Ionic aplikacija za praćenje novosti o kriptovalutama

Kosić, Jerko

Master's thesis / Diplomski rad

2021

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:147510

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-26**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA

Sveučilišni studij

IONIC APLIKACIJA ZA PRAĆENJE NOVOSTI O KRIPTOVALUTAMA

Diplomski rad

Jerko Kosić

Osijek, 2021.

SADRŽAJ

1. UVOD 1
1.1. Zadatak diplomskog rada 1
2. PREGLED POSTOJEĆIH RJEŠENJA 2
2.1. CryptoMarketCap 2
2.2. Crypto App 2
2.3. CoinGecko 2
3. TEHNOLOGIJE KORIŠTENE ZA RAZVOJ APLIKACIJE 3
3.1. Internet (Web) aplikacije 3
3.2. Android aplikacije 4
3.3. iOS aplikacije 5
3.4. Ionic okvir 6.12.3 5
3.4.1. Ionic i angular navigacija 7
3.4.2. Ionic kartice 8
3.4.3. Cordova 9
3.4.4. Otklanjanje pogrešaka na Android i iOS uređaju 10
3.5. Angular okvir 10.0.14 11
3.6. CryptoCompare API 12
4. PROCES RAZVOJA APLIKACIJE 15
4.1. Ionic stranice 16
4.2. Navigacija u aplikaciji 17
4.3. Pozivanje API-ja 19
4.4. Prikaz liste kriptovaluta 20
4.5. Prikaz detalja kriptovalute 23
4.5.1. ApexCharts 25
4.6. Prikaz novosti iz svijeta kriptovaluta 26

5. KRIPTOVALUTE 28
5.1. Razlike fiat i kripto valuta 28
5.2. Blockchain 29
5.3. Graf svijeća 29
6. NAČIN KORIŠTENJA APLIKACIJE 31
7. ZAKLJUČAK 34
LITERATURA 35
SAŽETAK 36
ABSTRACT 37
PRILOZI 38

1. UVOD

Tema ovog završnog rada je izrada Ionic aplikacije koja u bilo kojem trenutku omogućuje praćenje vrijednosti, te najnovije vijesti iz svijeta kriptovaluta. S obzirom kako im je vrijednost i popularnost sve veća, korisnici istih, ali i oni koji to žele postati imaju potrebu pratiti stanje u bilo kojem trenutku. Slična rješenja već postoje i neka od njih su prikazana u drugom poglavlju. Danas većina ljudi ima pristup android, desktop ili iOS uređaju u skoro svakom trenutku, te je zbog toga odabran Ionic okvir za izgradnju aplikacije. Ionic omogućuje izgradnju web aplikacije sa responzivnim dizajnom i pretvorbu te web aplikacije u android i iOS aplikaciju bez pisanja dodatnog koda, te se jednom napisani kod koristi za sve platforme. Ionic može koristiti razne okvire za izgradnju aplikacija, kao što su Angular, React, Vue.js ili čisti JavaScript. Svaki okvir ima svoje prednosti i nedostatke, ali u ovom diplomskom radu se koristi Angular. Za dohvaćanje vrijednosti i novosti o kriptovalutama koristi se API (Application Programming Interface) koji omogućuje aplikaciji dohvaćanje informacija sa udaljenog servera. Način izrade aplikacije, tehnologije koje su korištene, te način rada API-a će biti opisano u trećem poglavlju. Dakle biti će opisane web, android i iOS aplikacije, zatim Ionic i angular okviri koji su se koristili za izradu aplikacije, te CryptoCompare API. Proces razvoja aplikacije će biti opisan u četvrtom poglavlju. Biti će opisane Ionic stranice koje su glavni dio aplikacije, zatim način rada navigacije u aplikaciji, te pozivanje API-ja i način prikaza liste i detalja kriptovaluta. To je vrsta digitalnog novca koja postoji samo u električnom obliku, a više o njima će biti opisano u petom poglavlju. Također će biti opisana razlika između fiat i kripto valuta, te što je to blockchain i graf svijeća (engl. candle graph). Nakon toga je opisan način korištenja aplikacije.

