неког атрибута или целог профила. У случају неважећег поља приказује се порука о грешци.



Update profile

First Name *
Cameroon

Last Name *
Sullivan

Username *
cam

Email Address *
cam@gmail.com

Old password

New password

Address *
Corso Umberto, Taormina

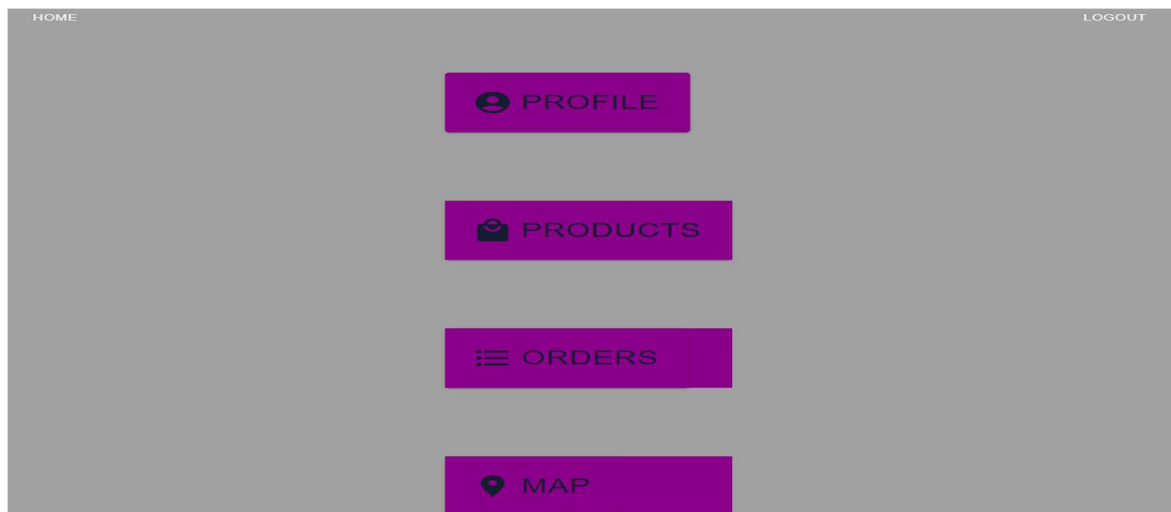
Birth Date
08/11/1994

SAVE

Слика 3.2.1. Измена профила



Продавац

Улога продавца је да креира нови производ.



Слика 3.3.1. Почетна страница корисника типа продавац

Продавац може да види листу производа које је направио и да управља истим.

ID	Image	Name	Description	Amount	Price	ADD NEW PRODUCT
3		Limun	Svez limun	45	5	CHANGE DELETE
5		Lubenica	Sveza lubenica	100	15	CHANGE DELETE

Слика 3.3.2. Листа артикала продавца


Кликом на дугме за додавање новог артикла појављује се прозор за унос података о новом артиклу који продавац **жели** да креира.

Name *

Description *

Amount *

Price *



NO IMAGE AVAILABLE

ADD CANCEL

Слика 3.3.2. Креирање новог артикла

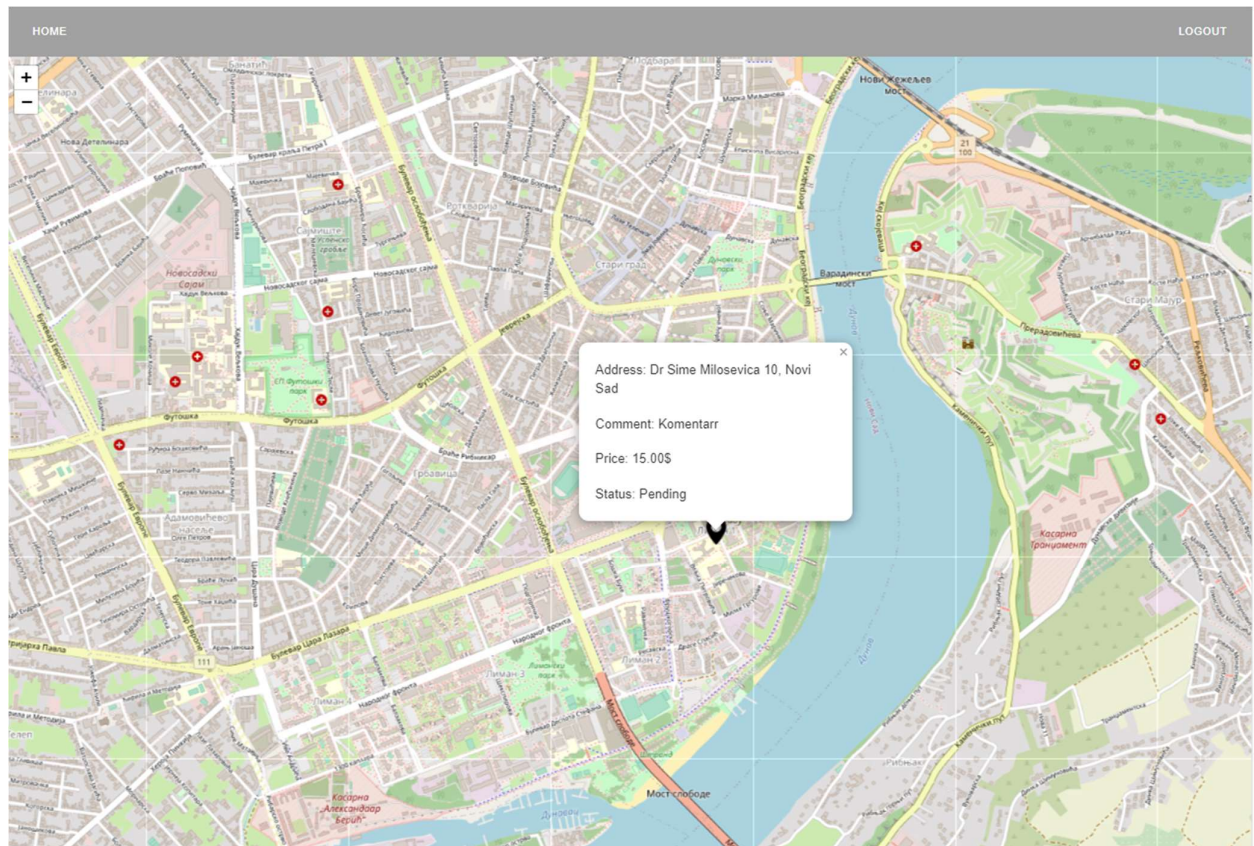
Продавац такође може да врши измену или брисање артикала употребом дугмета *Change* или *Delete*. Продавац може да види листу свих поруџбина на којој се налази бар један његов производ. У листи поруџбина може видети све детаље везано за његове поруџбине.

