Aplikacija za nadziranje financijskog tržišta

Petričević, Marko

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

Sveučilišni prijediplomski studij Računarstvo

Smjer računalno inženjerstvo

Aplikacija za nadziranje financijskog tržišta

Završni rad

Marko Petričević

Osijek, 2024.

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. POSTOJEĆA RJEŠENJA 3

2.1. Yahoo Finance 3

2.2. Google Finance 4

2.3. TradingView 5

2.4. Robinhood 5

3. KORIŠTENE TEHNOLOGIJE 7

3.1. Visual Studio Code 7

3.2. C# 7

3.3. Flutter 7

3.4. MSSQL 10

3.5. POLYGON.IO 11

4. PROGRAMSKO RJEŠENJE APLIKACIJE 13

4.1. Registracija korisnika 13

4.2. Funkcija sažimanja 14

4.3. Prijava korisnika 15

4.4. Odjava korisnika 16

4.5. Odabir željenih financijskih izvještaja 17

4.6. Programsko rješenje dohvaćanja podataka 17

4.7. Programsko rješenje baze podataka 19

4.8. Programsko rješenje za prikaz podataka 20

5. TESTIRANJE I ANALIZA APLIKACIJE 22

5.1. Kako koristiti aplikaciju 22

5.2. Testiranje rada aplikacije 22

5.2.1. Testiranje prijave korisnika u aplikaciju 22

5.2.2. Testiranje odabira željenih financijskih izvještaja 24

5.2.3. Testiranje numeričkog prikaza odabranih financijskih stavki 25
5.2.4. Testiranje grafičkog prikaza odabranih financijskih stavki 26
5.3. Testiranje aplikacije na mobilnom uređaju 27
5.4. Analiza aplikacije 30
6. ZAKLJUČAK 32
LITERATURA 33
SAŽETAK 34
ABSTRACT 35

1. UVOD

Financije su neizostavan dio našeg svakodnevnog života, od upravljanja osobnim budžetom do donošenja investicijskih odluka. Prema [1] financijska tržišta igraju ključnu ulogu u alokaciji kapitala i poticanju gospodarskog rasta, utječući na sve od investicija do potrošnje. U današnjem dinamičnom financijskom okruženju, gdje se tržišta neprestano mijenjaju, a informacije se šire brzinom svjetlosti, praćenje svih zbivanja i donošenje informiranih odluka može biti izazovno

Upravo tu do izražaja dolaze aplikacije za praćenje financijskog tržišta. One pružaju korisnicima pristup ažurnim podacima, analizama i vijestima, omogućavajući im da prate svoje investicije, istražuju nove prilike i donose promišljene financijske odluke. Bilo da se radi o praćenju kretanja cijena dionica, kriptovaluta ili drugih financijskih instrumenata, ove aplikacije pružaju korisnicima sveobuhvatan uvid u financijska tržišta, omogućujući im pristup informacijama neovisno o lokaciji i vremenu.

Osim toga, mnoge aplikacije nude napredne alate za analizu, personalizirane preporuke i mogućnost postavljanja upozorenja o promjenama cijena, što korisnicima omogućava da budu uvijek korak ispred tržišta. Ove aplikacije su postale neprocjenjiv alat za sve, od iskusnih investitora koji traže dublje uvide do onih koji tek ulaze u svijet financija i žele steći osnovna znanja.

U suštini, aplikacije za praćenje financijskog tržišta olakšavaju navigaciju kroz kompleksnost financijskog svijeta, pružajući korisnicima informacije i alate potrebne za donošenje odluka koje će im pomoći da ostvare svoje financijske ciljeve. Bilo da se radi o štednji za mirovinu, ulaganju u dionice ili jednostavno praćenju stanja na tržištu, ove aplikacije su postale ključni saveznici u postizanju financijske sigurnosti i uspjeha.

Cilj ovog završnog rada je stvoriti jednostavnu, ali i dovoljno dobro razvijenu aplikaciju koja će pomoći svima u praćenju financijskih izvještaja. Aplikacija je prilagođena svima od početnika do iskusnih korisnika.

1.1. Zadatak završnog rada

U teorijskom dijelu završnog rada istražit ćemo i opisati probleme i izazove s kojima se korisnici suočavaju pri praćenju i analiziranju financijskog tržišta. Analizirat ćemo postojeće aplikacije i

platforme, identificirajući njihove prednosti i nedostatke kako bismo bolje razumjeli trenutno stanje na tržištu.

Na temelju analize problema i postojećih rješenja, definirat ćemo funkcionalne i nefunkcionalne zahtjeve naše aplikacije za nadzor financijskog tržišta. To će uključivati funkcionalnosti poput praćenja cijena u stvarnom vremenu, prikaza povijesnih podataka, alata za tehničku analizu te mogućnost personaliziranih preporuka. Nefunkcionalni zahtjevi će obuhvaćati performanse, sigurnost, skalabilnost i jednostavnost korištenja.

Zatim ćemo razviti model, arhitekturu i dizajn aplikacije, kako na strani korisnika tako i na strani poslužitelja. Korisničko sučelje biti će razvijeno koristeći Flutter tehnologiju kako bi se osiguralo intuitivno korisničko iskustvo. Programsko rješenje na strani korisnika biti će odgovorno za obradu podataka, integraciju s vanjskim izvorima financijskih podataka, implementaciju algoritama za analizu i generiranje personaliziranih preporuka.

Za razvoj aplikacije koristit ćemo prikladne programske jezike, tehnologije i razvojne okvire. Izbor tehnologija će biti temeljen na analizi zahtjeva i najboljim praksama u razvoju mobilne aplikacije.

Na kraju, provest ćemo temeljito testiranje aplikacije kako bismo osigurali da ispunjava sve definirane zahtjeve i da pruža korisnicima optimalno iskustvo. Testiranje će uključivati testiranje funkcionalnosti, performansi, sigurnosti i korisničkog iskustva na različitim uređajima i preglednicima.

2. POSTOJEĆA RJEŠENJA

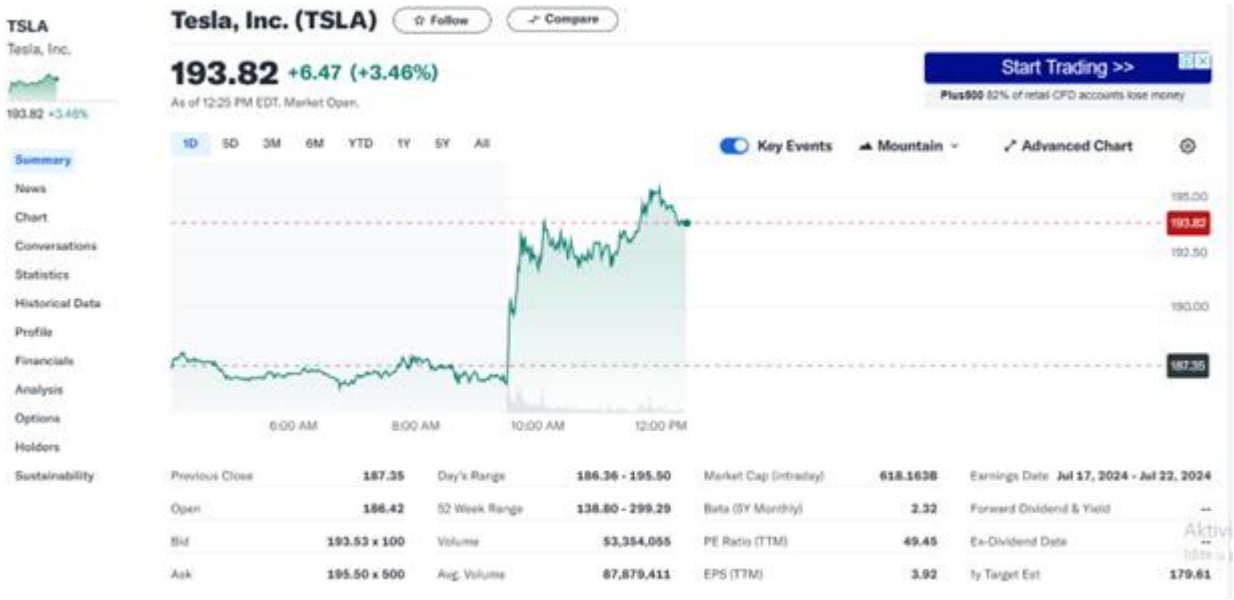
Kroz ovo poglavlje dobit će se uvid u trenutno stanje na tržištu aplikacija koje služe za praćenje financijskog tržišta. Prikazat će se neke od najpopularnijih web ili mobilnih aplikacija kao što su: Yahoo Finance, Google Finance i Robinhood te TradingView. Analizirat će se njihove ključne značajke, prednosti i nedostatci, nastavno ući u srž aplikacija te vidjeti kakvo je korisničko iskustvo sa spomenutim aplikacijama za nadzor financijskog tržišta.

2.1. Yahoo Finance

Yahoo Finance [2] je jedna od najpoznatijih aplikacija za nadzor financijskog tržišta koja korisnicima pruža prikaz kretanja određenih dionica, kriptovaluta i sličnog u stvarnom vremenu. Velika prednost Yahoo Finance je ta što je besplatan i dostupan svima, kako preko mobilne aplikacije tako i preko web aplikacije.

Aplikacija omogućuje korisnicima kreiranje njihovog vlastitog portfelja pomoću kojega mogu pratiti svoje financijsko stanje na jednom mjestu. Preko aplikacije moguće je pratiti određene kompanije kao i njihovo kretanje na tržištu, čime je olakšana odluka u koju kompaniju uložiti. Aplikacija nam omogućuje mnogo alata, a jedan od najpotrebnijih je mogućnost postavljanja određenih upozorenja prilikom promjene cijena, nekih bitnih događaja unutar kompanije itd., Na slici 2.1. prikazan je graf kretanja cijene kompanije Tesla, Inc unutar Yahoo Finance aplikacije.