1.1. Zadatak diplomskog rada

U ovom diplomskom radu potrebno je napraviti Ionic aplikaciju koja prikazuje najnovije vrijednosti kriptovalute, te najnovije vijesti o kriptovalutama.

2. PREGLED POSTOJEĆIH RJEŠENJA

U ovom poglavlju opisana su postojeća rješenja koja omogućuju praćenje kriptovaluta.

2.1. CryptoMarketCap

Crypto Market Cap je android aplikacija koja je slična aplikaciji u ovom diplomskom radu. Također omogućuje korisniku praćenje trenutne cijene kriptovaluta, te prikaz grafa koji prikazuje cijenu kroz vrijeme. Osim toga daje korisniku i druge podatke kao cijenu valute u Bitcoinu, ukupnu vrijednost kupljene i prodane valute u dolarima. Također daje korisniku mogućnost gledanja posljednjih novosti, te osim toga omogućuje korisniku povezivanje sa svojim virtualnim novčanicima kako bi pratio stanje na računu, te pratio svoje transakcije. Ova aplikacija nudi više mogućnosti nego aplikacija u ovom diplomskom radu. [1]

2.2. Crypto App

Crypto App je aplikacija dostupna na android i iOS platformi. Slično kao i aplikacija opisana u ovom diplomskom radu nudi korisnicima mogućnost praćenja trenutne vrijednosti kriptovaluta, prikaz cijene kroz vrijeme na grafu svijeća (engl. candle graph), te prikaz najnovijih novosti. Osim toga ova aplikacija omogućuje korisniku postavljanje alarma koji će ga obavijestiti o promjeni cijene valute, te postavljanje widgeta koji korisniku olakšavaju praćenje cijene valute. Ova aplikacija nudi par mogućnosti više od aplikacije u ovom diplomskom radu. [2]

2.3. CoinGecko

CoinGecko je aplikacija dostupna na android i iOS platformi. Ova aplikacija ima najbolju ocjenu od aplikacija navedenih pod poglavljem Pregled postojećih rješenja. Svaka od ovih aplikacija ima više od milijun preuzimanja. Isto kao i druge aplikacije opisane i ova aplikacija nudi praćenje trenutne vrijednosti kriptovaluta, prikaz grafa sa vrijednostima kroz vrijeme, te neke dodatne informacije kao ukupan volumen kupljene i prodane valute u zadnja dvadeset i četiri sata. Također omogućuje korisnicima korištenje widgeta kako bi korisnik na zaslonu mobitela mogao pratiti vrijednosti svojih omiljenih valuta, te kalkulator za pretvaranje valuta iz jedne u drugu. [3]

3. TEHNOLOGIJE KORIŠTENE ZA RAZVOJ APLIKACIJE

Za izradu aplikacije odabran je Ionic okvir jer omogućuje jednostavnu izradu web aplikacije, te jednostavnu pretvorbu te web aplikacije u android i iOS aplikacije. Odnosno jednom napisani kod se koristi za sve platforme. Izradu web aplikacije uvelike olakšava korištenje Angular okvir zajedno sa Ionicom. Aplikacija prikazuje novosti i cijene kriptovaluta, a za dohvaćanje tih podataka se koristi CryptoCompare API.

3.1. Internet (Web) aplikacije

Web aplikacija je računalni program koji koristi web preglednike i web tehnologiju za izvršavanje zadataka putem interneta. Milijuni tvrtki koriste internet kao isplativ komunikacijski kanal. Omogućuje im razmjenu informacija sa svojim ciljnim tržištem i obavljanje brzih i sigurnih transakcija.

