Web aplikacija za upravljanje NFT-ovima te kriptovalutama

Adžić, Karlo

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

Sveučilišni studij

Web aplikacija za upravljanje NFT-ovima te kriptovalutama

Diplomski rad

Karlo Adžić

Osijek, 2023.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 19.09.2023.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Karlo Adžić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1182R, 07.10.2021.
OIB studenta:	28595580840
Mentor:	izv. prof. dr. sc. Mirko Köhler
Sumentor:	,
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	izv. prof. dr. sc. Ivica Lukić
Član Povjerenstva 1:	izv. prof. dr. sc. Mirko Köhler
Član Povjerenstva 2:	Miljenko Švarcmajer, mag. ing. comp.
Naslov diplomskog rada:	Web aplikacija za upravljanje NFT-ovima te kriptovalutama
Znanstvena grana diplomskog rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	Rezervirano za studenta: Karlo Adžić. Web aplikacija koja omogućuje općenito upravljanje NFT-ovima te kriptovalutama. Potrebno je omogućiti organizaciju i upravljanje njihovom kupnjom, prodajom, kreiranjem, trgovanje te cjelokupni pregled njihove statistike.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Vrlo dobar (4)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 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:	19.09.2023.

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, 01.10.2023.

Ime i prezime studenta:	Karlo Adžić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1182R, 07.10.2021.
Turnitin podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: Web aplikacija za upravljanje NFT-ovima te kriptovalutama**

izrađen pod vodstvom mentora izv. prof. dr. sc. Mirko Köhler

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

U današnje vrijeme svakodnevno sudjelujemo i osviješteni smo napretkom tehnologije. Od automobilske industrije, različitih razvojnih postrojenja, mobline industrije i sličnog do kriptovaluta i NFT tokena koji će promjeniti način trgovanja i upravljanje proizvoda u budućnosti. Takva nova tehnologija označava blockchain tehnologiju. Kriptovalute predstavljaju virtualne valute koje su osigurane kriptografijom, disciplinom koja se bavi pretvaranjem razumljive informacije u određenu kriptiranu informaciju ili određenu šifru te ju mogu razumjeti samo one osobe koje ju znaju dešifrirati. S druge strane, NFT (eng. non-fungible token) tokeni predstavljaju unikatne ili nezamjenjljive tokene, dio blockchain tehnologije, koji su također zaštićeni određenom kriptografijom kako bi bili sigurni za trgovanje i njihovo upravljanje. Mogu predstavljati određeno digitalno vlasništvo ili imovinu kao što su kolekcionarski predmeti, glazbena djela, digitalne slike, videozapisi i slično. Iako su obje tehnologije bazirane na blockchain tehnologiji, glavna razlika je da su tokeni (NFT) jedinstveni i ne može ih se zamjeniti s nekim drugim tokenom, što predstavlja određenu zainteresiranost kupaca odnosno klijenata, dok je određenu kriptovalutu moguće zamjeniti za neku drugu ili za određenu svotu novca. Vrijednosti kriptovaluta i tokena se konstantno mijenjaju tijekom vremena te zbog toga predstavljaju jednu od najzanimljivih tehnologija koje je potrebno razumijeti i pratiti. Razvitkom ove tehnologije dolazi i do razvoja web aplikacija i portala koje omogućuju upravljanje kriptovalutama i tokenima kroz jednostavna sučelja koja su omogućena krajnjem korisniku. Tržište ovim tehnologijama je u svakodnevnom razvoju te predstavljaju važan ekonomski čimbenik. Osobe mogu upravljati njima u cilju razumijevanja nove tehnologije ili zbog isključivog pokušaja zarade novca. Cilj ovog diplomskog rada je izrada web aplikacije za upravljanje NFT-ovima te kriptovalutama pod nazivom SIGMA, koji će u konačnici to i omogučiti krajnjim korisnicima. U drugom poglavlju ovog rada navedeno je nekoliko današnjih aplikacija koje imaju slične funkcionalnosti te koje se nalaze u produkciji dostupne krajnjem korisniku. Unutar trećeg poglavlja opisane su određene tehnologije izrade koje su korištene kroz izradu projekta. Zatim, kroz četvrto poglavlje definirani su svi načini izrade pojedinog dijela ove web aplikacije, dok se unutar petog poglavlja definira na koji način su ti dijelovi aplikacije, odnosno projekta, povezani u cjelinu. Na kraju, unutar šestog poglavlja, slikama i vizualnim prikazima je objašnjeno kako se zapravo aplikacija koristi te koje su sve njene kodirane funkcionalnosti.

1.1. Zadatak diplomskog rada

Web aplikacija koja omogućuje općenito upravljanje NFT-ovima te kriptovalutama. Potrebno je omogućiti organizaciju i upravljanje njhovom kupnjom, prodajom, kreiranjem, trgovanje te cjelokupni pregled njihove statistike.

2. PREGLED PODRUČJA TEME

Prilikom kreiranja određene aplikacije ili općenito novog projekta potrebno je ono što se izrađuje usporediti s već postojećim ili sličnim tehnologijama koje se nalaze u produkciji na tržištu i koje su dostupne za korištenje krajnjim korisnicima. Kao što je u naslovu rada navedeno, za potrebe ovog diplomskog rada bit će kreirana web aplikacija stoga je potrebno isto tako uspoređivati projekt s ostalim web portalima.

2.1. Binance

Binance ili Binance Holdings Ltd. je tvrtka koja je osnovana 2017. godine u Kini. To je tvrtka koja je uspostavila svoje poslovanje na globalnoj razini te predstavlja jednu od najvećih burza kriptovaluta u smislu njihove prodaje i trgovanja. Na web aplikaciji postoji sučelje koje omogućava razjmenu, kupovanje i prodaju kriptovaluta te unikatnih tokena (NFT) i brojne druge načine trgovanja.

Slika 2.1. Binance Holdings Ltd. logo

Omogućuje prikaz cijena svih kriptovaluta i tokena, te praćenje kretanja cijena u realnom vremenu. Isto tako, moguće je pratiti svaku kriptovalutu na bazi grafa, odnosno mijenjanje njenog iznosa iz sekunde u sekundu (rast i pad), kreirati uplate i isplate s računa u nekoj od novčanih jedinica kao što je euro ili u obliku kriptovalute te mnoge druge brojne operacije koje su dostupne.

Slika 2.2. Primjer prikaza rasta i pada kriptovalute

2.2. Uphold

Uphold je tvrtka koja je osnovana 2015. godine te omogućava jednostavniju uporabu i upravljanje kriptovalutama s obzirom na Binance kroz jednostavnije sučelje za korisnike. Omogućava praćenje rasta i pada cijena kriptovaluta kroz jednostavne grafove i brojne informacije. Izgrađen je na bazi aplikacija koje upravljaju e-novcem te na taj način osigurava pristup transparentnim, pravednim i pistupačnim financijskim uslugama.

Slika 2.3. Uphold logo

3. KORIŠTENE TEHNOLOGIJE

Prilikom izrade ovog diplomskog rada odnosno projekta, korištene su brojne tehnologije i načini programiranja te povezivanja funkcionalnog koda. Kako bi aplikacija bila što skalabilnija i jednostavnija za korištenje, bilo je potrebno definirati i programirati odvojene funkcionalnosti te ih u konačnici povezati u jednu cijelinu. Neki od korištenih tehnologija odnosno alata su sljedeći: ASP.NET MVC framework korišten uz programski jezik C# za programiranje backend dijela aplikacije, Postman aplikacije koja omogućuje testiranje, slane zahtjeva te dobivanje odgovora na zahtjeve koristeći različite url-ove na endpoint-e koji su kreirani unutar sustava, lokalna baza podataka odnosno SQL Server baza podataka koja omogućuje pohranu svih podataka koje aplikacija koristi kako bi radila na pravilan način, te React framework za implementaciju frontend dijela aplikacije. Isto tako, korišteno je i MS Visual Studio razvojno okruženje koje omogućuje kreiranje svih potrebnih dijelova za izradu.

Slika 3.1. Prikaz .NET framework loga (backend dio aplikacije)

Slika 3.2. Prikaz React loga (frontend dio aplikacije)

3.1. Programski jezik C#

Kao što je već navedeno, za izradu backend dijela projekta odnosno aplikacije korišten je programski jezik C#. Programski jezik C# kreiran je od strane Microsoft-a te služi kao nadogradnja na programske jezike C i C++. Koristi se u mnogo segmenata i grana izrade robusnih i sigurnih aplikacija, kao što su razvoj web aplikacija, razvoj desktop aplikacija, korištenje prilikom razvoja gaming industrije i slično. U potpunosti je objektno-orijentiran stoga omogućuje rad s različitim klasama, objektima, modelima i drugim tehnologijama. U ovom slučaju, korišten je za izradu web aplikacije uz WebAPI tehnologiju koja je implementirana unutar samog programskog jezika. Od svog nastanka, 2001. godine, dodane su brojne poboljšane značajke za podršku na različitim platformama, popularan je u današnjici te ga koriste brojni razvojni programeri i korisnici. Jednostavan C# programski kod, koji u konzolu nakon pokretanja programa ispisuje *"Hello World"*, izgleda ovako:

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine("Hello World");
    }
}
```

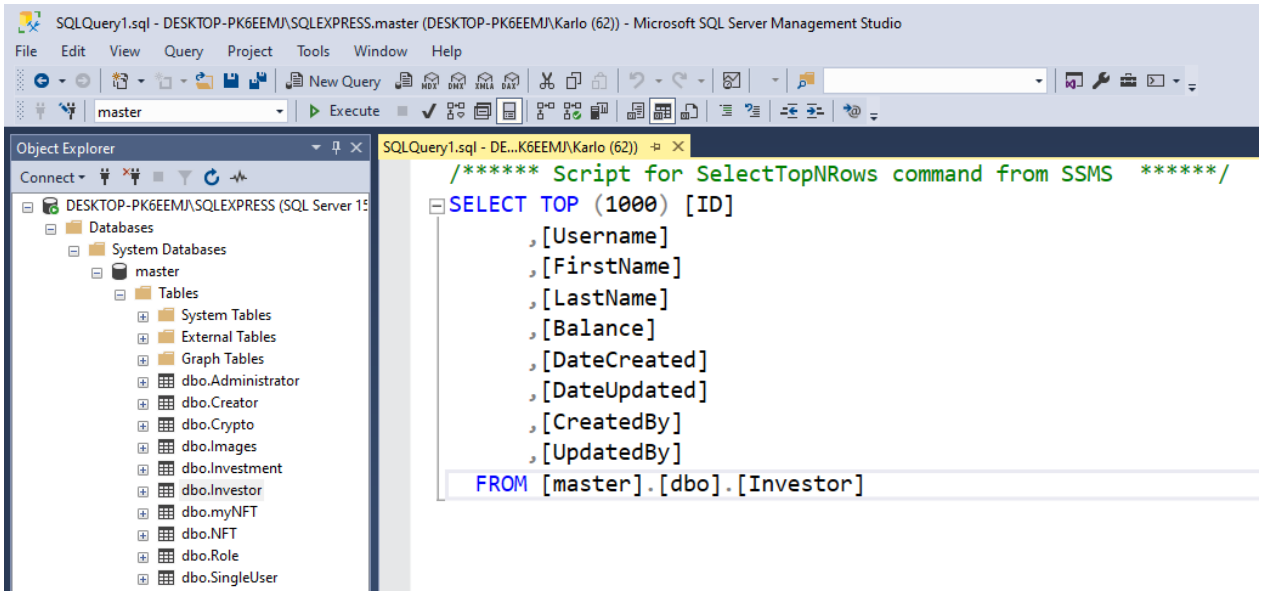
3.2. ASP.NET framework

ASP.NET nudi tri okvira ili framework-a za izradu web aplikacija a to su Web Forms, ASP.NET MVC te ASP.NET Web Pages. S obzirom na planirane i razređene funkcionalnosti web aplikacija korišten je ASP.NET MVC okvir. Ovakav način izrade web aplikacija koristi MVC programski i arhitekturalni obrazac (eng. Model-View-Controller) koji osigurava razvoj zasebnih odnosno pojedinih dijelova funkcionalnosti aplikacije te kasnije implementiranje u kompletnu funkcionalnost tj. web aplikaciju. Prilikom izrade ovog projekta, korišteni su mnogi modeli koji predstavljaju osnovu čijim entitetima se upravlja na web aplikaciji, kontroleri u kojima su kreirani endpoint-i (krajnje točke) odnosno funkcije za obavljanje određenog dijela funkcionalnosti sustava ili izmjenu podataka koji se koriste te prikazi određenog dijela aplikacije koje vidi krajnji korisnik. Uz sve to, ASP.NET framework omogućava kreiranje multilayer arhitekture koja predstavlja

kreiranje određenih programskih slojeva unutar aplikacije gdje se svaki od njih koristi za neku određenu funkcionalnost ili komunikaciju s drugim slojevima.

3.3. SQL baza podataka

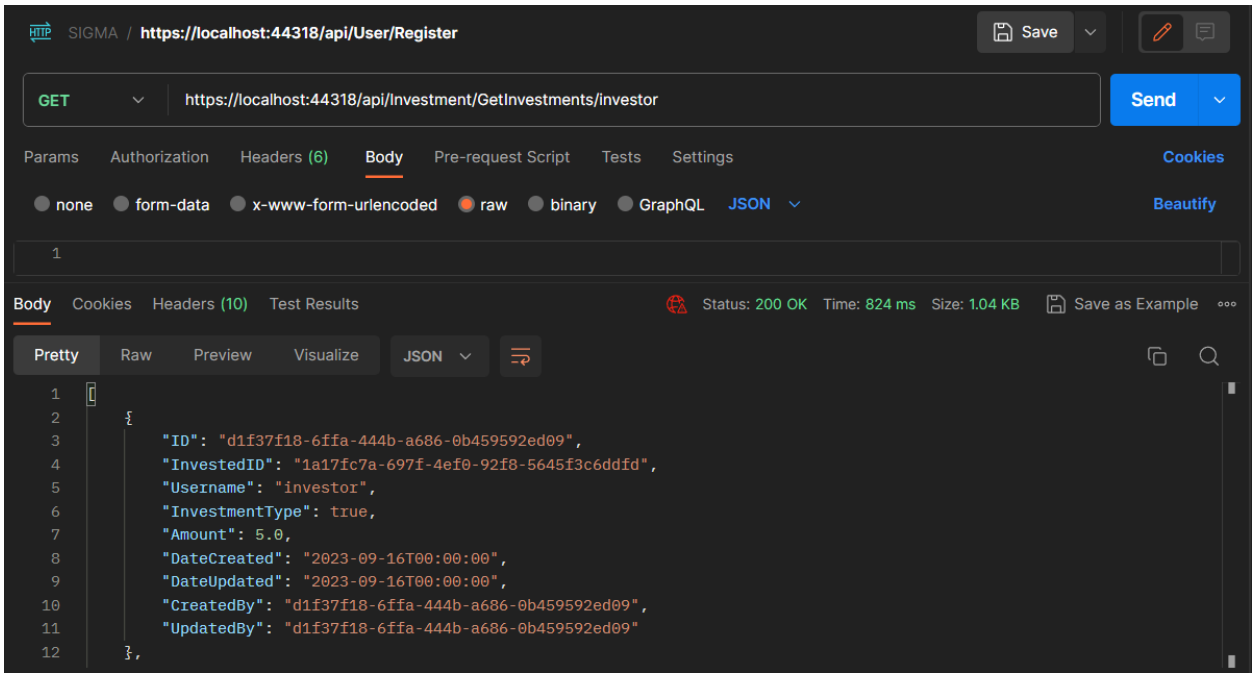
Uz kreiranje funkcionalnosti web aplikacije, potrebno je i osigurati način na koji će se pohranjivati i gdje će se pohranjivati svi potrebni podaci s kojima aplikacija upravlja. Na primjer, to može biti pohranjivanje novih registriranih korisnika, dohvaćanje njihovih ključnih informacija prilikom njihovom logiranja u sustav kao što su korisničko ime, lozinka i slično. Za potrebe ovog diplomskog rada korištena je lokalna baza podataka, odnosno SQL Server baza podataka. SQL Server baza podataka kreirana je od strane Microsoft-a te predstavlja relacijsku bazu podataka koja omogućava kreiranje različitih tablica koje su potrebne unutar aplikacije, zapis podataka unutar tablica, komunikaciju između različitih tablica unutar sustava te u konačnici cjelokupnu implementaciju sustava podataka. Svaki od tih podataka moguće je mijenjati i dohvaćati korištenjem SQL jezika odnosno korištenjem njegovih naredbi. Za konkretan rad s podacima, korišteno je razvojno okruženje SQL Management Studio (SSMS) koji omogućuje upravo ono što je potrebno za ovakav razvoj web aplikacija.



Slika 3.3. Prikaz dijela razvojnog okruženja SQL Server Management Studio

3.4. Postman

Prilikom kreiranja različitih endpoint-a odnosno funkcija koje šalju određene zahtjeve i primaju odgovor na takve zahtjeve, a još ne postoji frontend dio aplikacije, potrebno je imati način na koji se određene funkcionalnosti backend dijela mogu testirati. Upravo postman aplikacija omogućuje slanje zahtjeva i dobivanje ili primanje odgovara na zahtjev koristeći određene HTTP metode kao što su GET, POST, PUT, DELETE, UPDATE i slične, te url-ove koji su kreirani unutar sustava. Neke od funkcija kao zahtjev mogu samo dohvaćati određene podatke ali isto tako, neke od funkcija mogu poslati određene podatke koje je potrebno obraditi ili izmjeniti te ih kao takve vratiti ili dohvatiti nazad za prikaz krajnjem korisniku. Postoji dva načina slanja podataka, jedan kroz body funkcije, odnosno postman aplikacije a drugi koristeći url.



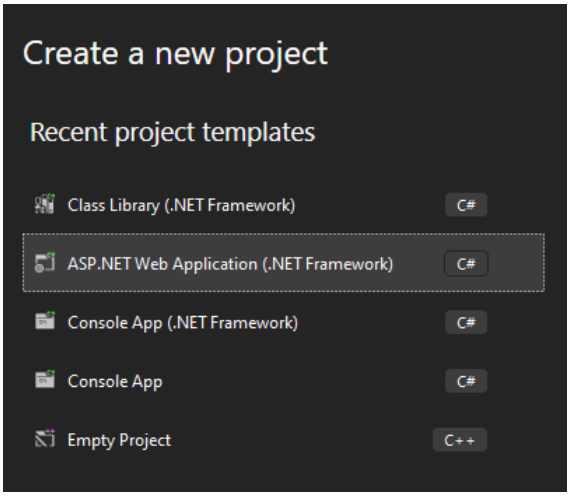
Slika 3.4. Prikaz jednostavnog slanja zahtjeva i dobivanja odgovora na zahtjev kroz Postman

4. IZRADA WEB APLIKACIJE

U ovom poglavlju definirani su načini izrade aplikacije korištenjem određene tehnologije. Prilikom kreiranja svake web aplikacije, potrebno je imati backend dio aplikacije koji omogućava komunikaciju s bazom i izmjene određenih podataka, te frontend dio aplikacije koji omogućuje jednostavan i lijep prikaz funkcionalnosti za krajnjeg korisnika. Prije kreiranja dijelova aplikacije potrebno je kreirati repozitorij na github servisu kako bi mogli osigurati svaku promjenu na aplikaciji. Korištenjem Git Bash programa na lokalnom računalu, upravljamo svim kreiranim datotekama i dokumentima unutar samog projekta te ih postavljamo na github servis. Lokalnim kloniranjem repozitorija dolazimo do funkcionalnosti te u konačnici do cjelokupnog koda aplikacije.

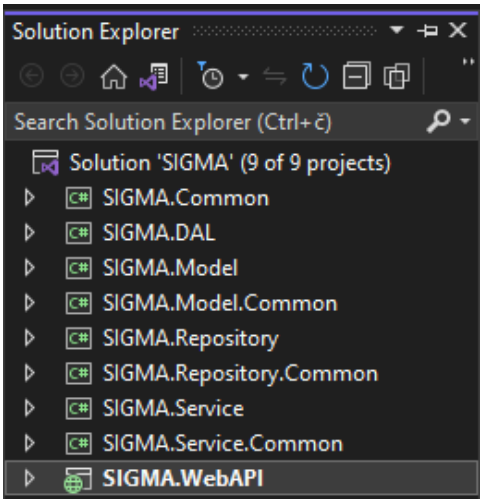
4.1. Backend dio web aplikacije

Kod same izrade web aplikacije i razrade te upravljanjem funkcionalnostima i podacima, potrebno je postaviti određene navedene tehnologije u obliku razvojnog okruženja. Za razvoj backend dijela aplikacije, kao što je već navedeno potrebno je kreirati inicijalni projekt unutar MS Visual Studio programa, podesiti određene postavke, te kreirati multilayer arhitekturu koja se u ovom slučaju sastoji od devet slojeva.



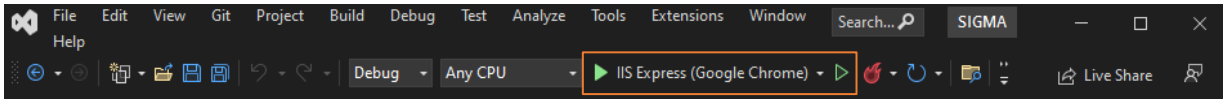
Slika 4.1. Prikaz ispravnog odabira prilikom kreiranja incijalnog projekta

Nakon kreiranja inicijalnog projekta, dolazimo do početnog zaslona programa u koji dodajemo različite slojeve arhitekture. Svaki sloj se dodaje na način da se unutar Solution Explorer-a dodaje novi projekt u obliku Class Library datoteke koja predstavlja svaki pojedini sloj.

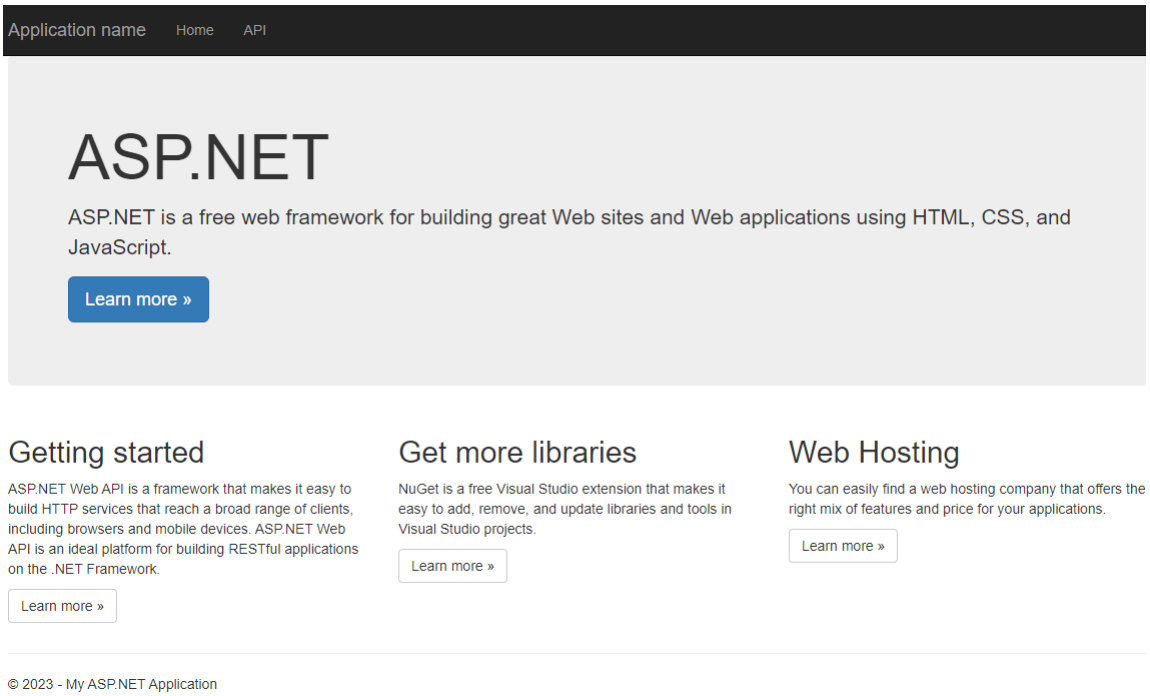


Slika 4.2. Prikaz dodanih slojeva unutar programa

Na kraju postavljanja svih dijelova backend dijela aplikacije, potrebno je pokrenuti i testirati inicijalni projekt. Pokretanje ovog dijela se odvija na način da se pokrene debugger unutar MS Visual Studio programa.



Slika 4.3. Prikaz pokretanja backend dijela aplikacije



Slika 4.4. Prikaz uspješnog pokretanja backend dijela web aplikacije

Ukoliko smo uspješno pokrenuli backend dio web aplikacije, otvorit će nam se web preglednik koji će izgledati kao što je prikazano na slici 4.4. To je određeni localhost server sa svojim kreiranim portom. U ovom konkretnom slučaju, on je **https://localhost:44318/**.

4.2. Frontend dio web aplikacije

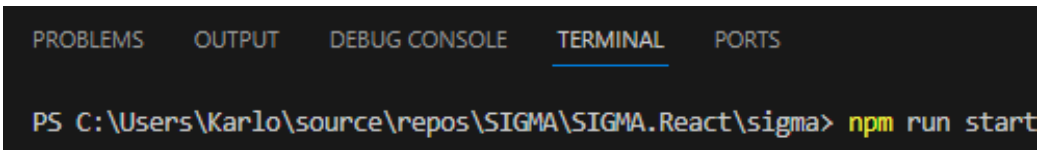
Nakon uspješnog postavljanja i razvoja backend dijela web aplikacije, potrebno je kreirati i postaviti frontend dio aplikacije koji u ovom diplomskog radu označava postavljanje React framework-a odnosno Visual Studio Code programa gdje će se kroz određene komponente prikazivati određene funkcionalnosti te koje će biti ugodne izgledom za krajnjeg korisnika. Korištenjem Git Bash programa koji omogućuje upravljanje podacima projekta, potrebno je pozicionirati se u određeni direktorij s već kreiranim backend dijelom aplikacije korištenjem određenih naredbi. Unutar istog direktorija kreira se novi React projekt koji predstavlja frontend dio aplikacije, a isto tako i deseti sloj multilayer arhitekture. Na taj način omogućili smo jednostavnu strukturu projekta te mogućnost brze i lake izmjene određenih datoteka i dokumenata.

Slika 4.5. Prikaz pozicioniranja unutar direktorija projekta