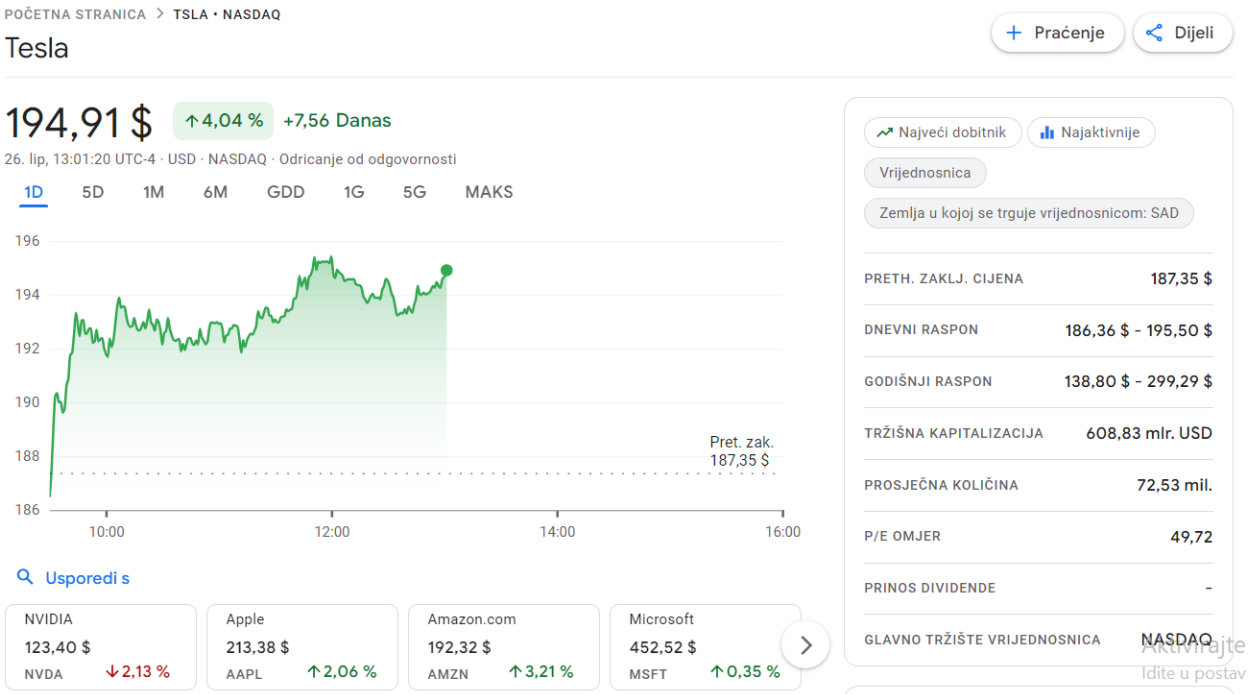
Također, na Yahoo Finance web stranici postoji vjesnik putem kojeg možemo pratiti najnovija događanja u svijetu financija kao što su: promjene na burzi, promjene cijena dionica određenog sektora kao i prošlost kretanja cijena dionica, kriptovaluta i sl..



Slika 2.1. Kretanje cijena dionice Tesla, Inc. u aplikacija Yahoo Finance [2]

2.2. Google Finance

Google Finance [3] je besplatna online aplikacija koja je vrlo slična Yahoo Finance. Yahoo Finance je kvalitetnija u smislu da korisniku pruža bolje i složenije informacije, ali samim time aplikacija je složenija za korištenje od Google Finance. Google Finance je aplikacija okrenuta prema korisniku u smislu jednostavnosti i preglednosti. Također, namijenjena je za početnike u području praćenja financija. Na slici 2.2. prikazano je kretanje cijena Tesla, Inc. u Google finance web aplikaciji.



Slika 2.2. Kretanje cijena dionice Tesla, Inc. u aplikacija Google Finance [3]

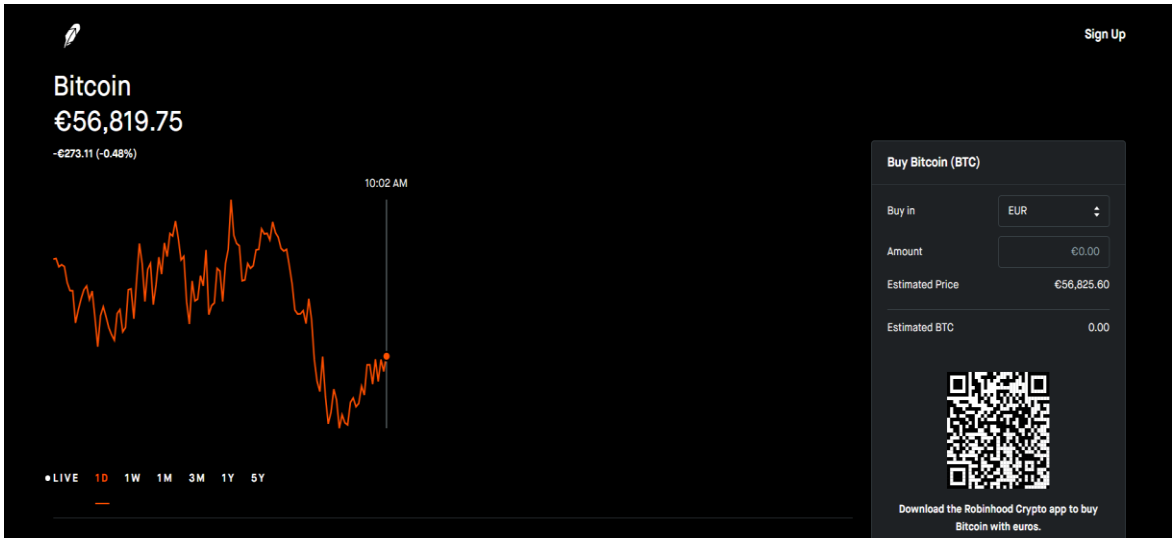
2.3. TradingView

TradingView [4] je popularna *web-based* platforma razvijena od istoimene kompanije. Platforma je postala poznata zbog svojih grafikona koji korisnicima pružaju detaljniju analizu krivulje promjene cijena i lakšu vidljivost trenutka kada je najbolje uložiti. Aplikacija je napravljena da bude pregledna i razumljiva svima, od početnika, do zahtjevnih korisnika. Na slici 2.3. možemo uočiti kako je grafikon promjene cijena dionica u kompaniji Tesla, Inc., vrlo detaljan. Nažalost, za razliku od gore navedenih aplikacija, TradingView nije u potpunosti besplatan pa ukoliko se želi postići potpuna funkcionalnost platforme mora se platiti profesionalna verzija.

Slika 2.3. Grafikon kretanja cijena dionica kompanije Tesla, Inc. pomoću platforme TrandingView [4]

2.4. Robinhood

Robinhood [5] je aplikacija nastala 2013. godine sa idejom da učini ulaganje novca jednostavnijim i pristupačnijim. Samo ime aplikacije govori nam kako je aplikacija pristupačna svima, odnosno aplikacija je besplatna. Prednosti aplikacije su mogućnost korištenja putem web platforme kao i putem mobilne aplikacije. Ono što izdvaja Robinhood od ostalih aplikacija je to što omogućava ljudima manje financijske moći kupnju djelomičnih dionica. Aplikacija, uz mogućnost ulaganja u dionice, isto tako pruža mogućnost ulaganja u kriptovalute. Robinhood nema širok spektar ponude kompanija i kriptovaluta, niti ima veliku ponudu alata, stoga su to dva nedostatka, odnosno negativnosti zbog kojih aplikacija nije popularnija u svijetu financija. Slika 2.4. nam prikazuje grafički prikaz kretanja kriptovaluta unutar aplikacije RobinHood.



Slika 2.4. Kretanje cijena kriptovalute Bitcoin promatrane u aplikaciji Robinhood [5]

3. KORIŠTENE TEHNOLOGIJE

Pri izradi aplikacije za nadziranje financijskog tržišta potrebni su nam određeni programski jezici koje ćemo objasniti u ovom poglavlju, navesti njihove prednosti i mane te primjenu. Svi programski jezici pisani su u editoru Visual Studio Code. Zatim, tehnologije koje ćemo objasniti u ovom poglavlju su:

1. Visual Studio Code (VS Code)
2. C#
3. Flutter
4. MSSQL
5. Polygon.io

3.1. Visual Studio Code

Visual Studio Code [6] je besplatni i *open-source* editor koda stvoren od strane kompanije Microsoft 2015. godine. VS Code je jako moćan editor koji pruža mogućnost programiranja u brojnim jezicima te zbog toga je stekao veliku popularnost. Vrlo je jednostavan za korištenje te je orijentiran prema početnicima kao i prema iskusnim programerima. Vrlo je prilagodljiv korisnicima kada je u pitanju dizajn, a i programski jezici. U pitanju dizajna VS Code ima mnoštvo načina kako urediti dizajn od boje koda, boje pozadine pa sve do najmanjih sitnica koje će programeru olakšati pisanje programa. S druge strane, VS Code nudi mnoštvo proširenja koja omogućuju korisniku pisanje koda u različitim programskim jezicima, a opet ne prisiljava korisnika da instalira proširenja koja mu nisu potrebna i time štedi memoriju računala. Unutar samog uređivača postoji alat IntelliSense. IntelliSense je moćan alat koji programeru tijekom pisanja koda unaprijed nudi moguća programska rješenja na temelju njegovih prijašnjih linija koda. Uz IntelliSense, VS Code programeru pruža mogućnost direktnog spajanja na Git što olakšava spremanje koda koji je napisan u VS Code-u. Uređivač ima integrirani terminal koji omogućuje pokretanje naredbi i skripti izravno iz VS Code-a.

3.2. C#

C# [7] je programski jezik opće namjene, razvijen 2000. godine od strane kompanije Microsoft. Nalazi se unutar .NET sustava. .NET je Microsoftova platforma za razvoj softvera koja omogućava izradu različitih vrsta aplikacija, uključujući desktop, web, mobilne i cloud

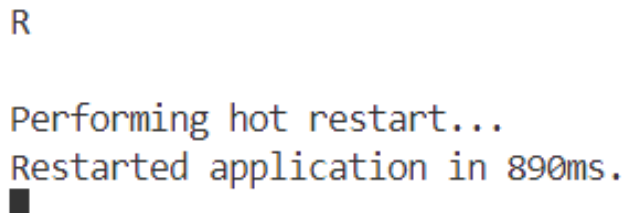
aplikacije. C# je objektno orijentirani programski jezik te se unutar njega sve tretira kao objekt, što znatno olakšava pisanje kao i povezivanje različitih dijelova programa. Objektno orijentirano programiranje omogućuje programerima da kreiraju modularni, lako održiv i proširiv kod. Koncepti poput klasa, objekata, nasljeđivanja, polimorfizam i enkapsulacije su temeljni dio C# svojstava. C# ima široku primjenu te pomoću njega je moguće izraditi različite vrste aplikacija kao što su: desktop, web i mobilne aplikacije. Lako integrira s drugim Microsoft tehnologijama poput SQL Servera, Azure oblaka i Visual Studia, što olakšava razvoj cjelovitih rješenja.