Web aplikacije koriste kombinaciju skripti na strani poslužitelja (PHP i ASP) za rukovanje pohranjivanjem i dohvatom informacija, a skripte na strani klijenta (JavaScript i HTML) za predstavljanje informacija korisnicima. To korisnicima omogućuje interakciju s tvrtkom pomoću internetskih obrazaca, sustava za upravljanje sadržajem, kolica za kupnju i još mnogo toga. Osim toga, aplikacije zaposlenicima omogućuju stvaranje dokumenata, razmjenu informacija, suradnju na projektima i rad na zajedničkim dokumentima bez obzira na lokaciju ili uređaj. Web aplikacije obično su kodirane na jeziku koji podržava preglednik, kao što su JavaScript i HTML jer se ti jezici oslanjaju na preglednik za izvršavanje programa. Neke su aplikacije dinamične i zahtijevaju obradu na strani poslužitelja. Drugi su potpuno statični i ne zahtijevaju obradu na poslužitelju. Web aplikacija zahtijeva web poslužitelj za upravljanje zahtjevima klijenta, poslužitelj aplikacija za izvršavanje traženih zadataka, a ponekad i bazu podataka za pohranu podataka. Tehnologija poslužitelja aplikacija kreće se od ASP.NET, ASP i ColdFusion, do PHP i JSP. Ovako izgleda tipičan tijek web aplikacija:

1. Korisnik pokreće zahtjev web poslužitelju putem Interneta, bilo putem web preglednika ili korisničkog sučelja aplikacije
2. Web poslužitelj prosljeđuje ovaj zahtjev odgovarajućem poslužitelju web aplikacija
3. Poslužitelj web aplikacija izvršava traženi zadatak - poput upita u bazi podataka ili obrade podataka - zatim generira rezultate traženih podataka

4. Poslužitelj web aplikacija šalje rezultate na web poslužitelj sa traženim podacima ili obrađenim podacima
5. Web poslužitelj odgovara klijentu sa traženim podacima koji se zatim pojavljuju na korisnikovom zaslonu

Web aplikacije uključuju internetske obrasce, kolica za kupovinu, obrađivače teksta, proračunske tablice, uređivanje videozapisa i fotografija, pretvorbu datoteka, skeniranje datoteka i programe za e-poštu kao što su Gmail, Yahoo. Još neki primjeri web aplikacija su Facebook, Twitter, Wikipedija itd. Web aplikacije rade na više platformi bez obzira na OS ili uređaj sve dok je preglednik kompatibilan, te svi korisnici pristupaju istoj verziji, uklanjajući probleme s kompatibilnošću. [4]

3.2. Android aplikacije

Razvoj Android aplikacija čini veliki dio svih projekata razvoja mobilnih aplikacija. Svaka aplikacija prolazi kroz vlastiti razvojni proces, ovisno o timu koji je gradi, kako funkcionira itd. Međutim, najbolje Android aplikacije razvijene su pomoću izvornih programskih jezika, Jave ili Kotlina, i koriste Android Studio i Android SDK. Izvorne Android aplikacije moraju koristiti posebne razvojne alate i programske jezike kako bi se povećale mogućnosti izvornih značajki prisutnih na Android uređajima. Izvorni razvoj mobilnih aplikacija zahtijeva specijalizaciju, pa iako postoje sličnosti u usporedbi razvoja Androida s razvojem iOS -a, razvoj aplikacija za Android potpuno je drugačiji. Izvorni Android razvoj koristi jedan od dva programska jezika: Java ili Kotlin. Sve do nedavno Java je bila službeni programski jezik za razvoj Android aplikacija. Iako Java više nije službeni jezik koji Google promovira za razvoj Androida, Java se još uvijek široko koristi i snažno podržava od strane Googlea i Trgovine Play. Java postoji već duže vrijeme i ima ogromnu zajednicu za podršku koja programerima pomaže u pronalaženju odgovora na rješavanje problema i dijeljenju savjeta, trikova i najboljih praksi kada je u pitanju korištenje programskog jezika za razvoj mobilnih aplikacija. Nedostatak Jave je što za početnike ovaj jezik može biti previše složen korištenje. Također, Android SDK dodaje još jedan sloj složenosti kada se koristi s Javom. Sve u svemu, Java je izvrstan programski jezik za razvoj Android aplikacija, čak i ako ga je malo teško shvatiti.