```
npx create-react-app my-app
cd my-app
npm start
```

Slika 4.6. Prikaz načina kreiranja novog React projekta te njegovog pokretanja

Unutar Visual Studio Code programa, otvaramo direktorij SIGMA.React te zatim ***sigma*** koji označava glavni kreirani direktorij frontend dijela aplikacije. Pokrećemo ga na način da unutar terminal-a upišemo naredbu **npm start** ili **npm run start**.

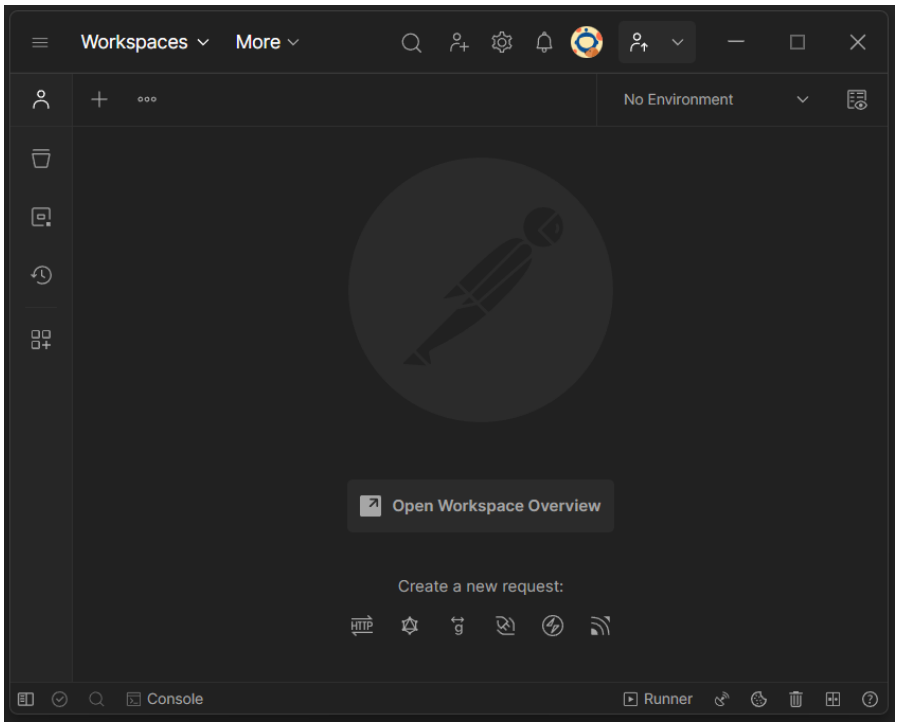


Slika 4.7. Prikaz pokretanja frontend dijela aplikacije

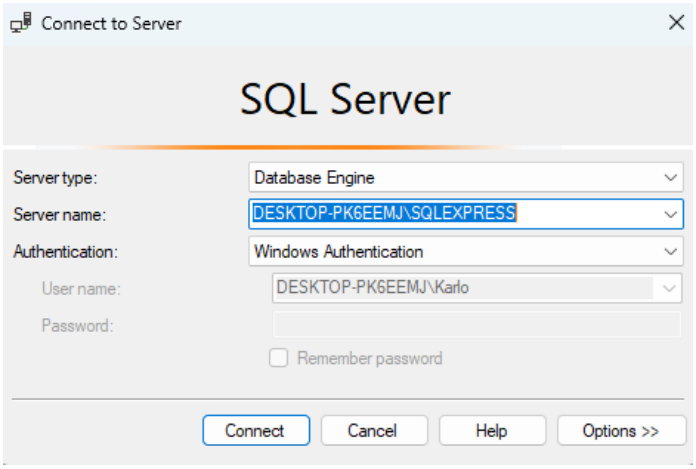
Nakon uspješnog pokretanja dobivamo konkretan prikaz elemenata aplikacije u web pregledniku kojima može upravljati krajnji korisnik.

4.3. Ostali dijelovi web aplikacije

Pod ostalima dijelovima postavljanja web aplikacije smatraju se dijelovi koji nisu vezani uz kodiranje nego uz određene druge funkcionalnosti i postavljanje cjelokupnog sustava. Potrebno je postaviti razvojno okruženje Postman njegovim instaliranjem i pokretanjem te lokalnu bazu podataka koje zahtjeva svoje povezivanje sa serverom prije početka rada.

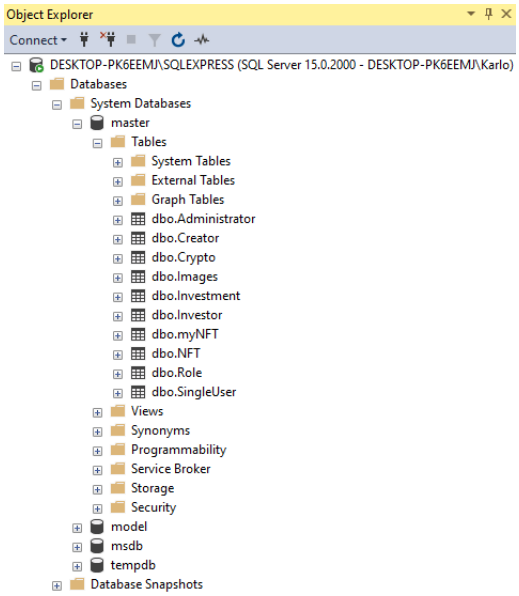


Slika 4.8. Prikaz osnovnog sučelja programa Postman



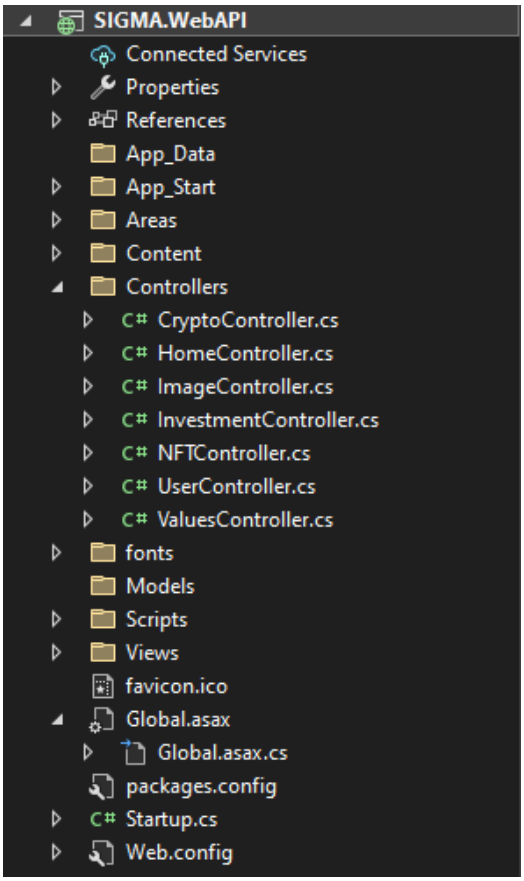
Slika 4.9. Prikaz povezivanja lokalne baze sa SQL serverom

Nakon povezivanja, dostupno je sučelje za korištenje i kreiranje određenih višebrojnih tablica i relacija među njima.



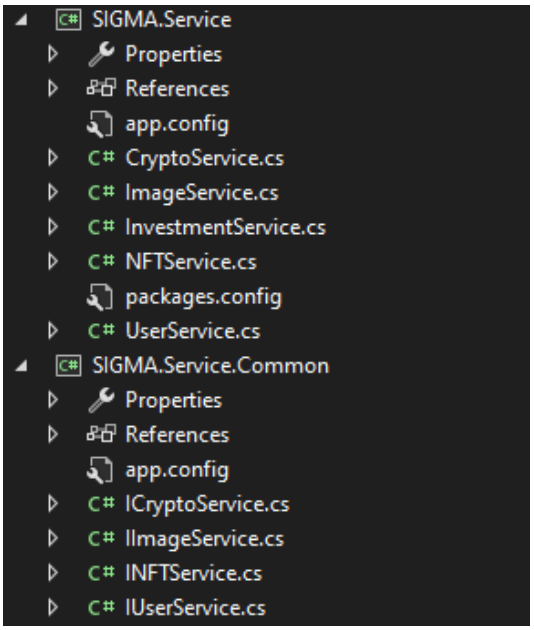
Slika 4.10. Prikaz stabla datoteka i direktorija unutar baze podataka

Kao što je već definirano, backend dio aplikacije se sastoji od devet slojeva arhitekture, te je potrebno objasniti način na koji se ti slojevi koriste te na koji način pomažu pri razvoju ovakve web aplikacije. Unutar početnog i glavnog sloja aplikacije pod nazivom SIGMA.WebAPI, kreirani su svi kontroleri koji su bili potrebni pri izradi ovog projekta te određeni direktoriji koji sadrže konfiguraciju inicijalnog sustava.

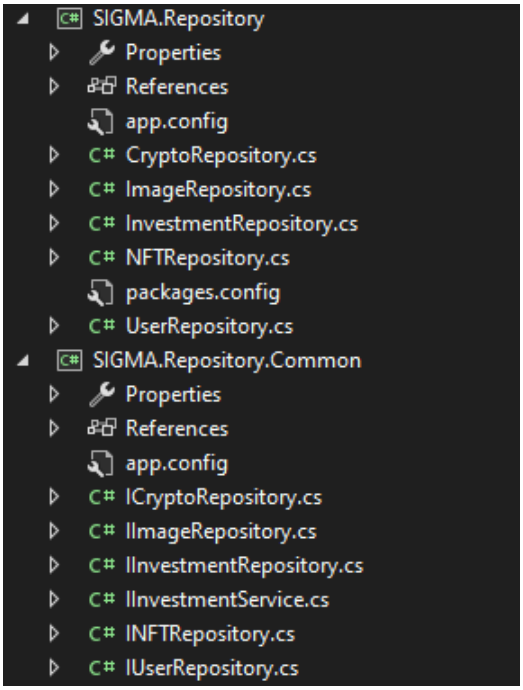


Slika 4.11. Prikaz sloja SIGMA.WebAPI unutar projekta

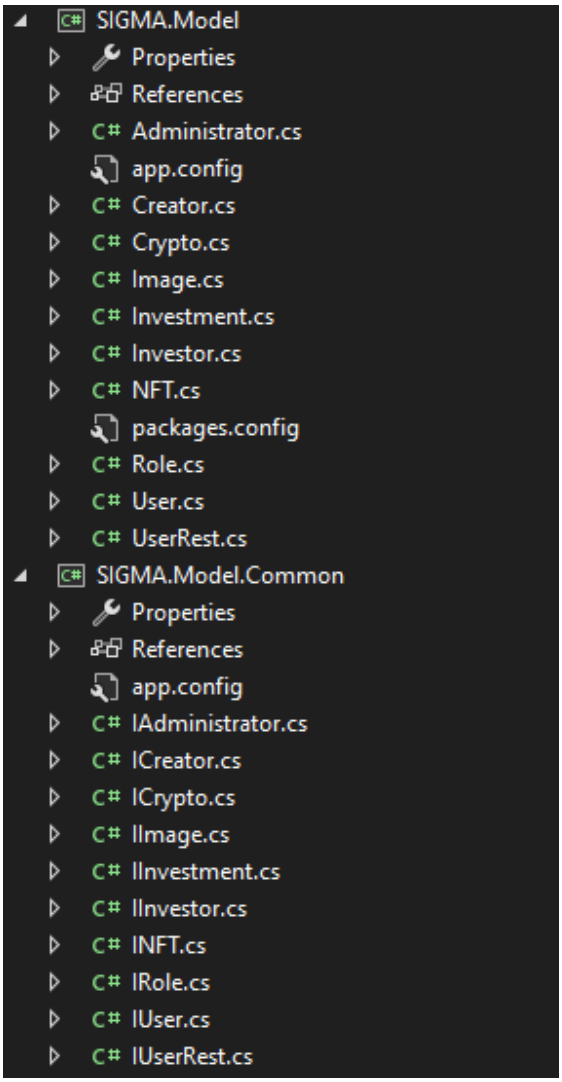
U sklopu cijelog projekta, korištena je tzv. dependency injection arhitektura programiranja, koja se u ovom slučaju ubrizgava kroz konstruktore unutar kontrolera, servisa i repozitorija, kako bi se smanjile međuovisnosti različitih komponenata aplikacije. U konkretnoj primjeni, to znači kreiranje sučelja (eng. interface) za svaki od modela, servisa te repozitorija i njihovo nasljeđivanje u konkretnim klasama istih.



Slika 4.12. Prikaz slojeva SIGMA.Service i SIGMA.Service.Common

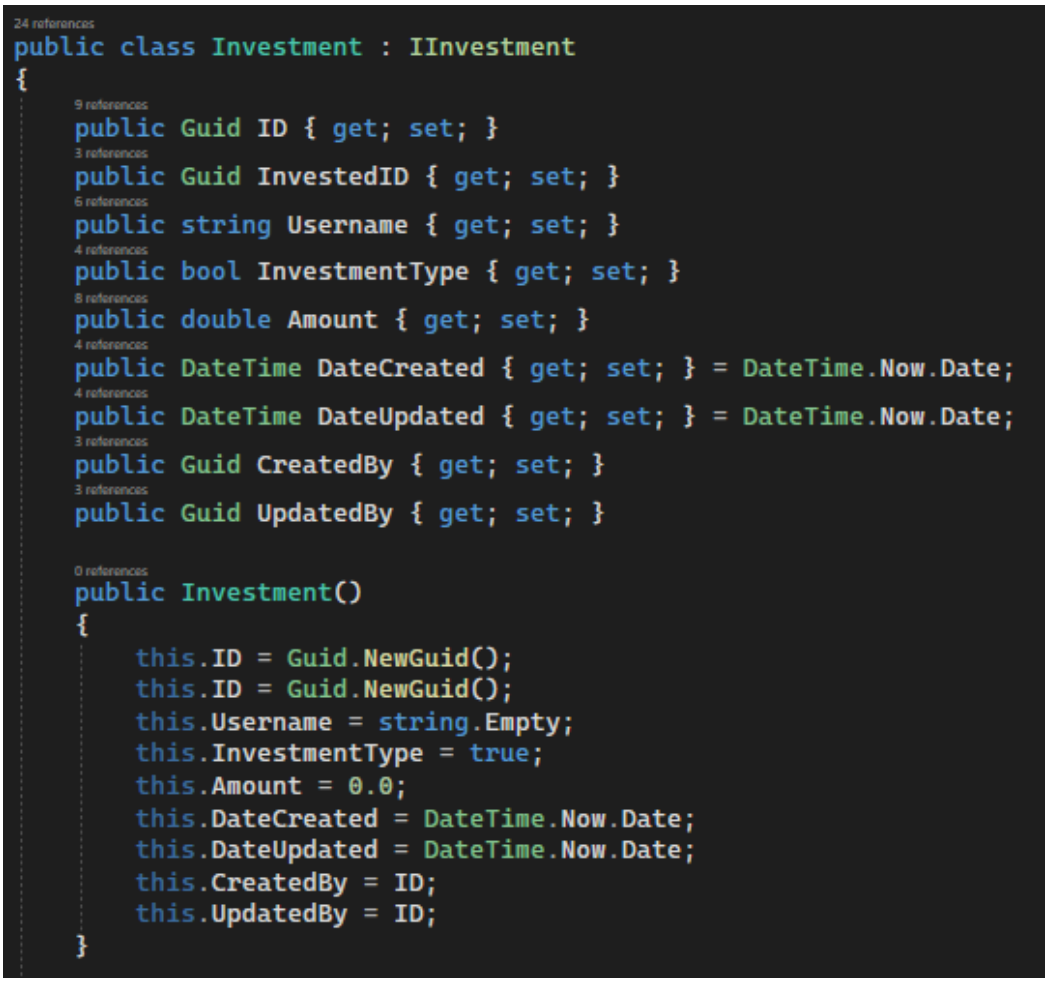


Slika 4.13. Prikaz slojeva SIGMA.Repository i SIGMA.Repository.Common



Slika 4.14. Prikaz slojeva SIGMA.Model i SIGMA.Model.Common

Kao što se može vidjeti na slikama 4.12., 4.13. i 4.14. svaka model klasa sadrži svoje sučelje unutar svog common sloja (eng. layer), svaka servis klasa sadrži svoje sučelje unutar svog common sloja te svaka repozitorij klasa sadrži svoje sučelje unutar svog common sloja.



Slika 4.15. Prikaz dijela koda Investment modela

Prilikom dodavanja novih funkcionalnosti unutar koda koji moraju sadržavati svoj model, servis i repozitorij, potrebno je u određene slojeve dodati navedeno te isto tako za svaki od njih kreirati sučelja u common slojevima i podesiti ih unutar SIGMA.WebAPI sloja, **Global.asax.cs** direktorija. Unutar njega definira se i konfigurira dependency injection za svaku od kreiranih klasa koristeći Autofac paket kojeg je prilikom programiranja bilo potrebno instalirati. Ovakav način rada baziran je na inverziji kontrole (eng. Inversion Of Control, IoC) koji omogućava lakšu izgradnju aplikacija koje se moraju i trebaju testirati te koje će se u budućnosti održavati.