Uz to C# je *cross-platforma* odnosno nudi mogućnost da razvijene aplikacije rade na različitim platformama. Za razliku od mnogih programskih jezika, C# nudi mogućnost upravljanja memorijom. C# koristi *garbage collector* za upravljanje memorijom koji to radi automatski, ali nedeterministički. C# dolazi s opsežnom standardnom bibliotekom koja pruža gotove klase i funkcije za rad s datotekama, mrežom, bazama podataka, grafikom i mnogim drugim.

3.3. Flutter

Flutter [8] je Googleova *cross-platforma* za korisničko sučelje koji služi pri izradi aplikacija za mobilne uređaje, web i desktop. Flutter radi s postojećim kodom te ga koriste programeri i organizacije diljem svijeta, besplatan je i otvorenog koda. Programski kod piše se u Dart programskom jeziku. Dart je objektno orijentirani programski jezik koji je nalik C# i C++ programskim jezicima. Flutter nudi bogat set već definiranih *widgeta* koji omogućuju jednostavno kreiranje atraktivnih i prilagodljivih korisničkih sučelja. *Widgeti* su dizajnirani na način da se ponašaju i izgledaju kao izvorni elementi na svakoj platformi. Kao i C#, Flutter je *cross-platforma* koja omogućuje da se prikaz sučelja, napravljen u Flutter-u, prilagođava različitim veličinama ekrana uređaja. Posebno je privlačna činjenica to da se jedan kod prevodi u izvornu (engl. *native*) mobilnu aplikaciju za iOS i za Android operativne sustave, što skraćuje vrijeme razvoja mobilnih aplikacija. Smanjuje se trošak izrade aplikacije jer programeri ne moraju pisati dva različita programska rješenja za Android i iOS aplikacije, a samim time i kasnije održavanje same aplikacije. Unutar samog alata dostupna je *hot reload* funkcija. *Hot reload* funkcija je zapravo mogućnost mijenjanja koda tijekom izvođenja aplikacije, odnosno mijenjanje programskog rješenja aplikacije u stvarnom vremenu. *Hot reload* funkcija se unutar VS Code-a postiže slanjem slova R unutar VS Code terminala. Slika 3.1. prikazuje kako pokrenuti *hot reload* funkciju unutar

VS Code terminala te kako je ta funkcija mnogo brža od novog pokretanja aplikacije. Često vrijeme ponovnog pokretanja aplikacije putem navedene funkcije iznosi manje od jedne sekunde, dok je za pokretanje aplikacije putem *flutter run-a* potrebno preko 10 sekundi. Flutter je postao popularan jer je pojednostavio izradu jednostavnog, a privlačnog korisničkog sučelja. Za vizualizaciju podatka unutar Flutter-a imamo mnogo biblioteka. Jedna od najpopularnijih biblioteka za vizualizaciju podataka je Fl_chart. Fl_chart je biblioteka za prikaz podataka pomoću grafikona. Biblioteka omogućava mnoge vrste grafikona kao što su: linijski, stupčasti, kružni i sl. Prilikom izrade aplikacije za nadziranje financijskog tržišta korištena je Fl_chart biblioteka koja se pokazala kao i više nego dovoljno rješenje za prikaz kretanja financijskog tržišta kroz linijski grafikon.



Slika 3.1. Korištenje *hot reload* funkcije unutar terminala te brzina ponovnog pokretanja aplikacije

Flutter je relativno nova tehnologija te zbog toga nema preveliku podršku i znanja kao ostale mnogo razvijenije i popularnije tehnologije. Iako Flutter zajednica brzo raste, još uvijek je manja od zajednica drugih popularnih okvira. To znači da može biti teže pronaći pomoć i odgovore na specifična pitanja ili probleme. Ipak, važno je naglasiti kako rastom popularnosti raste i Flutter zajednica, a samim time i programska podrška je stabilnija. S obzirom na prednosti koje Flutter nudi, poput brzog razvoja, izvrsnih performansi i mogućnosti izrade aplikacija za više platformi iz jedne kodne baze, mnogi vjeruju da će on u budućnosti postati još popularniji i raširenij. Za upotrebu Flutter-a unutar VSCode-a potreban je Flutter SDK sa službene Flutter stranice. Nakon instalacije potrebno je instalirati Dart i Flutter ekstenzije unutar VSCode-a, a aplikacija se pokreće sa naredbom *flutter run*. Nakon navedene naredbe potrebno je odabrati način pokretanja aplikacije, odnosno hoće li se pokrenuti kao mobilna, web ili stolna aplikacija. Ukoliko želimo mobilnu aplikaciju potreban je emulator unutar Android studio-a ili putem fizičkog uređaja. Na slici 3.2.

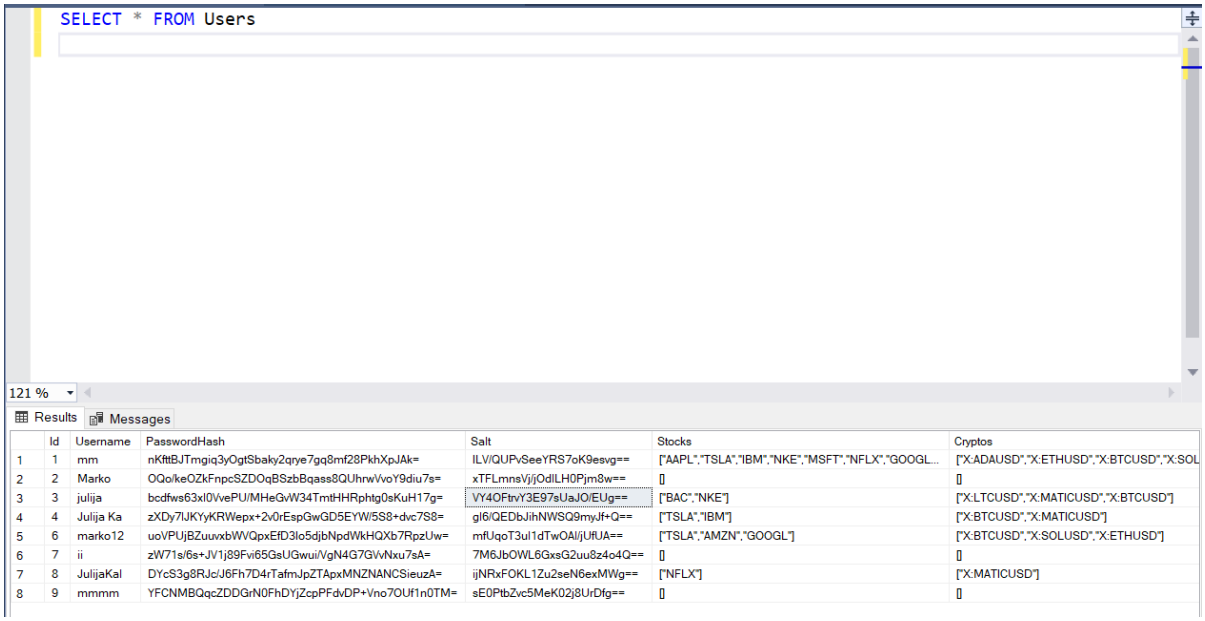
prikazano je pokretanje Flutter aplikacije unutar terminala VSCode-a te dostupni uređaji pomoću kojih je moguće pokrenuti aplikaciju.

```
PS C:\Users\Marko\financial_app> flutter run
Connected devices:
Windows (desktop) • windows • windows-x64    • Microsoft Windows [Version 10.0.19045.4780]
Chrome (web)      • chrome  • web-javascript • Google Chrome 128.0.6613.86
Edge (web)        • edge    • web-javascript • Microsoft Edge 128.0.2739.42
[1]: Windows (windows)
[2]: Chrome (chrome)
[3]: Edge (edge)
```

Slika 3.2. Prikaz dostupnih uređaja za pokretanje Flutter aplikacije

3.4. MSSQL

Microsoft SQL [9] server je tehnologija za upravljanje relacijskim bazama podataka. MSSQL je jedan od najpopularnijih sustava za upravljanje bazama podataka te ima vrlo široku primjenu, od velikih korporacijskih sustava do manjih aplikacija. MSSQL koristi T-SQL kao svoj primarni upitni jezik. T-SQL je proširenje standardnog SQL-a i pruža dodatne mogućnosti za rad s podacima. MSSQL se može koristiti kao pozadinska baza podataka za web aplikacije pružajući pritom pohranu podataka i logiku aplikacije. Sustav je vrlo pouzdan u radu sa velikom količinom podataka te se zbog toga koristi u aplikacijama u kojima pogreške u bazama podataka moraju biti minimalne kao što je to slučaj sa aplikacijama za nadziranje financijskog tržišta. On također omogućuje različite vrste zaštite kao i veliku korisničku podršku od strane Microsofta. Upravljanje bazom postiže se korištenjem SQL Server Management Studio (SSMS) tehnologije. Prema [10] SSMS je moćan alat koji se koristi za upravljanje i administraciju Microsoft SQL Server baza podataka. To je integrirano okruženje koje pruža sve potrebne alate za konfiguriranje, nadzor, razvoj i održavanje vaših SQL Server instanci i baza podataka. Olakšava kreiranje novih tablica te pruža okruženje za pisanje i izvršavanje SQL upita. Na slici 3.3. prikazano je kako postaviti upit za dobivanje pohranjenih podataka unutar tablice korisnika te se na dnu nalazi prikaz kako su podatci pohranjeni u tablicu korisnika.