Kotlin je službeni programski jezik za razvoj Android aplikacija jer je Google to odlučio 2019. godine. Zgodna stvar kod Kotlina je to što je interoperabilan s Javom, odnosno Kotlin kod se

može dodati u Java aplikaciju i obrnuto. Ako je potrebno migrirati Java aplikaciju u Kotlin, to se može učiniti u fazama, a ne odjednom, zahvaljujući Kotlinovoj interoperabilnosti. On radi i na Java virtualnom stroju. Osim toga, iako je izvorni Android programski jezik, može se koristiti i u drugim razvojnim projektima i dijeliti s iOS programerima. Mnogo jednostavniji od Jave. To mu daje prednost u razvoju Androida. U usporedbi s Javom, Kotlin kod je kraći, lakši za čitanje, sigurniji. Kotlinove prednosti uvelike su posljedica njegove jednostavnosti. Zahtijeva manje redaka koda, lakše ga je čitati, održavati i osigurati. Ukratko, to je moderan programski jezik koji je dizajniran kako bi bio jednostavan.

Naravno, ne mora se nužno koristiti Kotlin ili Java za izradu Android aplikacije. No, za korištenje drugih programskih jezika u razvoju Android aplikacija moraju se koristiti dodatni okviri i razvojni alati. Android aplikacije izrađene su i sa C++, Pythonom, C#, HTML-om, CSS-om i JavaScriptom, Rustom. Dva najvažnija alata u razvoju Android aplikacija su:

- Android Studio
- Android SDK

Android aplikacije se mogu napraviti bez korištenja bilo kojeg od ovih alata, ali oba, osobito kada se koriste zajedno, olakšavaju razvoj Android aplikacija. [5]

3.3. iOS aplikacije

iOS (iPhone Operating System) aplikacija je softverska aplikacija razvijena za upotrebu na Appleovim iPhone uređajima. iPhone aplikacije dostupne su putem Apple App trgovine i osmišljene su za rad na Appleovom mobilnom operativnom sustavu iOS, koji pokreće iPhone, kao i Appleove iPad i iPod Touch uređaje. Apple potiče programere da programiraju vlastite aplikacije za iPhone za preuzimanje putem App Store-a, a tvrtka je izdala SDK s uzorcima projekata koda koji će programerima pomoći u početku. Podrška za kupnje unutar aplikacija u iPhone aplikacijama nudi programerima dodatnu mogućnost prihoda. [6]

3.4. Ionic okvir 6.12.3

Ionic framework je skup korisničkih sučelja otvorenog koda za izgradnju učinkovitih, visokokvalitetnih mobilnih aplikacija, desktop aplikacija i progresivnih web aplikacija koje koriste web tehnologije kao što su HTML, CSS i JavaScript. Omogućuje programerima

pokretanje aplikacije na svim platformama kada je jednom naprave. Napravili su ga Max Lynch, Ben Sperry i Adam Bradley iz tvrtke Drifty Co. 2013. Prva beta verzija Ionic okvira objavljena je u ožujku 2014. godine. Ionic okvir se uglavnom fokusira na frontend korisničko iskustvo ili interakciju s korisničkim sučeljem, koje upravlja izgledom aplikacije. Lako ga je naučiti i može se integrirati s drugim bibliotekama ili okvirima kao što su Angular, Cordova itd. Službeno, ionic okvir je integriran s Angular-om, ali također pruža podršku za Vue.js i React.js. Prednosti Ionic okvira za razvoj aplikacija su:

1. **Lagan za učenje:** Ako programer ima osnovno znanje o CSS-u, HTML-u i JavaScriptu, tada je učenje i razvoj aplikacija pomoću Ionic okvira vrlo jednostavno. Također omogućuje razvojnim tvrtkama prelazak na Ionic ako njihovi zahtjevi i potrebe zahtijevaju razvoj hibridnih aplikacija.
2. **Jednostavna dokumentacija:** Ionic pruža vrlo dobru i strukturiranu dokumentaciju. Službena dokumentacija pokriva većinu stvari potrebnih programerima.
3. **Cross-platform:** Ionic aplikacija se može primijeniti na više platformi kao što su iOS, Android, stolna računala i web, sve s jednom bazom kodova. Ove aplikacije mogu pisati jednom i rade svugdje.
4. **Korisničko sučelje:** Korisničko sučelje, poput tema i komponenti, se može lako uređivati i mijenjati. Ionic okvir omogućuje svojim komponentama prilagodbu ovisno o platformi na kojoj se aplikacija izvodi.