```
protected void Application_Start()
{
    var builder = new ContainerBuilder();
    var config = GlobalConfiguration.Configuration;

    builder.RegisterType<NFT>().As<INFT>();
    builder.RegisterType<Crypto>().As<ICrypto>();
    builder.RegisterType<Administrator>().As<IAdministrator>();
    builder.RegisterType<Investor>().As<IInvestor>();
    builder.RegisterType<Creator>().As<ICreator>();
    builder.RegisterType<UserRest>().As<IUserRest>();
    builder.RegisterType<Investment>().As<IInvestment>();
    builder.RegisterType<Image>().As<IImage>();

    builder.RegisterType<UserService>().As<IUserService>();
    builder.RegisterType<NFTService>().As<INFTService>();
    builder.RegisterType<CryptoService>().As<ICryptoService>();
    builder.RegisterType<InvestmentService>().As<IInvestmentService>();
    builder.RegisterType<ImageService>().As<IImageService>();
```

Slika 4.16. Prikaz dijela koda unutar Global.asax.cs direktorija (DI)

4.4. Kreiranje endpoint-a aplikacije

Unutar ovog projekta, prilikom izgradnje web aplikacije razdvojene su funkcionalnosti kako bi smanjili ovisnosti prilikom korištenja i izmjena aplikacije. Jedan od ključnih načina postizanja takvog rada je kreiranje endpoint-a, odnosno krajnjih funkcija koje se nalaze u kontrolerima ovisno o onom s čime se treba rukovati. Na primjer, za upravljanje cijelim funkcionalnostima što se tiče modela korisnika (eng. User), kreirana je UserController klasa odnosno kontroler (eng. Controller) unutar SIGMA.WebAPI sloja unutar kojeg se nalaze endpoint-i koji upravljaju samim korisnikom. Jedan od takvih endpoint-a predstavlja registraciju korisnika. Svaki endpoint sadrži svoju rutu (eng. route) na kojoj se poziva ili izvršava ta funkcija i na kojoj se on može testirati unutar postman aplikacije ili bilo koje druge aplikacije koja to omogućuje, koja metoda se koristi unutar tog endpoint-a (HttpPost) te kod same funkcije. Unutar ovakve funkcije nalazi se referenca na njegov servis, a unutar servisa referenca na njegov repozitorij. Unutar repozitorija se odvija komunikacija s bazom podataka pomoću string-a za konekciju (eng. connection string) te se odrađuje konkretan posao (npr. izmjena podataka, dohvaćanje podataka, uređivanje podataka i slično).

```
string connString = @"Server = DESKTOP-PK6EEMJ\SQLEXPRESS; Database = master; Trusted_Connection = True;";
```

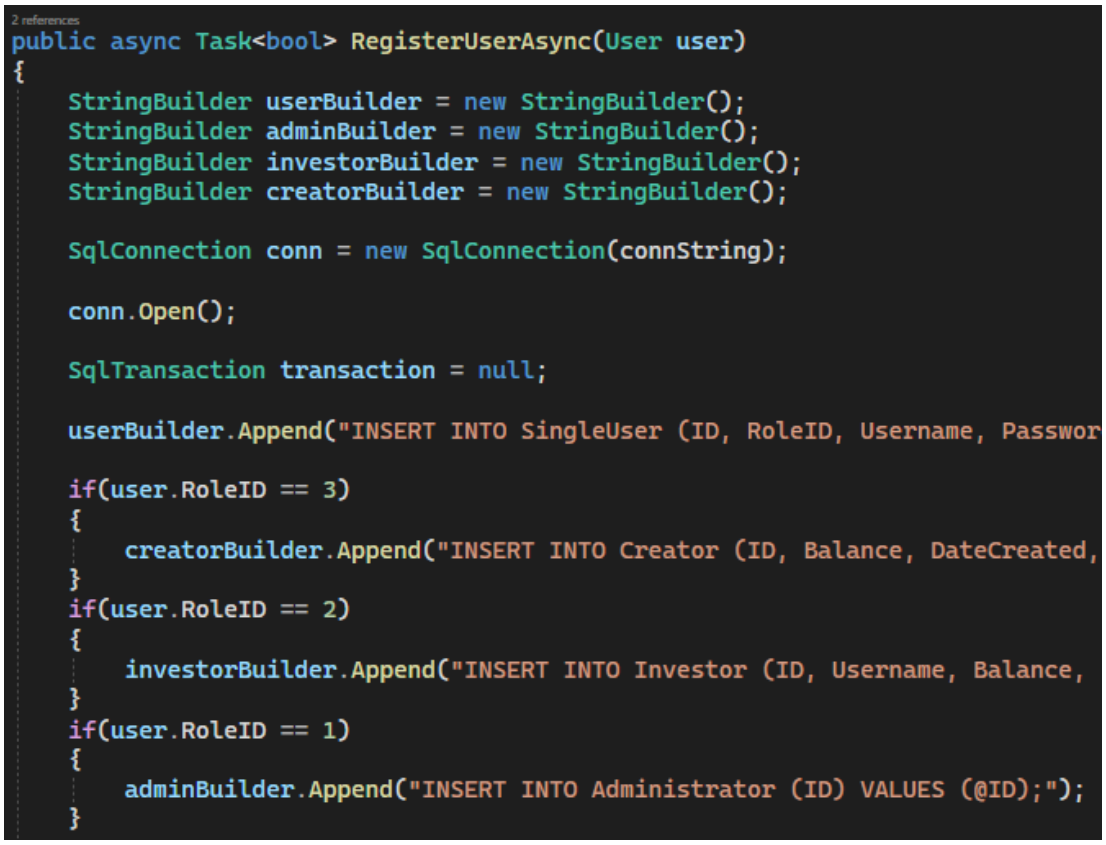
Slika 4.17. Prikaz string-a za konekciju (eng. connection string) s bazom podatka

```
[Route("api/User/Register")]
[HttpPost]
0 references
public async Task<HttpResponseMessage> RegisterUserAsync(User request)
{
    CreatePasswordHash(request.Password, out byte[] passwordHash, out byte[] passwordSalt);

    User user = new User();

    user.ID = request.ID;
    user.RoleID = request.RoleID;
    user.Username = request.Username;
    user.Password = request.Password;
    user.PasswordHash = passwordHash;
    user.PasswordSalt = passwordSalt;
    user.Email = request.Email;
    user.Investor.FirstName = request.Investor.FirstName;
    user.Investor.LastName = request.Investor.LastName;
    user.DateCreated = request.DateCreated;
    user.DateUpdated = request.DateUpdated;
    user.CreatedBy = request.CreatedBy;
    user.UpdatedBy = request.UpdatedBy;

    if(await userService.RegisterUserAsync(user) == true)
    {
        return Request.CreateResponse(HttpStatusCode.OK, user);
    }
    else
    {
        return Request.CreateResponse(HttpStatusCode.BadRequest, "Not registered!");
    }
}
```

Slika 4.18. Prikaz endpoint-a (funkcije) za registraciju korisnika

```
2 references
public async Task<bool> RegisterUserAsync(User user)
{
    return await userRepository.RegisterUserAsync(user);
}
```

Slika 4.19. Prikaz servisa (funkcije) za registraciju korisnika