	Id	Username	PasswordHash	Salt	Stocks	Cryptos
1	1	mm	nKfttBJTmgiq3yOgtSbaky2qrye7gq8mf28PkhXpJAk=	ILV/QUPvSeeYRS7oK9esvg==	["AAPL","TSLA","IBM","NKE","MSFT","NFLX","GOOGL...	["X:ADAUSD","X:ETHUSD","X:BTCUSD","X:SOL
2	2	Marko	OQo/keOZkFnpcSZDOqBSzbBqass8QUhrwVvoY9diu7s=	xTFLmnsVjjjOdILH0Pjm8w==	[]	[]
3	3	julija	bcdfws63xl0VvePU/MHeGvW34TmtHHRphtg0sKuH17g=	VY4OFtnY3E97sUaJO/EUg==	["BAC","NKE"]	["X:LTCUSD","X:MATICUSD","X:BTCUSD"]
4	4	Julija Ka	zXDy7IJKYyKRWepx+2v0rEspGwGD5EYW/5S8+dvc7S8=	gl6/QEDbJihNWSQ9myJf+Q==	["TSLA","IBM"]	["X:BTCUSD","X:MATICUSD"]
5	6	marko12	uoVPUjBZuuvxbWVQpxEfD3lo5djbNpdWkHQXb7RpzUw=	mfUqoT3ul1dTwOAl/jUfUA==	["TSLA","AMZN","GOOGL"]	["X:BTCUSD","X:SOLUSD","X:ETHUSD"]
6	7	ii	zW71s/6s+JV1j89Fvi65GsUGwui/VgN4G7GVvNxu7sA=	7M6JbOWL6GxsG2uu8z4o4Q==	[]	[]
7	8	JulijaKal	DYcS3g8RJcIJ6Fh7D4rTafmJpZTApxMNZNANCSieuzA=	ijNRxFOKL1Zu2seN6exMWg==	["NFLX"]	["X:MATICUSD"]
8	9	mmmm	YFCNMBQqcZDDGrN0FhDYjZcpPFdvDP+Vno7OUf1n0TM=	sE0PtbZvc5MeK02j8UrDfg==	[]	[]

Slika 3.3. Postavljanje upita unutar SSMS-a

3.5. POLYGON.IO

Polygon.io [11] je platforma koja pruža širok spektar financijskih podataka i alata za programere, investitore i financijske institucije. Polygon.io se ističe svojom opsežnom kolekcijom podataka o tržištu, uključujući podatke o dionicama, kriptovalutama, valutama i još mnogo toga. Njihovi podaci su visoke kvalitete, ažurni i pokrivaju različite vremenske okvire, od povijesnih podataka do podataka u stvarnom vremenu. Ova raznolikost omogućuje korisnicima da analiziraju tržišne trendove, prate performanse svojih investicija i razvijaju sofisticirane algoritme za trgovanje.

Jedna od najvećih prednosti Polygon.io-a je njegova jednostavnost korištenja. Njihov API (*Application Programming Interface*) dobro je dokumentiran i nudi intuitivne metode za dohvaćanje podataka. Bez obzira jeste li iskusan programer ili tek počinjete, lako se možete integrirati s njihovom platformom i početi koristiti njihove podatke u svojim projektima. Velika prednost Polygon.io-a je to što unutar besplatne verzije pruža mogućnosti koje će zadovoljiti većinu korisnika.

Pristup financijskim podacima na Polygon.io-u ostvaruje se putem specifičnih web linkova dostupnih u njihovoj dokumentaciji. Ovi linkovi omogućuju dohvat različitih vrsta i grupa financijskih podataka, a njihov točan format ovisi o konkretnim potrebama korisnika. Svi linkovi sadrže API ključ, jedinstveni identifikator koji se dodjeljuje svakom korisniku nakon uspješne prijave na platformu. U izradi ove aplikacije korišten su linkovi za dohvat kretanja cijena za

kriptovalute i dionice kroz povijest. Primjer linka za dohvat dionica kompanije Apple prikazan je na slici 3.4..

```
https://api.polygon.io/v2/aggs/ticker/AAPL/range/1/day/2023-01-09/2023-02-10?
adjusted=true&sort=asc&apiKey=eQfUnvjUhH2rtGelyAyrXPUg4M6DiRdb
```

Slika 3.4. Dohvat podataka pomoću platforme Polygon.io

4. PROGRAMSKO RJEŠENJE APLIKACIJE

U ovom poglavlju proći će se kroz programsko rješenje aplikacije za nadziranje financijskog tržišta, detaljno objašnjavajući ključne komponente, tehnologije i arhitekturu koje čine ovu aplikaciju funkcionalnom i intuitivnom. Od prikaza podataka u stvarnom vremenu do interaktivnih grafikona i mogućnosti personalizacije, istražit ćemo kako smo iskoristili Flutter i druge tehnologije kako bismo stvorili alat koji će vam pomoći da donosite informirane investicijske odluke.

4.1. Registracija korisnika

Zbog sigurnosti kao i personalizacije prikaza željenih financijskih izvještaja uvedeni su prijava i registracija korisnika. Sa programske strane korištene su dvije tehnologije: C# kao programsko rješenje na strani poslužitelja te Flutter kao programsko rješenje na strani korisnika aplikacije. Prilikom prvog korištenja aplikacije korisnik se mora registrirati prije korištenja same aplikacije. Slika 4.1. prikazuje programsko rješenje registracije na korisnikovoj strani dok slika 4.2 prikazuje programsko rješenje registracije na strani poslužitelja. Zahtjev prvo zaprimi klijentska strana u Flutter-u koji zatim šalje zahtjev serverskoj strani u C# te se pokreće postupak registracije. Zahtjev na stranu poslužitelja šalje se putem http zahtjeva, slika 4.3. prikazuje kako je taj zahtjev obrađen. Korisnik unosi svoje željeno korisničko ime i lozinku u polja za unos na stranici za registraciju. Aplikacija provjerava jesu li polja popunjena i omogućuje gumb za registraciju samo ako su oba polja popunjena. Prilikom registracije provjerava se postoje li uneseni podatci u bazi podataka. Ukoliko ne postoje, registracija će biti uspješna.

```
  final success = await _apiService.register(username, password);
  if (success) {
    setState(() {
      _isLoginMode = true;
    });
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(content: Text('Registration successful! You can now log in.')),
    );
  } else {
    _showErrorDialog('Registration failed', 'Username already exists or another error occurred.');
  }
}
```

Slika 4.1. Programsko rješenje registracije na korisnikovoj strani u Flutter-u

```
public async Task<bool> CreateUser(string username, string password)
{
    if (await _dbContext.Users.AnyAsync(u => u.Username == username))
        return false; // Korisničko ime već postoji
    var salt = GenerateSalt();
    var passwordHash = HashPassword(password, salt);
    var user = new User
    {
        Username = username,
        PasswordHash = passwordHash,
        Salt = salt
    };
    _dbContext.Users.Add(user);
    await _dbContext.SaveChangesAsync();
    return true;
}
```

Slika 4.2. Programsko rješenje registracije na strani poslužitelja

```
[HttpPost("register")]
0 references
public async Task<IActionResult> Register([FromBody] RegisterRequest request)
{
    var success = await _userService.CreateUser(request.Username, request.Password);
    if (!success)
        return BadRequest("Username already exists");

    return Ok("User registered successfully");
}
```

Slika 4.3. Obrada http zahtjeva na strani poslužitelja

4.2. Funkcija sažimanja

Kako bi sigurnost bila na razini prije spremanja lozinke u bazu podataka, lozinka se šifrira pomoću funkcije sažimanja (engl. *hash*) za šifriranje podataka. Na slici 4.4. prikazano je programsko rješenje za funkciju sažimanja. Funkcija sažimanja funkcionira tako da uzme lozinku te joj dodaje tzv. sol za šifriranje podataka. Sol se nasumično generira kako bi se sigurnost povećala. Nadalje, lozinka sa dodanom solju prolazi kroz ugrađenu metodu *ComputeHash()*. Nakon provedene funkcije sažimanja šifrirana lozinka sprema se u bazu podataka.

```
2 references
private string HashPassword(string password, string salt)
{
    using (var sha256 = SHA256.Create())
    {
        var saltedPassword = password + salt;
        var saltedPasswordBytes = Encoding.UTF8.GetBytes(saltedPassword);
        var hashBytes = sha256.ComputeHash(saltedPasswordBytes);
        return Convert.ToBase64String(hashBytes);
    }
}
```

Slika 4.4. Funkcija sažimanja na strani poslužitelja

4.3. Prijava korisnika

Ukoliko je korisnik uspješno izvršio registraciju mora se prijaviti za daljnji pristup aplikaciji. Korisnik unosi korisničko ime i lozinku koju je koristio tijekom registracije. Ako podatci odgovaraju podatcima u bazi podataka, prijava je uspješna u suprotnome pojaviti će se poruka o pogrešci prilikom prijave u aplikaciju. Na slici 4.5. prikazano je programsko rješenje prijave korisnika na strani poslužitelja, dok slika 4.6. prikazuje programsko rješenje prijave na strani korisnika. Nakon uspješne prijave korisnik je proslijeđen na početni zaslon.

```
    [HttpPost("login")]
0 references
public async Task<IActionResult> Login([FromBody] LoginRequest request)
{
    var user = await _userService.GetUserByUsername(request.Username);

    if (user == null || !_userService.ValidatePassword(request.Password, user.PasswordHash, user.Salt))
        return Unauthorized("Invalid username or password");

    // Dohvat preferencija
    var userStocks = await _userService.GetUserPreferences(request.Username, "stocks");
    var userCryptos = await _userService.GetUserPreferences(request.Username, "cryptos");

    // Ako je uspješno, vrati preferencije zajedno s porukom
    return Ok(new { message = "Login successful", stocks = userStocks, cryptos = userCryptos });
}
```

Slika 4.5. Prijava u aplikaciju na strani poslužitelja

```
try {
  if (_isLoginMode) {
    // Prijava
    final response = await _apiService.login(username, password);

    if (response) {
      final loginResponse = await _apiService.sendLoginRequest(username, password);
      final responseData = json.decode(loginResponse.body);

      if (loginResponse.statusCode == 200) {
        final stocks = List<String>.from(responseData['stocks'] ?? []);
        final cryptos = List<String>.from(responseData['cryptos'] ?? []);

        Navigator.pushReplacementNamed(
          context,
          '/home',
          arguments: {
            'stocks': stocks,
            'cryptos': cryptos,
            'username': username,
          },
        );
      } else {
        // Prikaz poruke o grešci iz odgovora servera
        _showErrorDialog('Login failed', responseData['message'] ?? 'An error occurred.');
      }
    } else {
      // Prikaz poruke o pogrešnoj prijavi
      _showErrorDialog('Login failed', 'Invalid username or password.');
    }
```