Nedostatci Ionic okvira za razvoj aplikacija su:

1. **Izvedba:** Izvedba aplikacije nije tako dobra u usporedbi sa izvornim mobilnim aplikacijama. Međutim razlika nije primjetna za većinu korisnika, jer su mobilni uređaji jako brzi i mogu podržati taj nedostatak u izvedbi.
2. **Sigurnost:** Aplikacija ne pruža toliko sigurnosti kao izvorna aplikacija. Na primjer, ako razvijate financijsku aplikaciju, npr. Aplikaciju za banku, Ionic okvir se ne preporučuje.
3. **Ograničena izvorna funkcionalnost:** Neke izvorne funkcije nisu dostupne u Ionic okviru. U tom slučaju je potrebno iskoristiti dodatke koji će zamjeniti te funckije.
4. **Nije dobar za video igre:** Nije savršen za vrhunske grafičke aplikacije ili video igre.

5. **Potreba za stručnjacima:** JavaScript može biti teško naučiti. Nekada postoji potreba za nekim tko ima duboko znanje o naprednim bibliotekama i tehnologijama, uključujući Angular, Cordovu itd.

Ionic tehnologija je još uvijek u razvoju. Redovito mijenja svoju podršku i standard, a biblioteke se u svakom trenutku mogu potpuno prepisati. [7]

3.4.1. Ionic i angular navigacija

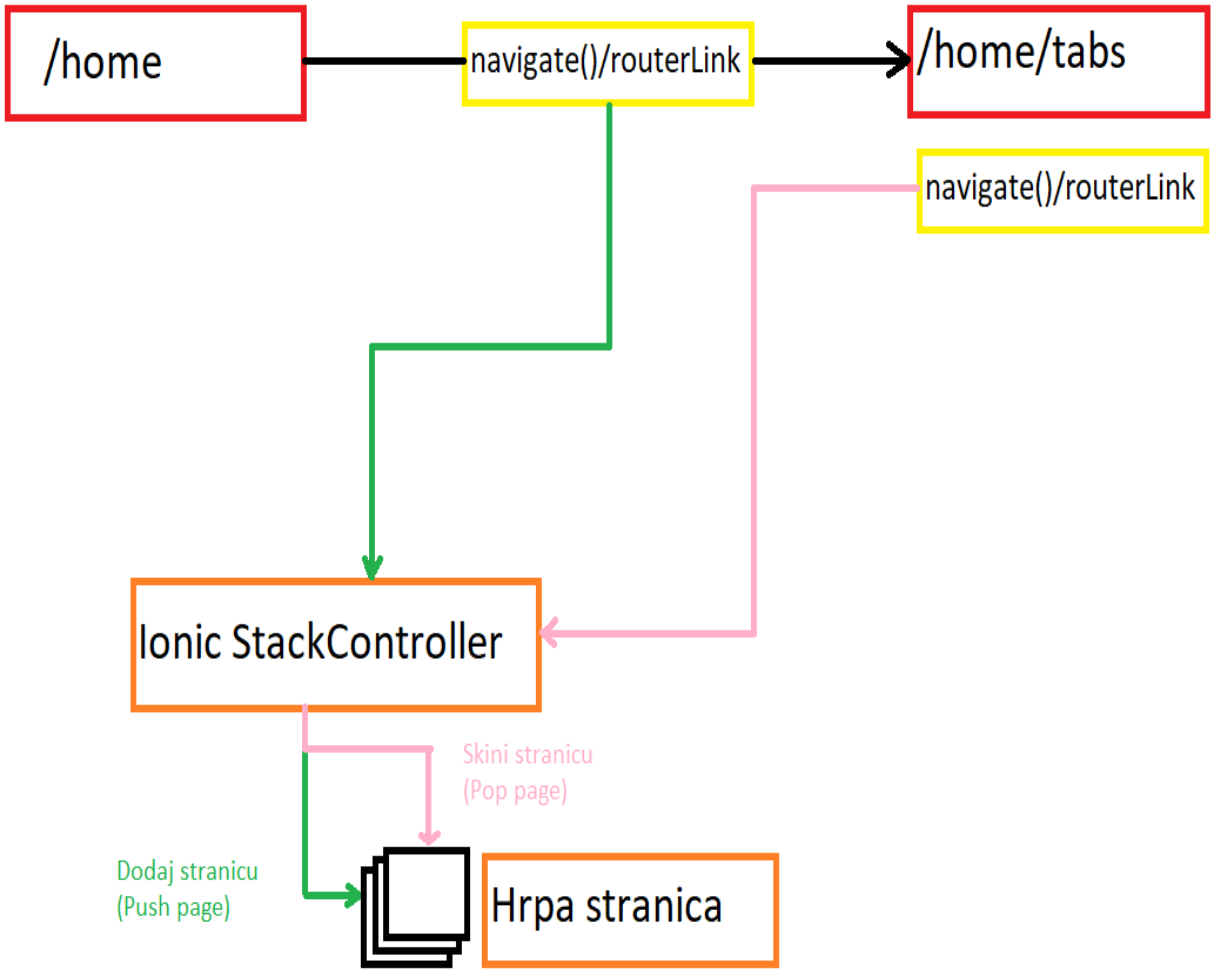
Kako bi se mogla napraviti Ionic aplikacija sa angular-om važno je razumjeti kako radi navigacija. Za navigaciju u aplikaciji se koristi Angular router. Rute kojima će aplikacija „hodati“ se postavljaju u *routing* datoteci. To je datoteka koja sadrži konfiguraciju ruta aplikacije. Na slici 3.1. se može vidjeti početna konfiguracija ruta ove aplikacije.

```
const routes: Routes = [
  {
    path: 'home',
    loadChildren: () => import('./home/home.module').then( m => m.HomePageModule)
  },
  {
    path: '',
    redirectTo: 'home',
    pathMatch: 'full'
  },
];
```