```
2 references
public async Task<bool> RegisterUserAsync(User user)
{
    StringBuilder userBuilder = new StringBuilder();
    StringBuilder adminBuilder = new StringBuilder();
    StringBuilder investorBuilder = new StringBuilder();
    StringBuilder creatorBuilder = new StringBuilder();

    SqlConnection conn = new SqlConnection(connString);

    conn.Open();

    SqlTransaction transaction = null;

    userBuilder.Append("INSERT INTO SingleUser (ID, RoleID, Username, Passwor

    if(user.RoleID == 3)
    {
        creatorBuilder.Append("INSERT INTO Creator (ID, Balance, DateCreated,
    }
    if(user.RoleID == 2)
    {
        investorBuilder.Append("INSERT INTO Investor (ID, Username, Balance,
    }
    if(user.RoleID == 1)
    {
        adminBuilder.Append("INSERT INTO Administrator (ID) VALUES (@ID);");
    }
```

Slika 4.20. Prikaz dijela repozitorija (funkcije) za registraciju korisnika

5. POVEZANOST SUSTAVA NA RAZLIČITIM RAZINAMA

Kako bi određena kreirana aplikacija bila što jednostavnija za održavanje te kako bi bila skalabilna i robusna nakon svoje cijele implementacije, od kreiranja baze podataka, backend dijela aplikacije do kreiranja frontend dijela aplikacije, potrebno je ostvariti određenu povezanost između tih slojeva.

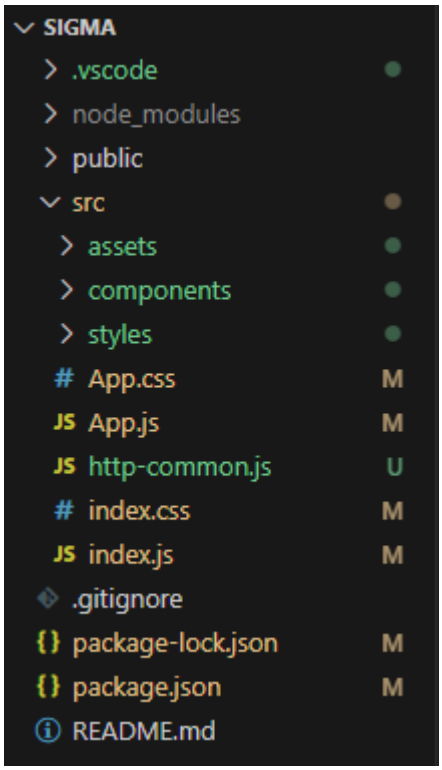
5.1. Povezivanje backend i frontend dijela aplikacije (Axios)

Nakon kreiranja određenih endpoint-a te njihovog testiranja unutar backend dijela aplikacije i unutar Postman programa, potrebno je implementirati korištenje zahtjeva i odziva na frontend dijelu aplikacije odnosno u ovom slučaju korištenjem React-a. Da bi to bilo moguće, bilo je potrebno instalirati paket Axios (unutar React sloja, odnosno na frontend strani aplikacije) koji predstavlja JavaScript biblioteku za kreiranje HTTP zahtjeva u različitim web aplikacijama. Dizajniran je za rad s preglednicima te je zbog toga pravi tehnološki alat za ovakvu funkcionalnost. Omogućava jednostavan način slanja zahtjeva s frontend strane aplikacije koristeći url-ove endpoint-a na backend strani aplikacije. Prije njegovog korištenja unutar React-a, potrebno ga je uvesti (eng. import) iz tzv. axios biblioteke koja je instalirana unutar direktorija **node_modules**.

```
axios
  .post("https://localhost:44318/api/User/Login", data)
  .then((res) => {
    setLoginData(JSON.stringify(res.data));
    localStorage.setItem("token", res.data);
    localStorage.setItem("username", loginData.Username);
    localStorage.setItem("roleID", loginData.RoleID);