Slika 4.6. Prijava u aplikaciju na korisnikovoj strani

4.4. Odjava korisnika

Korisnik nakon prijave na početnom zaslonu ima mogućnost odjave ukoliko ne želi koristiti aplikaciju. Odjava je napravljena tako da pritiskom tipke za odjavu korisnik se vraća na zaslon za prijavu gdje može napraviti novi račun ili se ponovo prijaviti. Jednostavno programsko rješenje odjave prikazano je na slici 4.7. .

```
Future<void> _logout() async {
  setState(() {
    username = null;
    selectedCryptocurrencies = [];
    selectedStocks = [];
  });

  Navigator.of(context).pushNamedAndRemoveUntil('/login', (Route<dynamic> route) => false);
}
```

Slika 4.7. Programsko rješenje odjave korisnika

4.5. Odabir željenih financijskih izvještaja

Nakon uspješne prijave korisnik će biti proslijeđen na početni zaslon gdje će imati opciju odabira željenih kompanija ili kriptovaluta koje želi pratiti. Korisnik nakon odabira ima mogućnost pomoću tipke za spremanje da mu se željene kompanije ili kriptovalute spreme uz njegovo korisničko ime u bazu podataka. Funkcija spremanja prikazana je na slici 4.8.. Funkcija za spremanje omogućuje korisniku da ne treba svaki puta ponovo označiti podatke koji ga zanimaju. Korisnik će, nakon odabira željenih stavki, pritiskom na tipku za nastavak biti proslijeđen na stranicu za prikaz financijskih izvještaja odabranih stavki.

```
public async Task<bool> SaveUserPreferences(string username, List<string> stocks, List<string> cryptos)
{
    var user = await GetUserByUsername(username);
    if (user == null) return false;

    user.Stocks = stocks;
    user.Cryptos = cryptos;

    _dbContext.Users.Update(user);
    await _dbContext.SaveChangesAsync();
    return true;
}
```

Slika 4.8. Spremanje korisnikovih odabira

Prilikom svake ponovne prijave korisnika poziva se metoda *GetUserPreferences()* kako bi se provjerilo postoje li u bazi podataka već spremljene željene kriptovalute ili kompanije za prijavljenog korisnika. Slika 4.9. prikazuje spomenutu metodu za dohvaćanje ranije spremljenih odabira.

```
public async Task<List<string>?> GetUserPreferences(string username, string preferenceType)
{
    var user = await GetUserByUsername(username);
    if (user == null) return null;

    if (preferenceType == "stocks")
    {
        return user.Stocks;
    }
    else if (preferenceType == "cryptos")
    {
        return user.Cryptos;
    }
    else
    {
        return null;
    }
}
```

Slika 4.9. Dohvaćanje već spremljenih korisnikovih odabira

4.6. Programsko rješenje dohvaćanja podataka

Prilikom pokretanja aplikacija provjerava postoje li podatci u bazi podataka ukoliko ne postoje podatci se dohvaćaju pomoću API-ja sa web stranice Polygon.io. Nakon dohvaćanja, povučeni podatci se uspoređuju sa spremljenih podatcima u bazi podataka te se spremaju ukoliko nisu dostupni u bazi. Iako aplikacija može funkcionirati bez spremanja podataka u bazu podataka, ali zbog mogućnosti greške pri pozivu API-ja podatci su spremljeni u bazu kako bi aplikacija mogla nesmetano raditi. Postoje dvije metode za dohvaćanje dionica i kriptovaluta, a jedina razlika je u linku prilikom dohvaćanja te dvije stavke te zbog toga dovoljna je samo slika 3.4kako bi prikazala programsko rješenje dohvaćanja podataka i pozivanje metode za raščlanjivanje.

```
public async Task<List<Stock>?> GetStockHistory(string symbol)
{
    try
    {
        var existingStockData = await _dbContext.Stocks
            .Where(s => s.Symbol == symbol && s.Date >= DateTime.Parse(startDate) && s.Date <= DateTime.Parse(endDate))
            .ToListAsync();

        if (existingStockData.Any())
        {
            return existingStockData;
        }

        string QUERY_URL = $"https://api.polygon.io/v2/aggs/ticker/{symbol}/range/1/day/{startDate}/{endDate}?apiKey={ApiKey}";
        var response = await _httpClient.GetAsync(QUERY_URL);

        if (response.IsSuccessStatusCode)
        {
            var jsonString = await response.Content.ReadAsStringAsync();
            var stockHistory = ParseStockHistory(jsonString, symbol);

            if (stockHistory != null && stockHistory.Any())
            {
                await SaveStockHistory(stockHistory);
            }

            return stockHistory;
        }
        else
        {
            _logger.LogError($"Failed to fetch stock history: {response.ReasonPhrase}");
            throw new Exception($"Failed to fetch stock history: {response.ReasonPhrase}");
        }
    }
```


Slika 4.10. Dohvaćanje podataka sa API-ja

Nakon uspješnog dohvaćanja podataka sa API-ja poziva se metoda raščlanjivanja (eng. *parse*) koja ima za zadatak pretvoriti dohvaćene podatke iz JSON zapisa u struktura podatka koje programski jezik razumije kako bi ih bilo moguće spremiti u bazu. Programsko rješenje metode za raščlanjivanje prikazana je na slici 4.11. .

```
private List<Crypto>? ParseCryptoWeekly(string jsonString, string symbol)
{
    try
    {
        using JsonDocument doc = JsonDocument.Parse(jsonString);
        JsonElement root = doc.RootElement;

        if (!root.TryGetProperty("results", out var results))
        {
            _logger.LogError("Results property not found in the JSON response.");
            return null;
        }

        var cryptoHistory = new List<Crypto>();

        foreach (JsonElement result in results.EnumerateArray())
        {
            var crypto = new Crypto
            {
                Symbol = symbol,
                Price = result.GetProperty("c").GetDouble(),
                Date = DateTimeOffset.FromUnixTimeMilliseconds(result.GetProperty("t").GetInt64()).DateTime
            };

            cryptoHistory.Add(crypto);
        }

        return cryptoHistory;
    }
    catch (Exception ex)
    {
        _logger.LogError($"Exception in ParseCryptoWeekly: {ex.Message}");
        return null;
    }
}
```

Slika 4.11. Metoda raščlanjivanja podataka na potrebne strukture podataka

4.7. Programsko rješenje baze podataka

Baza podataka podijeljena je u tri tablice: dionice, kriptovalute i korisnici. Slika 4.12. . prikazuje strukturu navedenih skupina unutar koda. Upravljanje bazom podataka postiže se putem klase *AppDbContext()* čija je zadaća upravljanje pristupa bazi kao i definiranje primarnih ključeva i izgleda baze podataka. Bazu podataka moguće je stvoriti putem kreiranja migracije, a promjene je moguće postići pomoću naredbe za ažuriranje migracije koja se upisuje u terminal VSCode-a. Slika 4.13. prikazuje strukturu klase *AppDbContext()*.

```
public class Crypto
{
    [Key]
    3 references
    public required string Symbol { get; set; }
    1 reference
    public double Price { get; set; }
    4 references
    public DateTime Date { get; set; }
}
public class Stock
{
    [Key]
    3 references
    public required string Symbol { get; set; }
    4 references
    public DateTime Date { get; set; }
    1 reference
    public double Price { get; set; }
}
```

```
public class User
{
    0 references
    public int Id { get; set; }
    3 references
    public string Username { get; set; } = string.Empty;
    2 references
    public string PasswordHash { get; set; } = string.Empty;
    2 references
    public string Salt { get; set; } = string.Empty;
    2 references
    public List<string> Stocks { get; set; } = new List<string>();
    2 references
    public List<string> Cryptos { get; set; } = new List<string>();
}
```

Slika 4.12. Programsko rješenje za strukture podataka kriptovaluta, dionica i korisnika

```
public class AppDbContext : DbContext
{
    2 references
    public DbSet<Stock> Stocks { get; set; }
    2 references
    public DbSet<Crypto> Cryptos { get; set; }
    4 references
    public DbSet<User> Users { get; set; }

    0 references
    public AppDbContext(DbContextOptions<AppDbContext> options) : base(options)
    {
    }

    0 references
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Stock>()
            .HasKey(s => new { s.Symbol, s.Date });

        modelBuilder.Entity<Crypto>()
            .HasKey(c => new { c.Symbol, c.Date });

        base.OnModelCreating(modelBuilder);
    }
```

Slika 4.13. Klasa za kontrolu baze podataka *AppDbContext()*

4.8. Programsko rješenje za prikaz podataka

Prikaz podataka unutar aplikacije moguć je na dva načina. Prvi način je prikaz podataka numeričkim putem odnosno prikaz kretanja cijena kroz povijest vidljivo kroz brojeve. Unutar aplikacije prikazuje se datum i vrijeme dohvaćanja podataka kao i cijena određenog financijskog

aspekta u tom promatranom trenutku. Na slici 4.14. moguće je vidjeti na koji način je postignut numerički prikaz dionica. Pritiskom na bilo koje očitovanje prelazi se na drugačiji način prikaza.

```
if (selectedCompany != null)
  buildDataCard(
    futureData: futureStocks,
    noDataText: 'No stock data available',
    itemBuilder: (stockData) => ListTile(
      title: Text('${stockData.date} - \$${stockData.price.toStringAsFixed(2)}',
      style: const TextStyle(fontSize: 16, color: Colors.white)), // Text
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => StockChartPage(
              symbol: selectedCompany!,
              isCrypto: false,
```

Slika 4.14. Prikaz dionica na numerički način

Drugi način prikaza podataka omogućen je putem interaktivnih linijskih grafikona, koji olakšavaju praćenje promjena cijena financijskih instrumenata kroz vrijeme. Ovaj vizualni prikaz omogućuje brzo uočavanje trendova rasta i pada, a jednostavnim prelaskom mišem preko pojedinih točaka na grafikonu, korisnik dobiva precizne informacije o cijeni i vremenu kada je ta cijena ostvarena. Na slici 4.15. moguće je vidjeti programsko rješenje za prikaz financijskih izvještaja putem linijskih grafikona.

```
lineBarsData: [
  LineChartBarData(
    spots: spots,
    isCurved: true,
    barWidth: 3,
    color: Colors.blueAccent,
    dotData: const FlDotData(show: true),
    belowBarData: BarAreaData(
      show: true,
      color: Colors.blueAccent.withOpacity(0.2),
    ), // BarAreaData
```

Slika 4.15. Prikaz dionica putem linijskog grafikona

5. TESTIRANJE I ANALIZA APLIKACIJE

Kako bi aplikacija uspješno radila uvijek ju je potrebno temeljito analizirati, a zatim i testirati. Kroz ovo poglavlje detaljno su opisane sve funkcionalnosti aplikacije, od procesa prijave i registracije korisnika, preko odabira i spremanja preferencija za praćenje financijskih instrumenata, do vizualizacije podataka kroz interaktivne grafikone.