HOME							LOGOUT
	Id	Comment	Address	Price	Order Time	Delivery Time	Status
✓	19	Komentarrrr	Pezos	15	2023-08-23T17:12:44		INPROGRESS
							APPROVE

Слика 3.3.4. Листа поруџбина продавца.

Продавац може да одобри поруџбину и тада креће одбројавање до доставе артикла купцу. Ако купац откаже поруџбину продавац неће моћи да је види.

Сваки продавац може да отвори мапу и види своје поруџбине са тачном локацијом доставе. Кликком на неку поруџбину на мапи приказаће му се сви детаљи о тој поруџбини.



Слика 3.3.5. Мапа са поруџбинама.



Купац

Главна функционалност купца је поручивање артикала и самим тим и прављење поруџбина. Купац може да види све своје поруџбине.

HOME							LOGOUT
	Id	Comment	Address	Price	Order Time	Delivery Time	Status
✓	14	yyyyyyyyyyyyyy	Bulevar Evrope 15, Novi Sad	1128	2023-08-22T12:49:51		INPROGRESS
✓	20	Komentarr	Dr Sime Milosevica 10, Novi Sad	15	2023-08-23T17:21:51		INPROGRESS

Слика 3.4.1. Поруџбине купца.

Купац може да купује артикле које су креирали продавци. На слици испод је приказан изглед прозора приликом куповине производа где се кликом на дугме *My cart* прелази на прозор са корпом одакле се завршава поруџбина артикала.

HOME						LOGOUT
ID	Image	Name	Description	Amount	Price	MY CART
1		Limun	Limunnn	43	450	ADD TO CART
5		Lubenica	Sveza lubenica	99	15	ADD TO CART

Слика 3.4.2. Креирање поруџбине.

My cart

Product	Price	Amount	
Lubenica	15 RSD	x1	<div>+</div> <div>-</div>

Price: 15\$

Delivery price: 20\$

ORDER

CLOSE

Слика 3.4.3. 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 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 промени неке податке пре саме потврде поруџбине.

Checkout

Address *

Comment

ORDER

CANCEL

PayPal

Debit or Credit Card

Powered by PayPal

Слика 3.4.4. Наплата поруџбине

Уколико су сва поља валидно попуњена креира се поруџбина. Статус поруџбине се ставља *Inprogress* 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.

Администратор

Овај тип корисника је директно убачен у базу података. Када се корисник типа администратор пријави у апликацију има приступ опцијама за преглед и измену профила, списак и верификација свих продаваца и списак свих поруџбина.

Администратор може да види листу свих поруџбина.

Такође, може да види списак свих продаваца.

ID	First name	Last name	Username	Email	Verification	Verificate
2	Pera	Peric	pera	pera@gmail.com	ACCEPTED	
3	Nikola	Nikolic	nikola	nikola@gmail.com	INPROGRESS	<button>ACCEPT</button> <button>DENY</button>

Слика 3.5.1. Листа и верификација свих продаваца

Сврха процеса верификације је да се купцима омогући приступ функционалностима апликације. Порука о статусу верификације стиже купцима на мејл.

Имплементација кода за слање мејла продавцу се налази на слици испод.

```
3 references
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 3.5.2. Потврда верификације мејлом.

ПРЕДЛОЗИ ЗА ДАЉА УСАВРШАВАЊА

- Како би побољшали перформансе фронтенд дела апликације може се користити *Vite* уместо *create-react-app*.
- Омогућити корисницима да поред коментара остављају рецензије које ће садржати и оцене.
- Функцију мапе би било пожељно проширити тако да је користе и купац и продавац како би у сваком тренутку могли да виде где се њихове поруџбине налазе и где треба да стигну.
- У циљу усавршавања апликације било би пожељно омогућити превод странице на више језика које би корисник могао да користи и могућност одабира језика.
- Требало би омогућити респонзивни дизајн апликације тако да корисници могу да је користе на различитим уређајима укључујући и мобилне телефоне.
- Може се имплементирати реакција на догађај заборављене лозинке.
- Анализирати перформансе апликације и размислити о оптимизацијама кода и инфраструктуре како би апликација била што бржа.
- Наставити са тестирањем апликације, укључујући и аутоматске тестове. Одржавати зависности ажурним и пратити нове верзије коришћених технологија.

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>