    setTimeout(() => {
      navigate("/");
      window.location.reload();
    }, 1000);
  })
  .catch((e) => {
    alert(e.response.data);
  });
```

Slika 5.1. Prikaz primjera kreiranja POST zahtjeva na frontend dijelu aplikacije

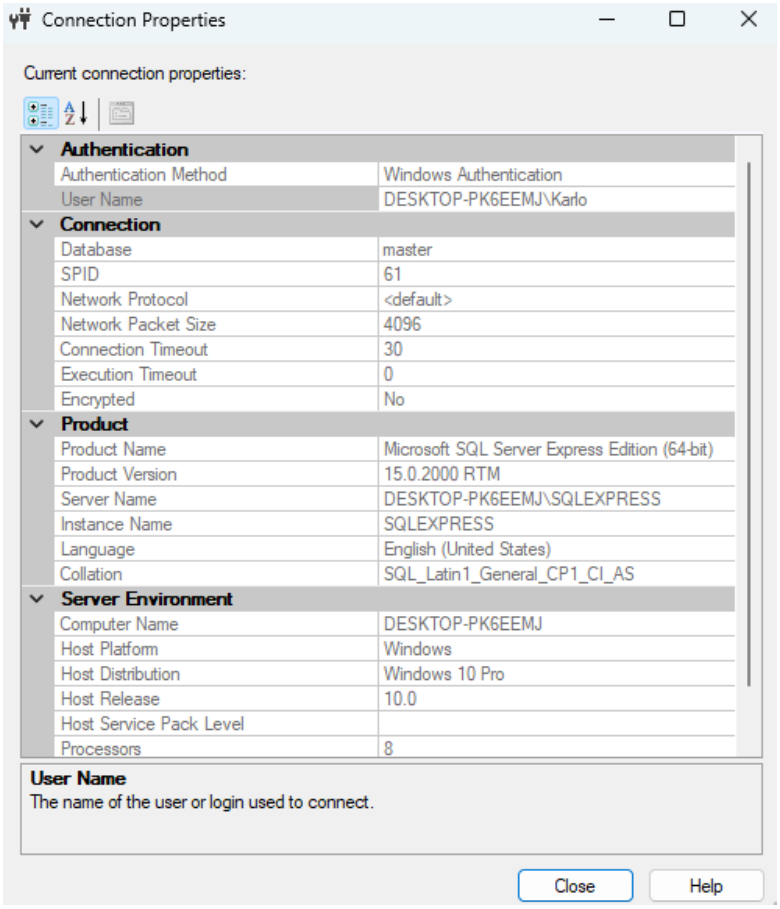
Na slici 5.1. prikazan je jednostavan POST zahtjev koji se nalazi unutar Login komponente na frontend dijelu aplikacije. Pomoću url-a, aplikacija šalje podatke backend-u na obradu i vraćanje odgovara na zahtjev, koje je korisnik uneo na frontend dijelu web aplikacije. Na primjer, korisnik u ovom slučaju unosi svoje korisničko ime i lozinku, ti podaci se šalju na backend dio aplikacije gdje će se oni obraditi, usporediti postoji li navedeni korisnik u bazi podataka, te će frontend dobiti odgovor. U slučaju da ne postoji takav korisnik, na frontend dijelu aplikacije će se ispisati validna poruka s obzirom na rezultat, a u slučaju da postoji takav korisnik, odgovor na zahtjev će biti kreiranje tokena s kojim će se korisnik ulogirati i moći se dalje koristiti web aplikacijom za ono što želi učiniti.



Slika 5.2. Prikaz svih direktorija i datoteka frontend dijela aplikacije

5.2. Povezivanje baze podataka i aplikacije

Da bi sama aplikacija mogla upravljati određenim podacima, potrebno je povezati backend dio aplikacije i lokalnu bazu podataka. Kao što je već navedeno, komunikacija na ovoj razini se odvija pomoću string-a za konekciju (eng. connection string). Njega je potrebno konfigurirati na temelju postavki unutar servera na koji je spojena baza podataka.



Slika 5.3. Prikaz svojstava servera baze podataka

Na slici 5.3. možemo vidjeti sva svojstva servera baze podataka. Prilikom kreiranja string-a od svojstava potrebno je koristiti ime baze podataka (Database), ime servera (Server Name) te postaviti zastavicu (eng. flag) da je ovakav tip povezivanja siguran odnosno da se ovom serveru može vjerovati (Trusted_Connection = True). Jednostavan primjer string-a za povezivanje (eng. connection string) izgleda ovako:

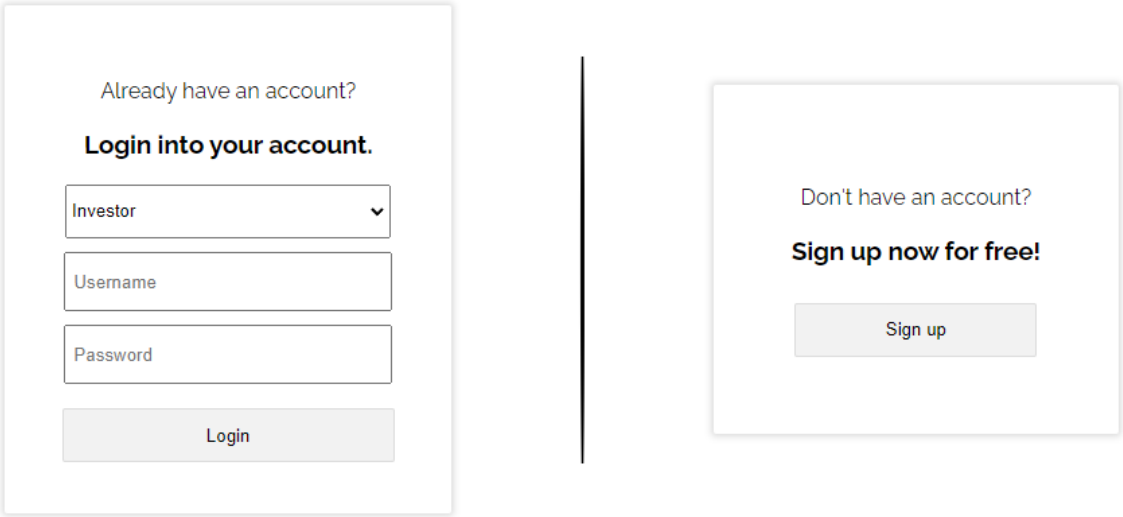
```
string connString = @"Server = DESKTOP-PK6EEMJ\SQLEXPRESS; Database = master;
Trusted_Connection = True;";
```

6. DIZAJN I FUNKCIONALNOST WEB APLIKACIJE

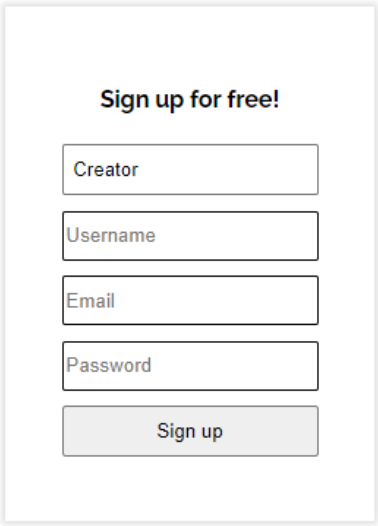
Prije pisanja koda aplikacije potrebno je dobro razmotriti i razraditi na koji način će se izvršavati određene funkcionalnosti te kako će određeni dio aplikacije biti najednostavniji za korištenje krajnjem korisniku. Isto tako, treba paziti na skalabilnost i jednostavnost tehničkog dijela, odnosno dijela koji kreira sam razvojni programer kako bi ta aplikacija bila što jednostavnija za održavanje. Primarne funkcionalnosti web aplikacije za upravljanje NFT-ovima i kriptovalutama bazirane su na prikazu onih podataka koji su potrebni s obzirom na registriranog i ulogiranog korisnika.

6.1. Registracija i prijava korisnika

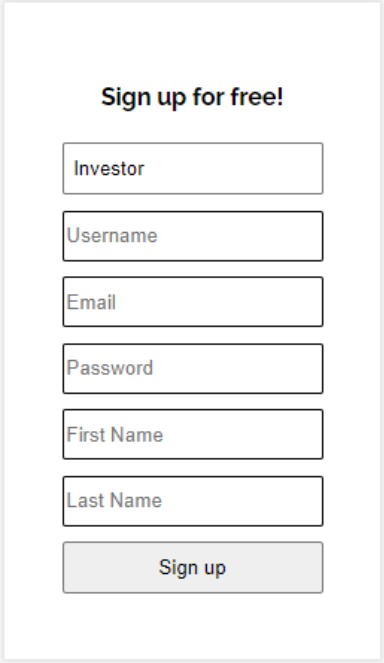
Korisnik može biti registriran kao investitor (eng. Investor) ili kao kreator (eng. Creator). Svakom investitoru prilikom registriranja omogućen je određeni iznos koji može investirati u određenu kriptovalutu ili određeni NFT, te istim novcem može kupiti ono što želi ukoliko je to na prodaju. Isto tako, svakom kreatoru omogućen je iznos koji može koristiti za kreiranje određene kriptovalute ili određenog NFT-a. Samim kreiranjem kriptovalute ili NFT-a kreator stavlja svoje proizvode na prodaju. Unutar sustava, s obzirom na ulogiranog korisnika, za svakog su prikazane drugčije funkcionalnosti, a na taj način se kreatora sprječava da pristupi onome što može napraviti i odraditi samo investitor te obratno.



Slika 6.1. Prikaz forme za prijavu/registraciju korisnika



Slika 6.2. Prikaz forme za registraciju kreatora



Slika 6.3. Prikaz forme za registraciju investitora

S obzirom na ono što korisnik želi biti, kreator koji će kreirati kriptovalute i NFT-ove te ih prodavati ili investitor koji će u to ulagati, postoje različite forme za registraciju. Korisnik se u osnovi (eng. default) prijavljuje kao investitor, te ukoliko on to nije dobit će obavijest da se ne može prijaviti. Isto vrijedi i za kreatora koji se želi prijaviti kao investitor. Na taj način osigurano je kreiranje i upravljanje novim korisnicima.

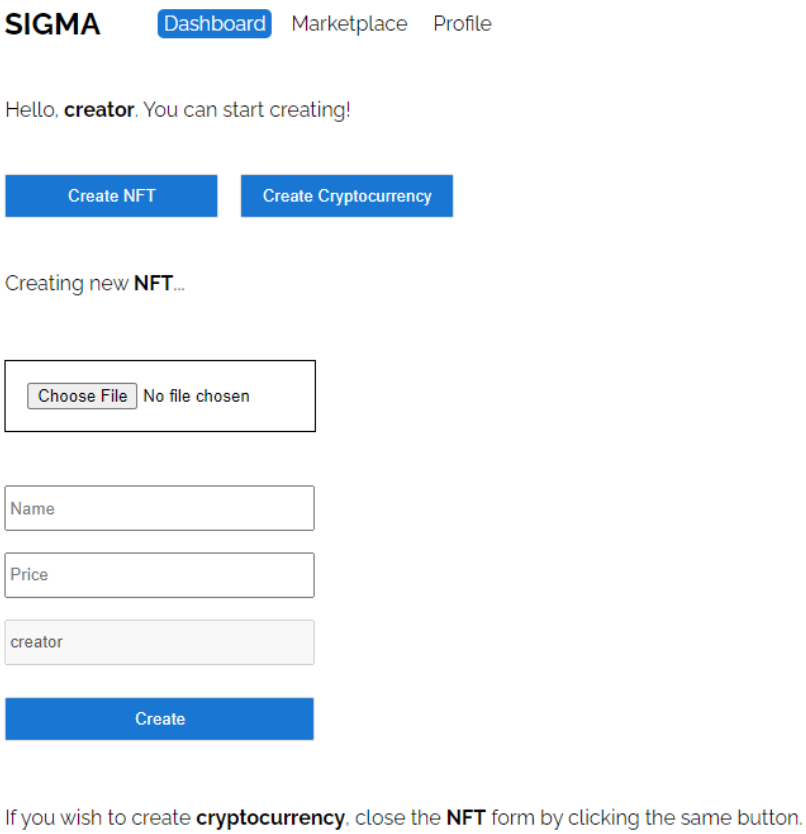
Nakon uspješne registracije, sustav korisnika vodi na zaslon za prijavu koji se može vidjeti na slici 6.1. Odabirom točne uloge i unosom točnih podataka za prijavu korisnik dolazi na početni zaslon web aplikacije. Ukoliko podaci za prijavu/registraciju nisu točni, korisnik će dobiti validnu obavijest s obzirom na ono što nije točno. Na primjer, ukoliko korisnik unese krivo korisničko ime, klikom na prijavu dobit će obavijest da je uneo krivo korisničko ime. Isto vrijedi za lozinku te odabir načina prijave.

Slika 6.4. Prikaz početnog zaslona aplikacije investitora (eng. Investor)

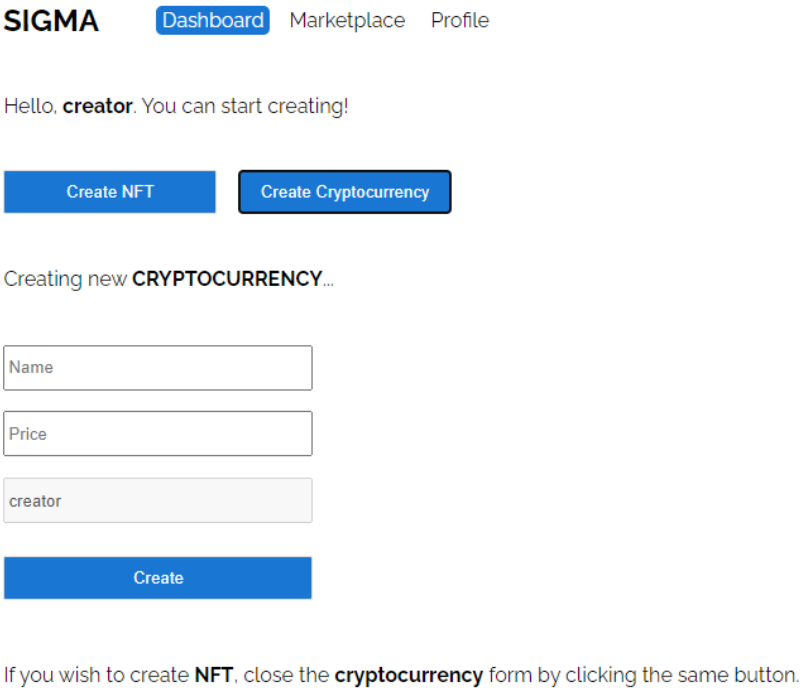
Slika 6.5. Prikaz početnog zaslona aplikacije kreatora (eng. Creator)

6.2. Kreiranje kriptovaluta i NFT-ova

Korsinici koji su prijavljeni kao kreatori mogu kreirati nove NFT-ove i nove kriptovalute. Kao što je prikazano na slici 6.5. postoje različite forme za kreiranje istih. Kreator klikom na gumb kreiraj NFT dinamički otvara formu za kreiranje NFT-ova. Ukoliko u tom trenutku ne želi kreirati NFT, može kreirati kriptovalutu zatvaranjem forme za kreiranje NFT-a klikom na isti gumb, te klikom na gumb kreiraj kriptovalutu otvara mu se forma za kreiranje kriptovaluta. Isti način upravljanja formama vrijedi i za kriptovalute. Kreator prilikom kreiranja novog NFT-a odabire bilo koju kreiranu sliku s lokalnog računala, upisuje naziv novog NFT-a te definira koja će biti njegova cijena. Ukoliko isti korisnik kreira kriptovalutu, upisuje naziv kriptovalute i definira koja će biti početna cijena jedne jedinice te valute.