Nakon detaljnog opisa svake funkcionalnosti, pristupili smo testiranju kako bismo osigurali da aplikacija radi ispravno i pruža očekivano korisničko iskustvo. Testiranje je obuhvatilo provjeru svih mogućih scenarija korištenja, uključujući ispravan unos podataka, rukovanje greškama, prikaz podataka, interakciju s grafikonima i navigaciju kroz aplikaciju. Posebna pažnja posvećena je testiranju na različitim uređajima i veličinama ekrana kako bismo osigurali optimalno korisničko iskustvo na svim platformama.

5.1. Kako koristiti aplikaciju

Nakon pokretanja aplikacije korisnik ima dvije mogućnosti. Mora izabrati između prijave ukoliko ima već registrirani račun ili odabrati registraciju ako prvi puta koristi aplikaciju. Nakon uspješne registracije korisnik se automatski prebacuje na prijavu gdje se prijavljuje sa istim podatcima. Nakon prijave korisniku se otvara početni zaslon gdje ima mogućnost odabira željenih financijskih izvještaja. Ukoliko želi da mu se isti spreme za iduće korištenje aplikacije ima opciju spremanja odabira. Nakon odabira željenih financijskih izvještaja, korisniku se otvaraju dva padajuća izbornika: jedan za odabir kriptovaluta, a drugi za dionice. Nakon odabira podatci se prikazuju numeričkim putem te klikom na određeni vremenski trenutak otvara se grafički prikaz podataka kroz vrijeme. Korisnik se u bilo kojem trenutku može odjaviti iz aplikacije kao i vratiti se na početni ekran i dodati još financijskih izvještaja.

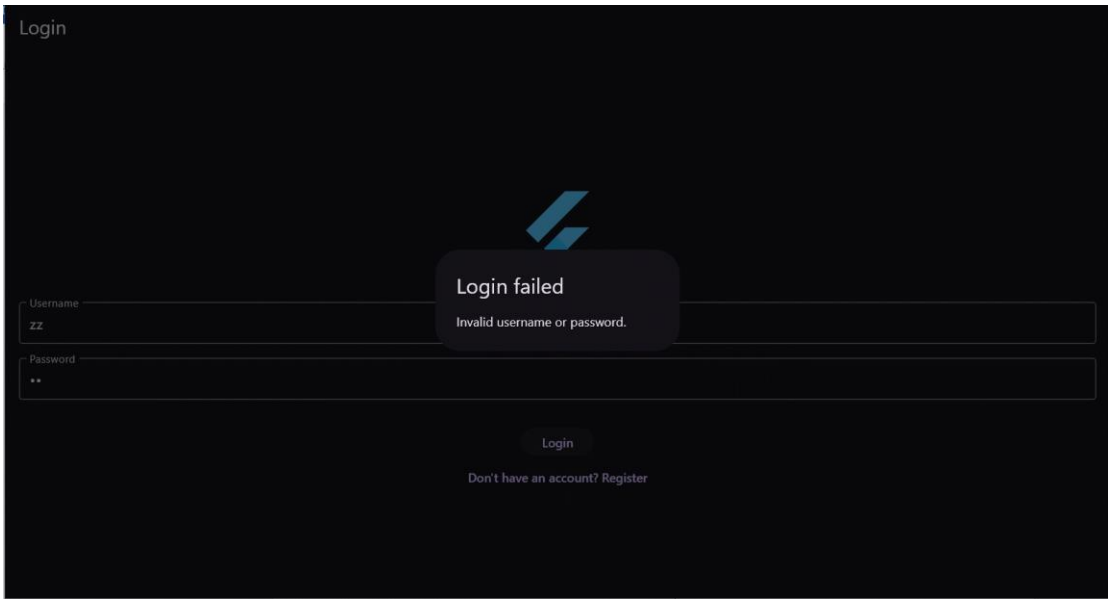
5.2. Testiranje rada aplikacije

5.2.1. Testiranje prijave korisnika u aplikaciju

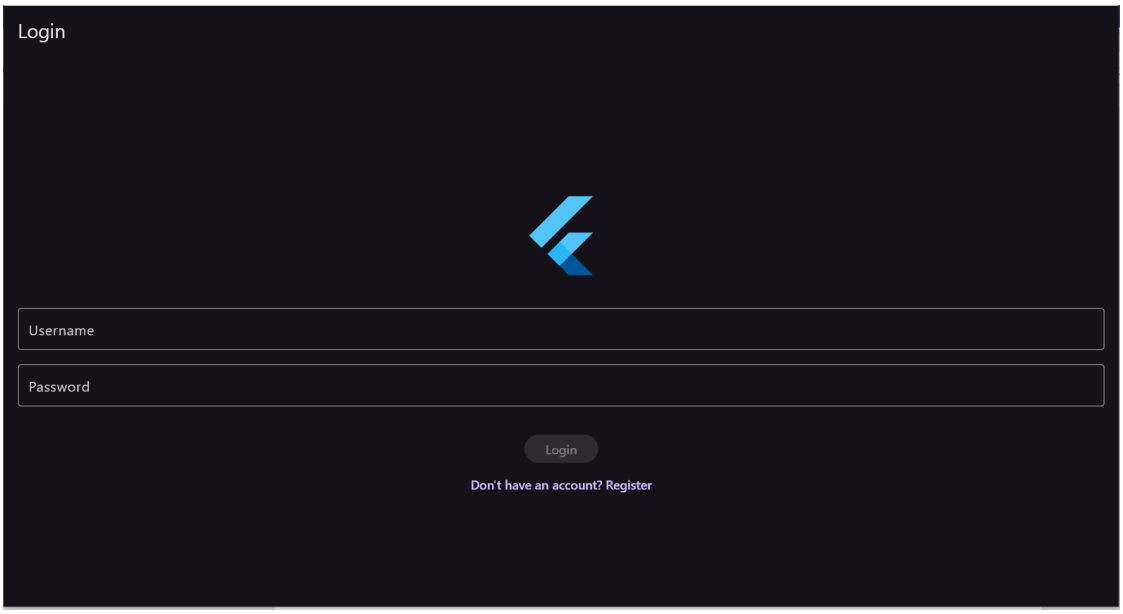
Prilikom prvog korištenja aplikacije korisnik mora izraditi račun odnosno registrirati se unutar aplikacije jer ukoliko se pokuša prijaviti bez korisničkog računa, aplikacija će poslati poruku kako korisnički račun ne postoji. Poruka o nepostojećem računu prikazana je na slici 5.1. . Ekran prijave jednostavno je konstruiran što je moguće vidjeti na slici 5.2., Korisnik može odabrati hoće li se registrirati ili prijavit te pritiskom na tipku za prijavu ili registraciju dobit će povratnu informaciju o uspješnosti postupka. Prilikom registracije ili prijave nije moguće izvršiti postupak bez da se

popune oba polja te time aplikacija pokriva slučajeve da se korisnik prijavi bez unošenja svih potrebnih podataka. Ako se korisnik pokuša registrirati sa već postojećim korisničkim imenom aplikacije mu to neće dopustiti te će mu biti poslana poruka o postojećem korisničkom imenu.

Nakon uspješne prijave, korisnik će biti proslijeđen na početni zaslon za odabir željenih financijskih izvještaja.



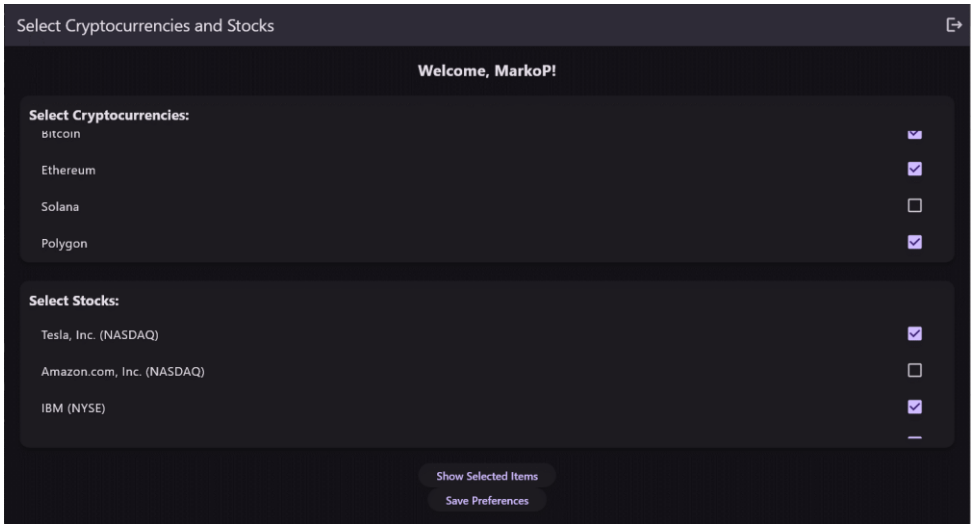
Slika 5.1. Pokušaj prijave sa neregistriranim računom



Slika 5.2. Ekran za prijavu i registraciju

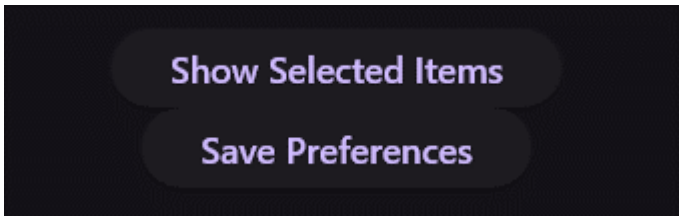
5.2.2. Testiranje odabira željenih financijskih izvještaja

Na početnom ekranu, korisniku je omogućeno da personalizira svoje iskustvo odabirom financijskih izvještaja koje želi pratiti. Dva intuitivna klizna prozora nude pregled dostupnih opcija, kako za dionice tako i za kriptovalute. Jednostavnim klikom na željenu stavku, korisnik je označava, čime izražava interes za praćenje njenih financijskih kretanja. Slika 5.3. nudi prikaz kliznih prozora za odabir željenih stavki. Također u desnom gornjem kutu kao što je vidljivo na slici 5.3. korisnik ima mogućnost odjave sa svog korisničkog računa.



Slika 5.3. Klizni prozori za mogućnost odabira željenih stavki

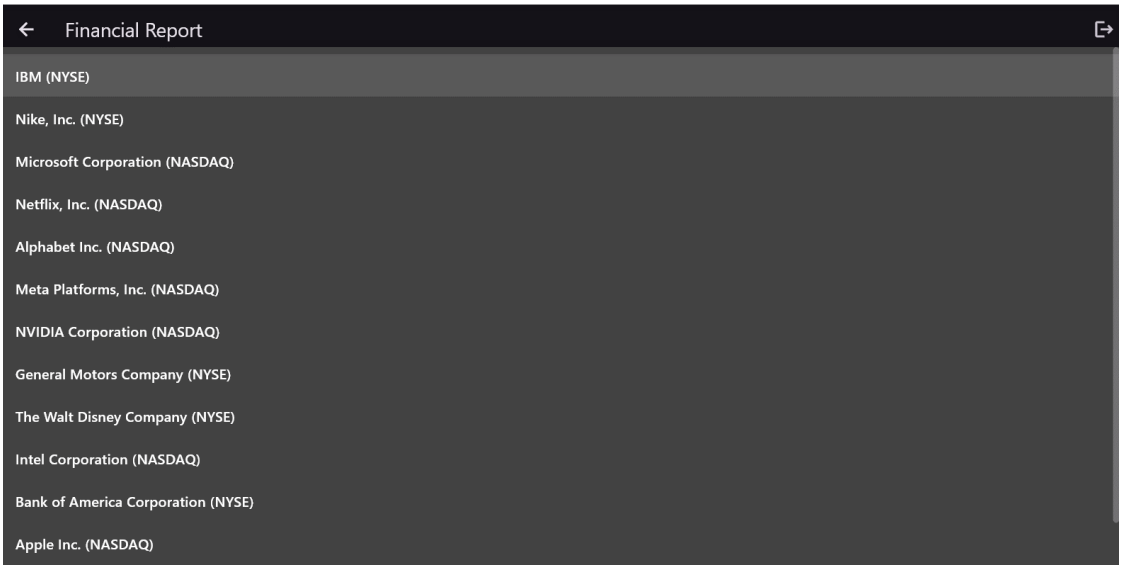
Nakon odabira željenih stavki korisnik ima pravo spremiti odabrane stavke kako bi prilikom idućeg korištenja aplikacije njegove želje ostale spremljene. Pritiskom na tipku za spremanje odabira korisnik će dobiti povratnu poruku ako je spremanje uspješno. Ukoliko korisnik ne želi ne mora spremiti odabrane stavke već ima pravo samo nastaviti na idući ekran, ali to znači da idući puta mora ponoviti postupak odabira željenih stavki. Korisnik, ako želi, lagano može ukloniti stavku koja ga je prije zanimala, a sada više ne. Korisnik može ukloniti kvačicu s već označene stavke ponovnim klikom na nju. Nakon što klikne tipku za spremanje, ta će stavka ostati neoznačena pri sljedećem otvaranju aplikacije. Ovim početnim ekranom postignuto je da se iskustvo svakog korisnika personalizira, jer iste stavke ne zanimaju svakog korisnika. Opcije spremanja odabranih stavki, kao i nastavak na idući ekran, prikazane su na slici 5.4..



Slika 5.4. Opcije spremanja i nastavka na idući ekran

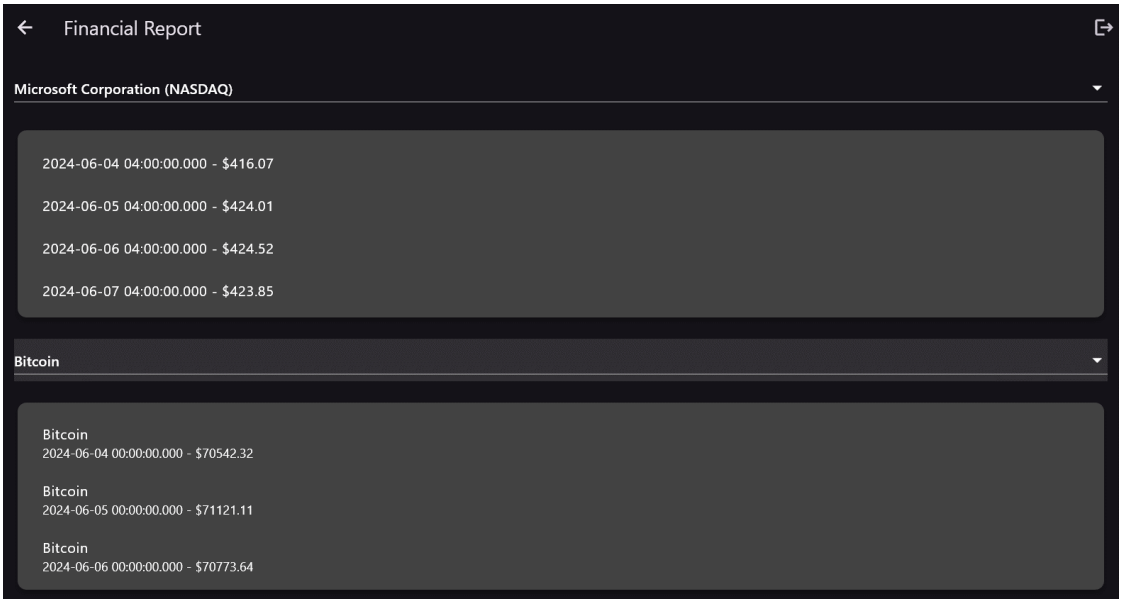
5.2.3. Testiranje numeričkog prikaza odabranih financijskih stavki

Nakon pritiska na tipku za dalje, korisniku se otvara ekran sa dva padajuća izbornika. Jedan izbornik služi za odabir dionica, dok drugi služi za kriptovalute. Kao što je prikazano na slici 5.5. gdje je prikazan padajući izbornik za odabir dionica. Na tom izborniku korisniku se prikazuju samo njegove prijašnje odabrane stavke.



Slika 5.5. Klizni prozor za odabira dionica

Odabirom stavke korisniku se otvara numerički prikaz dionica i kriptovaluta koje je odabrao u padajućim izbornicima. Prozori su klizni te prikazuju kretanje cijena kroz vrijeme odnosno kolika je cijena te stavke bila u tom trenutku. Kako to izgleda prikazano je na slici 5.6. Korisnik u svakom trenutku ima mogućnosti odjave pritiskom na tipku za odjavu koja se nalazi u gornjem desnom kutu.



Slika 5.6. Numerički prikaz odabranih stavki

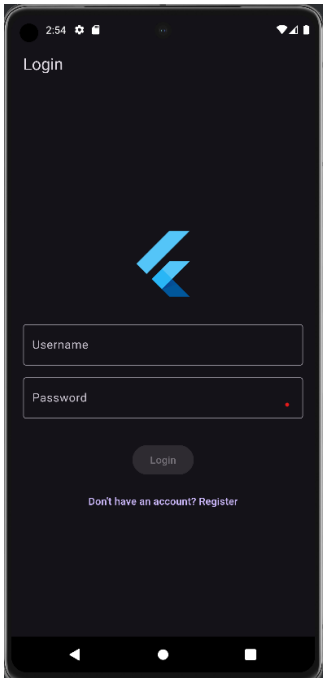
5.2.4. Testiranje grafičkog prikaza odabranih financijskih stavki

Posljednja, ali i najvažnija, stavka aplikacije je grafički prikaz dionica i kriptovaluta. Prilikom izrade aplikacije korišten je linijski grafikon jer se on najbolje pokazao za prikaz financijskih stavki te većina modernih aplikacija, koje imaju grafički prikaz financijskih izvještaja, koriste linijski grafikon. Grafikonu se pristupa tako što se na ekranu za prikaz numeričkih stanja, pritiskom na jedno od očitovanja, otvara se grafikon sa prikazom kretanja cijene te određene stavke kroz vrijeme. Izgled grafikona prikazan je na slici 5.7.. On se sastoji od naslova koju stavku prikazuje, y i x osi. Y os prikazuje cijene dok x os prikazuje vrijeme. Grafikon nudi mogućnost za preciznije očitovanje cijene u određenom trenutku tako što prelaskom preko određene točke u vremenu prikazati će se očitana cijena te vrijeme u kojem je ta cijena postignuta. Kao i na svakom ekranu korisnik ima pravo odjave u desnom gornjem kutu.

Slika 5.7. Grafički prikaz kriptovalute Litecoin

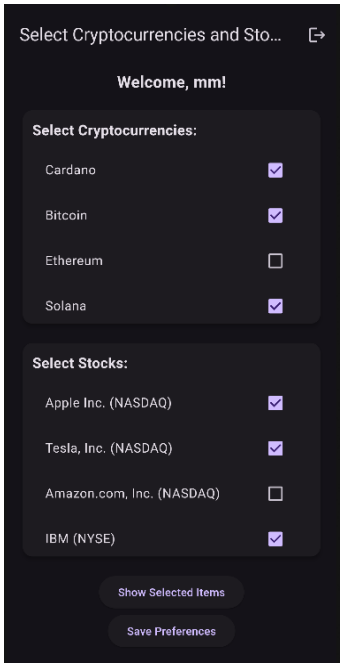
5.3. Testiranje aplikacije na mobilnom uređaju

Zbog velike prednosti programskog alata Flutter, aplikaciju je moguće pokrenuti na mobilnom uređaju. Ujedno je to i jedna od najvećih prednosti Flutter-a gdje je moguće jedan kod pokrenuti na više različitih platformi (Android, stolno računalo, iOS). Slika 5.8. prikazuje ekran za prijavu i registraciju na mobilnom uređaju. Kao što je moguće vidjeti, aplikacija je prilagođena veličini ekrana te bez poteškoća prikazuje ekran za prijavu.