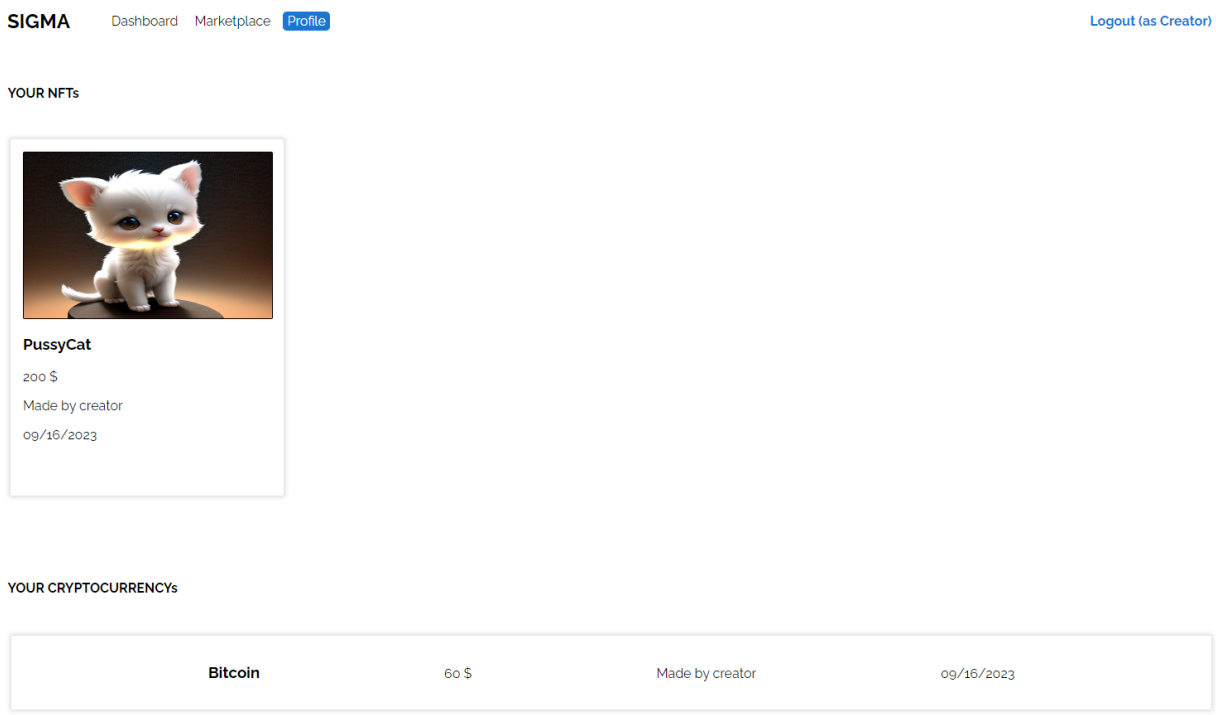


Slika 6.6. Prikaz forme za kreiranje novog NFT-a



Slika 6.7. Prikaz forme za kreiranje nove kriptovalute

Kreator nakon kreiranja može vidjeti sve svoje kreirane NFT-ove i kriptovalute na svom profilu na koji može otići klikom na gumb profil (eng. profile) na navigacijskoj traci.

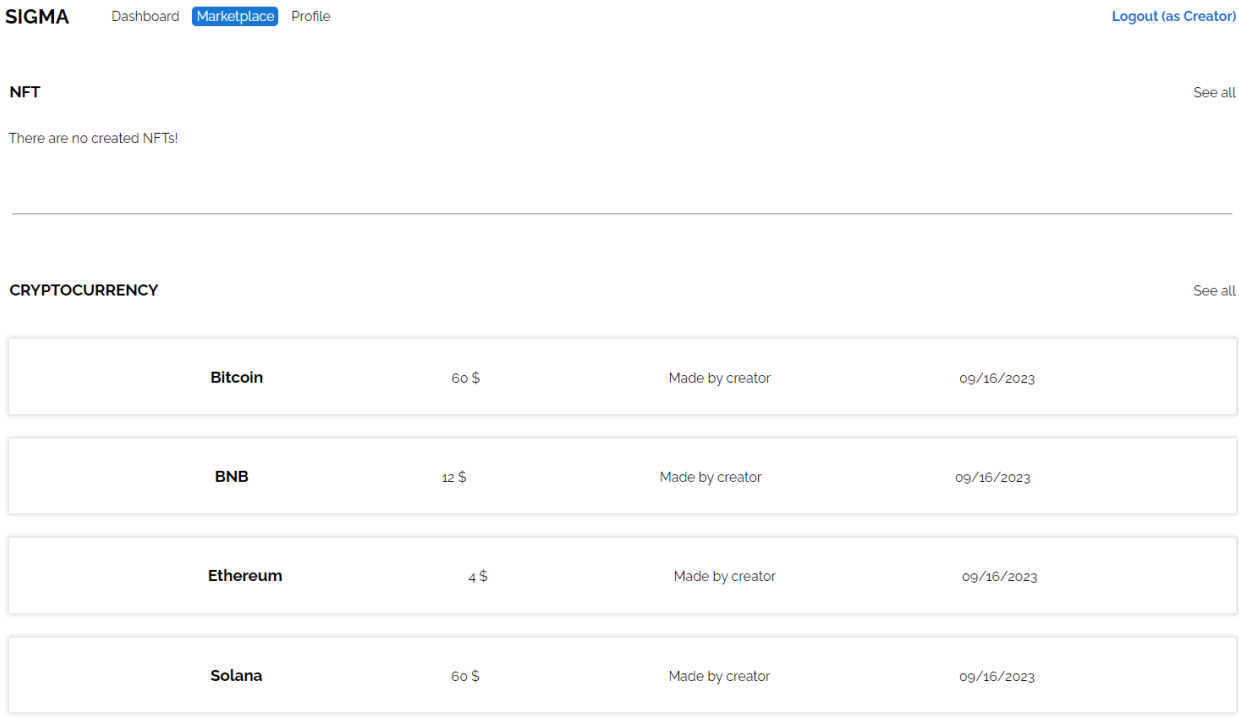


Slika 6.8. Prikaz profila kreatora koji je ulogiran

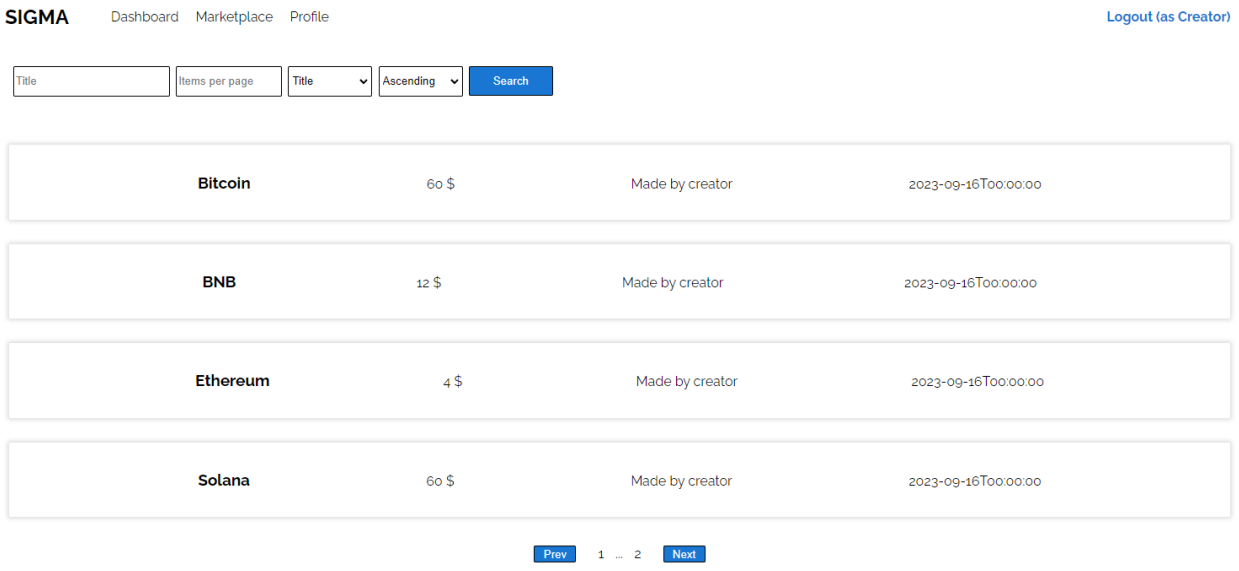
Isto tako, kreator može vidjeti sve kreirane kriptovalute i NFT-ove u cijelom sustavu klikom na gumb trgovina (eng. marketplace) na navigacijskoj traci.

6.3. Investiranje u NFT-ove i kriptovalute

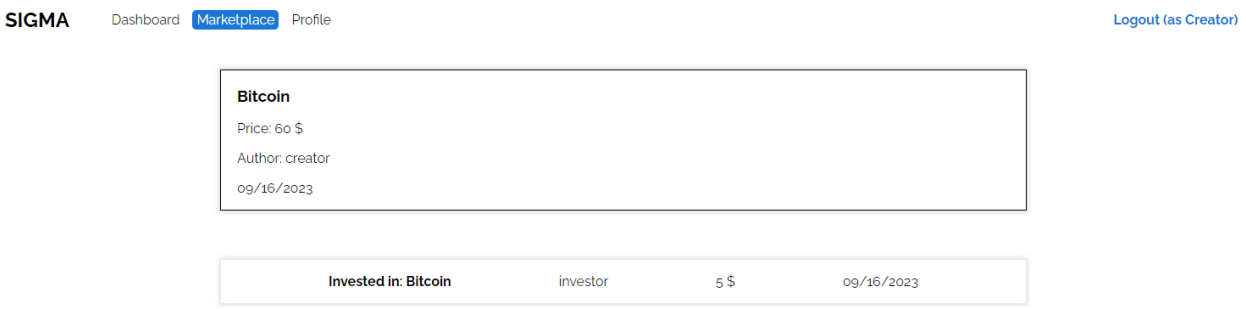
Odlaskom na trgovinu (eng. marketplace), investitor može pregledati određene kreirane NFT-ove i kriptovalute različitih kreatora te investirati u koju želi. Ukoliko želi pregledati sve kriptovalute ili sve NFT-ove koji nisu prikazani na trgovini, može kliknuti na gumb pogledaj sve (eng. See all) te dobiti mogućnost pretraživanja i sortiranja svih željenih elemenata. Kriptovalute i NFT-ovi se mogu pretraživati po nazivu te korisnik može odabrati koliko prikaza istog želi po jednoj stranici. Pri tome, on ima određeni iznos na profilu koji smije i može iskoristiti za investiranje. Klikom na određeni NFT ili na određenu kriptovalutu, korsnik, u ovom slučaju investitor, može pogledati da li je već investirao u određenu kriptovalutu ili NFT, dok kreator može pregledati da li je neki od investitora investirao u njegovu kriptovalutu ili NFT.



Slika 6.9. Prikaz trenutno kreiranih kriptovaluta i NFT-ova

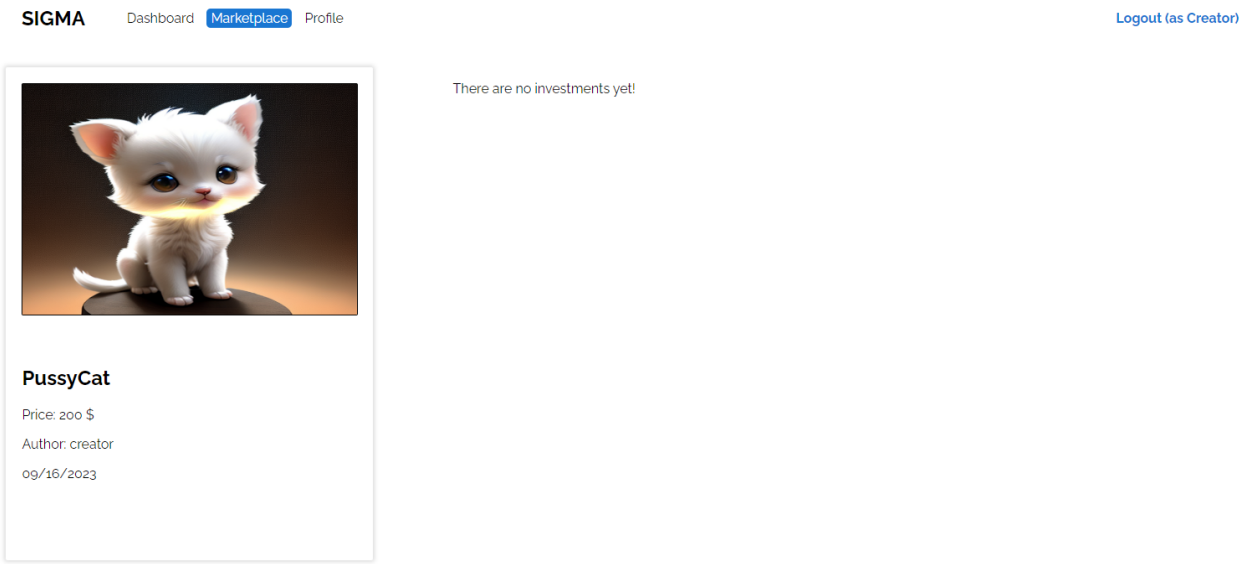


Slika 6.10. Prikaz svih kreiranih kriptovaluta (jednako za NFT-ove)



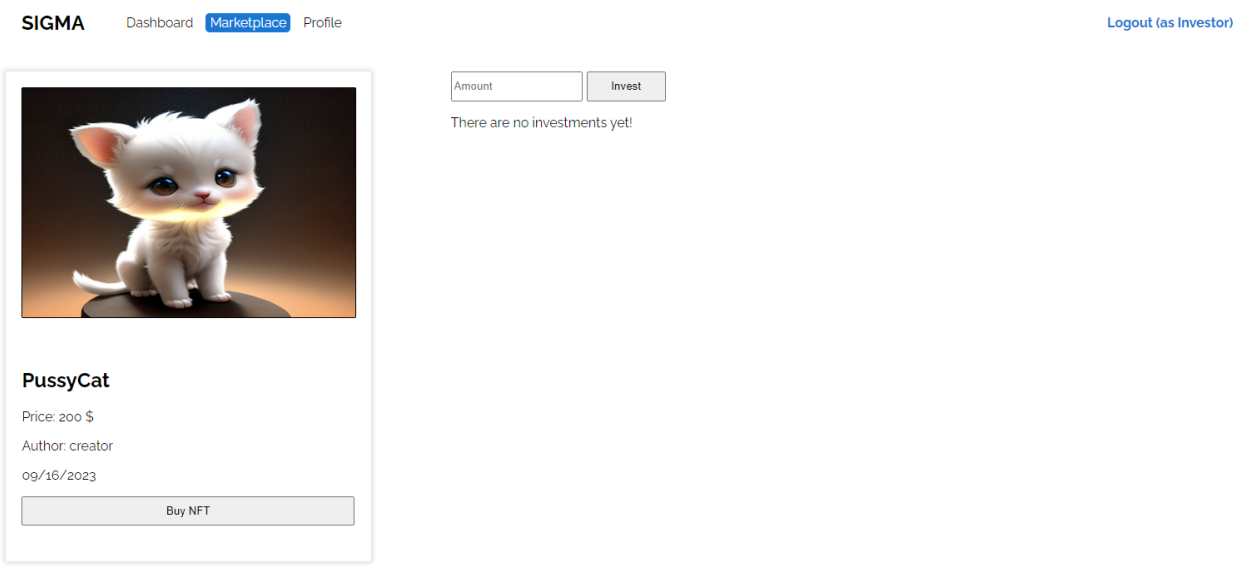
Slika 6.11. Prikaz svih investicija na jednoj kriptovaluti (jednako za NFT-ove)

Na slici 6.11. prikazane su sve investicije koje se odnose na **Bitcoin** kriptovalutu, koju je kreirao korisnik **creator**, a u nju je investirao korisnik **investor**. Jednako vrijedi i za NFT-ove.



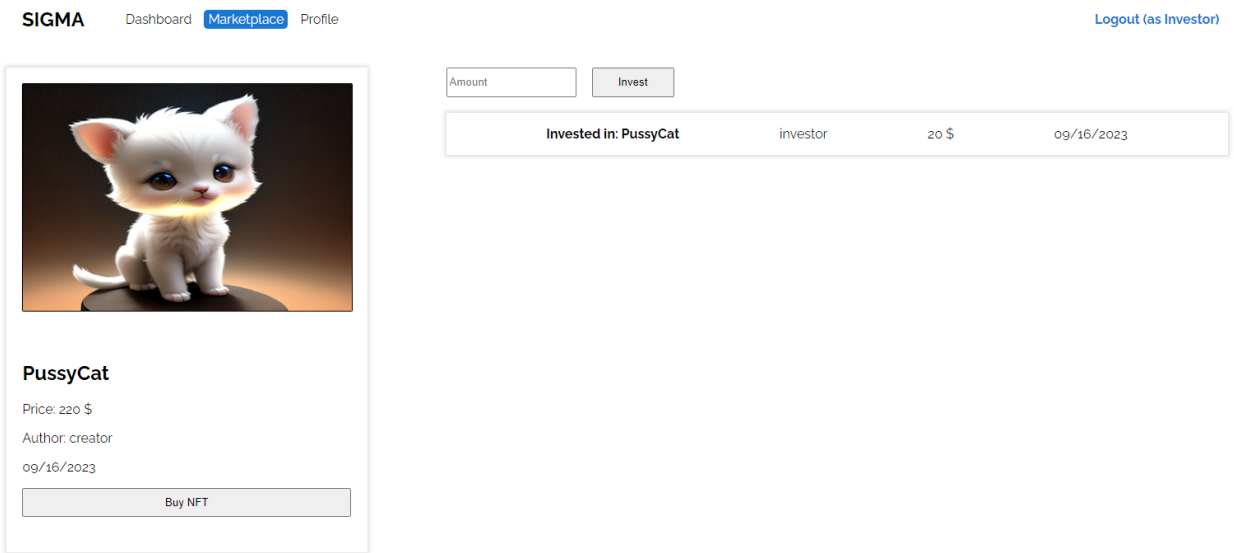
Slika 6.12. Prikaz svih investicija na jednom NFT-u jednog kreatora

Na slici 6.12. prikazan je zaslon **PussyCat** NFT-a (korisnik ulogiran kao kreator), te na njemu ne postoji nijedna investicija. Ukoliko investor želi investirati u ovaj određeni NFT, na istoj stranici prikazat će mu se obrazac koji može koristiti za investiranje.



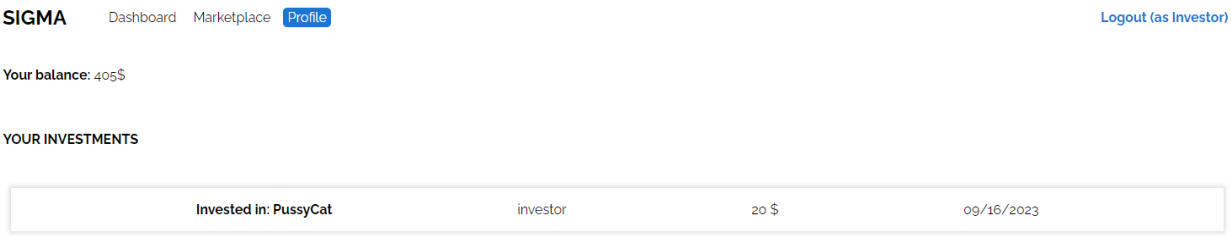
Slika 6.13. Prikaz detalja o NFT-u jednog investitora

Korisnik koji je ujedno i investitor upisuje određen iznos koji želi investirati u NFT te klikom na gumb investiraj (eng. Invest) to i odrađuje. Ukoliko je korisnik investirao 20 novčanih jedinica, cijena tog NFT-a će se povećati za investirani iznos, ta investicija će se prikazati na stranici koja je prikazana na slici 6.14., a investitorov iznos računa će se umanjiti za isti broj. Korisnici mogu pratiti svoje iznose računa na svom profilu, te mogu ulagati skroz dok imaju dostupan novac.



Slika 6.14. Prikaz jedne od investicija na jednom NFT-u

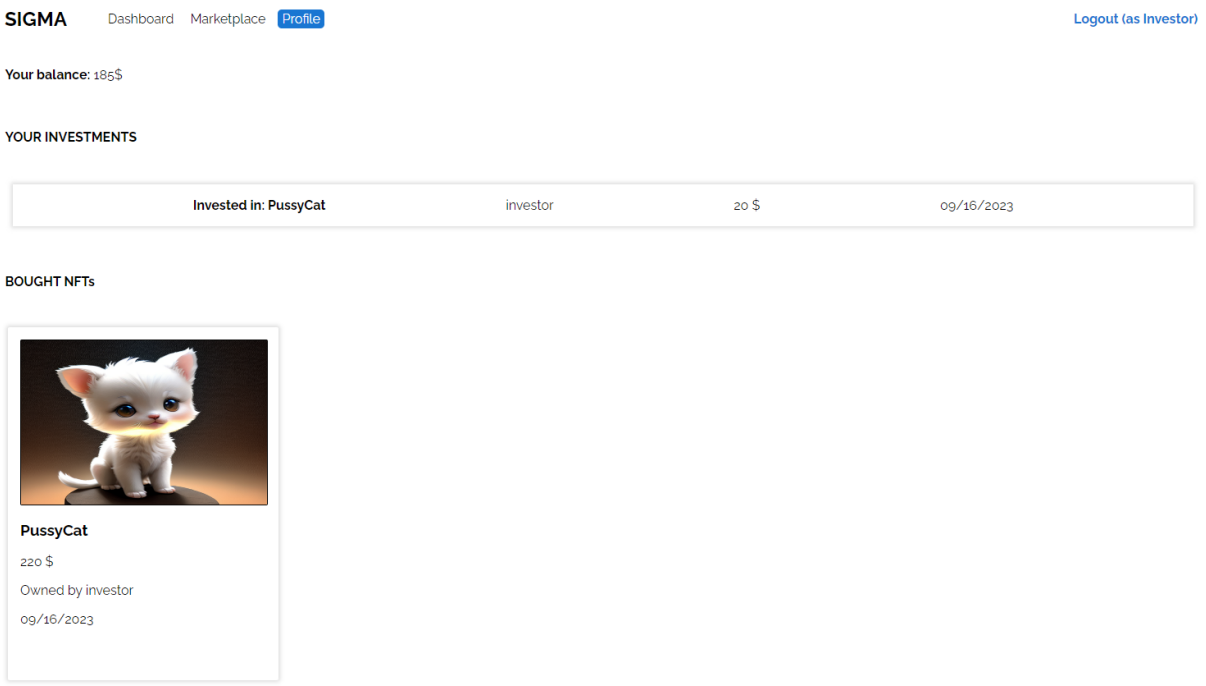
Ako je ulogirani korisnik ujedno i investitor, on može pregledati sve svoje dosadašnje investicije na svom korisničkom profilu.



Slika 6.15. Prikaz svih investicija na profilu investitora

6.4. Kupovanje NFT-ova i kriptovaluta

Kako bi korisnik mogao kupovati kriptovalute i NFT-ove, mora biti ulogiran kao investitor. Nakon svog logiranja u sustav, te odlaskom na trgovinu (eng. marketplace) korisnik može odabrati kriptovalutu ili NFT koji želi kupiti. Klikom na gumb kupi NFT (eng. Buy NFT) koji se nalazi na stranici sa detaljima o tom NFT-u on kupuje određeni NFT ukoliko ima dovoljnu svotu novca na računu, postaje vlasnik tog NFT-a te mu se iznos računa umanjuje za kupljeni iznos. Kupljeni NFT-ovi prikazani su na investorovom odnosno korisničkom profilu. Jednaka funkcionalnost vrijedi i za kriptovalute.



Slika 6.16. Prikaz profila investitora nakon kupovanja

7. ZAKLJUČAK

Izradom ovog diplomskog rada potvrđeni su tema i cilj rada a to je kreiranje i programiranje web aplikacije za upravljanje NFT-ovima te kriptovalutama. Tijekom rada bilo je potrebno paziti na pravilno izvršavanje svih dijelova aplikacije, na njihovo pravilno povezivanje te u konačnici na pravilno rukovanje i testiranje. Kroz rad, prikazani su brojni načini i tehnike izrade pojedinih dijelova funkcionalnosti i funkcionalnog sustava kao što su kreiranje lokalne baze podataka za njihovu pohranu, kreiranje i programiranje backend dijela web aplikacije kroz ASP.NET framework uz programski jezik C# za osiguranje pravilnog slanja zahtjeva te dobivanja odgovora na zahtjeve te kreiranje i kodiranje frontend dijela aplikacije koji je omogućio upravljanje konkretim elementima aplikacije koji su vidljivi krajnjem korisniku. Isto tako, osigurana je jednostavnost korištenja za svakog korisnika koji se odluči registrirati odnosno prijaviti u sustav te ga koristiti. Korisniku je omogućen izbor želi li biti kreator i kreirati NFT-ove te kriptovalute ili ima želju samo ulagati u ono što je već kreirano od strane drugih korisnika (kreatora). Ovakav sustav nije savršen, iz razloga što on predstavlja simulaciju prilikom korištenja novčanih sredstava te ne upravlja stvarnim novčanim iznosima korisnika. Za daljnje korištenje, postoje brojna poboljšanja i nove funkcionalnosti koje se mogu dodatno izraditi i kreirati unutar sustava. Za korištenje u stvarnom svijetu, neizbježna bi bila implementacija rada sa stvarnim novčanim sredstvima, dodatne opcije sigurnosti koje bi pomogle korisniku prilikom korištenja i razvojnom programeru prilikom razvoja te daljnjeg testiranja i održavanja kao što su provjere novčanih sredstava kroz sustav vlastite banke i slično. Ovaj rad predstavlja jedan od načina na koji je moguće organizirati načine upravljanja NFT-ovima i kriptovalutama koji se u današnjici sve više koriste i koji će se daljnje razvijati.