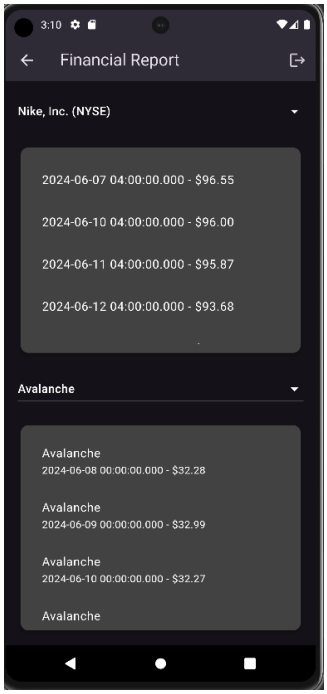
Slika 5.8. Prikaz ekrana za prijavu na mobilnom uređaju

Nadalje, slika 5.9. prikazuje izgled početnog ekrana na mobilno uređaju gdje je moguće vidjeti da aplikacija bez poteškoća učitava sve odabire te su svi dobro vidljivi. Tipke spremanje odabira kao i za nastavak na iduću stranicu nalaze se na dnu ekrana.



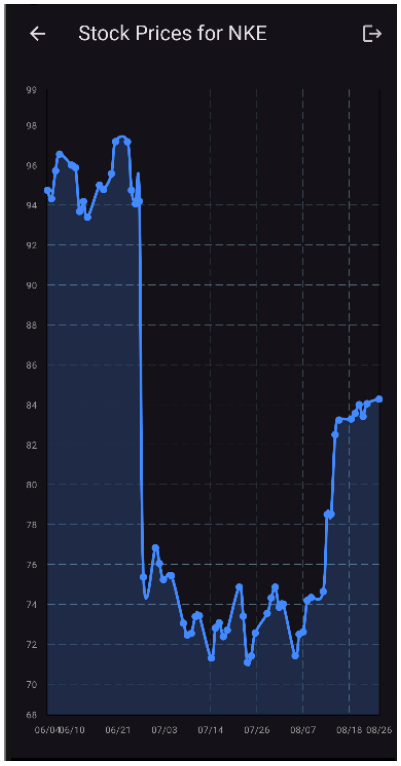
Slika 5.9. Početni ekran na mobilnom uređaju

Numerički prikaz podataka uspješno je prilagođen mobilnom uređaju, možda čak i bolje nego na stolnom računalu te klizni prozori tu dobivaju svoju glavnu ulogu. Prilagodbu numeričkog prikaza podataka moguće je vidjeti na slici 5.10.



Slika 5.10. Numerički prikaz odabranih stavki na mobilnom uređaju

Jedina mana kod korištenja aplikacije putem mobilnog uređaja pokazuje se kod grafičkog prikaza financijskog izvještaja, Prilikom grafičkog prikaza zbog ograničene širine uređaja grafički prikaz nije dobro vidljiv, odnosno teško je očitati kretanje cijena. Problem je moguće vidjeti na slici 5.11..



Slika 5.11. Grafički prikaz odabranih stavki na mobilnom uređaju

Rješenje za taj problem je okretanje zaslona mobilnog uređaja horizontalno kako bi se iskoristila dužina samoga ekrana te time grafički prikaz postaje upotpunjen kao što je moguće vidjeti na slici 5.12. .

Slika 5.12. Horizontalni grafički prikaz odabranih stavki na mobilnom uređaju

5.4. Analiza aplikacije

Aplikacija je uspješno prošla testiranje te radi bez ikakvih poteškoća. Kroz testiranje aplikacije moglo se uvidjeti da zadovoljava sve potrebne korisnika koji traže jednostavnu aplikaciju za prikaz

financijskih izvještaja. Aplikacija je namijenjena prvenstveno početnicima jer ne sadrži mnoštvo alata kao što sadrže slične aplikacije. Prednost aplikacije je lakoća korištenja i prezentacija osnovnih informacija. Iako je jednostavnija od ostalih aplikacija dovoljno je dobro razvijena da bez poteškoća može napraviti personalizirani izvještaj koji će korisniku uvijek prikazati kretanje cijena dionica i kriptovaluta. Mana aplikacije je ograničeni broj kriptovaluta i dionica te nedostatak mogućnosti praćenja drugih financijskih instrumenata poput obveznica, fondova ili roba. Također, aplikacija trenutno ne nudi napredne alate za analizu poput tehničkih indikatora ili mogućnosti postavljanja upozorenja o promjenama cijena. Dodatno, neki korisnici bi mogli preferirati mogućnost sinkronizacije podataka s drugim financijskim platformama ili alatima koje već koriste. Velika prednost aplikacije je mogućnost korištenja na svakom dostupnom uređaju.

Nadalje, iako je aplikacija namijenjena početnicima, ima nedostataka u pogledu naprednijih alata za analizu financijskih podataka. Aplikacija je vrlo jednostavna i intuitivna te će time zadovoljiti i iskusnije korisnike. Najveća prednost aplikacije je mogućnost personalizacije odabira jer svaki korisnik ima mogućnost odabrati koji financijski izvještaji ga zanimaju. Ovu mogućnost čak ne nude i mnogo naprednije aplikacije te je samim time ovo velika prednost aplikacije.

6. ZAKLJUČAK

Financijska tržišta postala su iznimno dinamična i kompleksna te je zbog toga nastala potreba za pravovremenim i pouzdanim informacijama. Razvijena aplikacija za nadzor financijskog tržišta predstavlja odgovor na ovaj izazov, pružajući korisnicima sveobuhvatan i intuitivan alat za praćenje, analizu i donošenje informiranih financijskih odluka. Temeljito testiranje aplikacije potvrdilo je njenu funkcionalnost i usklađenost sa zadanim specifikacijama. Korisnici sada imaju pristup ažurnim podacima o dionicama i kriptovalutama, a alati za analizu i personalizirane preporuke omogućuju im dublji uvid u tržišne trendove i potencijalne investicijske prilike.

Aplikacija je prošla kroz niz testova te ih je sve uspješno položila. Aplikacija nudi vrlo jednostavan način za praćenje financijskog tržišta. Pomoću svog jednostavnog korisničkog sučelja i mogućnosti kreiranja vlastitog računa i odabira željenih financijski stavki aplikacija pokazuje kako je kreirana za korisnika odnosno da bude prilagođena korisniku.

Iako je aplikacija u ovom obliku već funkcionalna i korisna, postoji prostor za daljnja unapređenja u vidu implementacija dodatnih funkcionalnosti poput mogućnosti postavljanja upozorenja o promjenama cijena, integracije s drugim financijskim platformama i uvođenje naprednih alata. Primjena naprednih algoritama za analizu dodatno bi obogatile aplikaciju i povećale njenu vrijednost za korisnike.

LITERATURA

[1] T. Economist, The role of financial markets, *The Economist,* 18. 03. 2023.

[2] Yahoo, Yahoo Finance, Yahoo, 1997. [Mrežno]. Dostupno: https://finance.yahoo.com/. [Pokušaj pristupa 25 06 2024].

[3] Google Finance, Google, 2006. [Mrežno]. Dostupno: https://www.google.com/finance/. [Pokušaj pristupa 25 06 2024].

[4] TradinView, 2011. [Mrežno]. Dostupno: https://www.tradingview.com/. [Pokušaj pristupa 25 06 2024].

[5] Robinhood, Robinhood, 2013. [Mrežno]. Dostupno: https://robinhood.com/us/en/about-us/. [Pokušaj pristupa 26 06 2024].

[6] Microsoft, Visual Studio Code, Microsoft, 2015. [Mrežno]. Dostupno: https://code.visualstudio.com/. [Pokušaj pristupa 26 06 2024].

[7] C#, Microsoft, 2000. [Mrežno]. Dostupno: https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/overview. [Pokušaj pristupa 26 06 2024].

[8] Flutter FAQ, Flutter, [Mrežno]. Dostupno: https://docs.flutter.dev/resources/faq. [Pokušaj pristupa 26 06 2024].

[9] Microsoft SQL server, Microsoft, [Mrežno]. Dostupno: https://learn.microsoft.com/en-us/sql/sql-server/what-is-sql-server?view=sql-server-ver16. [Pokušaj pristupa 26 06 2024].

[10] Microsoft, SQL Server Manager Studio., Microsoft, [Mrežno]. Dostupno: https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16. [Pokušaj pristupa 30 08 2024].

[11] Polygon.io, [Mrežno]. Dostupno : https://polygon.io/about. [Pokušaj pristupa 14 09 2024]

SAŽETAK

U ovom završnom radu cilj je bio razvoj mobilne aplikacije za nadzor financijskog tržišta koja korisnicima omogućava praćenje i analizu podataka o dionicama i kriptovalutama. Aplikacija koristi Polygon.io API za dohvat povijesnih podataka o cijenama, a pohranjuje ih u lokalnu bazu podataka pomoću Entity Framework Core-a. Korisnici se mogu registrirati i prijaviti, te spremiti svoje preferencije u vezi s dionicama i kriptovalutama koje žele pratiti. Aplikacija prikazuje podatke u obliku liste i grafikona, omogućavajući korisnicima da vizualiziraju trendove i donose informirane odluke.

Ključne riječi: dionice, financije, Flutter, kriptovalute, mobilna aplikacija, tržište

ABSTRACT

Title: Financial market monitoring application

The objective of this thesis was to develop a mobile application for financial market monitoring, allowing users to track and analyze data on stocks and cryptocurrencies. The application utilizes the Polygon.io API to fetch historical price data, which is then stored in a local database using Entity Framework Core. Users can register and log in, and save their preferences regarding the stocks and cryptocurrencies they wish to track. The application presents data in the form of lists and charts, enabling users to visualize trends and make informed decisions.

Keywords: cryptocurrencies, finance, Flutter, market, mobile application, stocks