LITERATURA

[1] ASP.NET overview [online], Microsoft, dostupno na:

https://learn.microsoft.com/en-us/aspnet/overview, [8.9.2023.]

[2] A tour of the C# language [online], Microsoft, dostupno na:

https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/, [10.9.2023.]

[3] Introducing SQL Server 2022 [online], Microsoft, dostupno na:

https://www.microsoft.com/en-us/sql-server, [24.8.2023.]

[4] The library for web and native user interfaces [online], React, dostupno na:

https://react.dev/, [24.8.2023.]

[5] What is Postman? [online], Postman, dostupno na:

https://www.postman.com/product/what-is-postman/, [24.8.2023.]

[6] Create a secure ASP.NET Web Forms app with user registration, email confirmation and password reset (C#) [online], Microsoft, dostupno na:

https://learn.microsoft.com/en-us/aspnet/web-forms/overview/security/create-a-secure-aspnet-web-forms-app-with-user-registration-email-confirmation-and-password-reset, [30.8.2023.]

[7] Convert DateTime to Date in ES6 [online], Stackoverflow, dostupno na:

https://stackoverflow.com/questions/62420647/convert-datetime-to-date-in-es6, [15.9.2023.]

[8] React conditional rendering [online], W3Schools, dostupno na:

https://www.w3schools.com/react/react_conditional_rendering.asp, [30.7.2023.]

[9] 4 Ways to Render JSX Conditionally in React [online], Dev, dostupno na:

https://dev.to/rasaf_ibrahim/4-ways-to-render-jsx-conditionally-in-react-2bal, [30.7.2023.]

[10] Uploading Image To Server Using Web API 2.0 [online], C# Corner, dostupno na:

https://www.c-sharpcorner.com/article/uploading-image-to-server-using-web-api-2-0/, [7.9.2023.]

[11] Returning Images from ASP.NET Web API [online], Codeguru, dostupno na: https://www.codeguru.com/dotnet/returning-images-from-asp-net-web-api/, [7.9.2023.]

[12] Base64 to Image [online], Base64Guru, dostupno na: https://base64.guru/converter/decode/image, [7.9.2023.]

[13] Introduction to SQL [online], W3Schools, dostupno na: https://www.w3schools.com/sql/sql_intro.asp, [22.7.2023.]

[14] GUIDs In C# And .NET [online], C# Corner, dostupno na: https://www.c-sharpcorner.com/UploadFile/prasoonk/guids-in-C-Sharp-and-net/, [20.7.2023.]

[15] .NET dependency injection [online], Microsoft, dostupno na: https://learn.microsoft.com/en-us/dotnet/core/extensions/dependency-injection, [20.7.2023.]

[16] Create an ASP.NET Web API 2 Project [online], okta Developer, dostupno na: https://developer.okta.com/blog/2019/03/13/build-rest-api-with-aspnet-web-api, [20.7.2023.]

[17] Common web application architectures [online], Microsoft, dostupno na: https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/common-web-application-architectures, [11.7.2023.]

[18] Uphold is changing the way people access money [online], Uphold, dostupno na: https://uphold.com/resources/about, [1.7.2023.]

[19] Welcome to Binance [online], Binance, dostupno na: https://www.binance.com/en/about, [1.7.2023.]

SAŽETAK

Ovaj projekt odnosno diplomski rad predstavlja jedan od načina prikazivanja logike sustava koji upravlja modernim trgovanjem odnsono trgovanjem NFT-ovima i kriptovalutama. U sklopu toga, razvijena je web aplikacija na zadanu temu koja je kroz rad i objašnjena kroz različite načine kreiranja i programiranja sustava upotpunjena različitim slikama koje predstavljaju dio prikaza aplikacije i različitim vizualnim prikazima koji predstavljaju dodatna objašnjenja za ono što je potrebno. Uz ovaj rad, priložen je i izvorni kod aplikacije kako bi se mogao vidjeti način izrade i način kodiranja sustava.

Ključne riječi: kriptovaluta, NFT, organizacija, sustav, trgovanje

ABSTRACT

Web application for managing NFTs and cryptocurrencies

This project or graduation thesis, represents one way of presenting the logical system that is used in moderna trading using NFTs and cryptocurrencys. As a part of this, a web application was developed based on a given topic, which was explained through working process and through different ways of creating and programming of the application, completed with multiple images that represent certain part of the application, visual representations that include additional explanations for what is needed. In addition to this work, source code has been attached in order to see how it was created and how it was coded.

Key words: cryptocurrency, NFT, organization, system, trading

ŽIVOTOPIS

Karlo Adžić rođen je 20. listopada 1999. godine u Požegi. Upisuje Osnovnu školu Dragutina Lermana u Brestovcu koju pohađa do 6. razreda te glazbenu školu u Požegi. Nakon završene Osnovne škole Dobriša Cesarić u Požegi i glazbene škole, upisuje matematičku gimnaziju u istom gradu. Srednju školu završava s vrlo dobrim uspjehom te 2018. godine upisuje preddiplomski studij, smjer računarstvo, na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku u želji za napredovanjem i razvijanjem. Nakon završenog preddiplomskog studija, upisuje diplomski studij na istom fakultetu. Trenutno je student 2. godine diplomskog studija.

_____

Potpis autora