Slika 3.1. Početna konfiguracija ruta aplikacije

Kao što se vidi na slici 3.1. imamo rutu *home* koja učitava home modul, i praznu rutu koja preusmjerava korisnika na rutu *home*. Prazna ruta se u većini slučajeva prva poziva kada korisnik želi koristiti aplikaciju. Kada se rute postave može se koristiti *routerLink* ili metode za navigaciju (kada se ubrizga usmjerivač) kako bi se prebacivalo s jedne stranice na drugu. Kada se *Angular Router* koristi u Ionic aplikaciji, Ionic zapravo na neki način obavije funkcionalnosti *Angular routing-a*, a to radi kako bi dodao lijepe prijelaze prilikom promjene stranice. On razmišlja o navigaciji ili različitim stranicama kao hrpi stranica. Uobičajeno mobilna aplikacija, kada vrši navigaciju, prikazuje jednu stranicu, te s nje prelazi na drugu, te ima mogućnost vraćanja na staru stranicu. Isto tako se može navigacija odvijati i na web aplikaciji. Zbog toga se o navigaciji može razmišljati kao o hrpi stranica, korisnik uvijek vidi stranicu na vrhu hrpe, dok

su druge stranice prisutne ali su sakrivene. On upravlja ovom hrpom stranica koristeći klasu *StackController*. Programeri ne upravljaju direktno tom klasom, već se koristi u pozadini. Kada korisnik pristupa novoj stranici, *StackController* ju stavlja na vrh hrpe, dok ako se korisnik vraća, koristeći na primjer tipku „nazad“, *StackController* skida stranicu sa hrpe. Opisani način rada Ionic i Angular navigacije se može vidjeti na slici 3.2. [7]



Slika 3.2. Način rada Ionic i Angular navigacije

3.4.2. Ionic kartice

U ovoj aplikaciji se koriste kartice za prikaz liste kriptovalute i za prikaz novosti o kriptovalutama. Odnosno aplikacija ima dvije kartice gdje svaka ima svoj sadržaj. Kartice se često koriste u mobilnim aplikacijama, mogu se također vidjeti i na web aplikacijama ali se uglavnom koriste na mobilnim aplikacijama. Kartice se nalaze na dnu aplikacije i omogućuju

korisniku učitavanje različitih stranica ovisno o odabranoj kartici. Svaka kartica u Ionicu ima različitu hrpu stranica za navigaciju. Ako korisnik na prvoj kartici ode na jednu podstranicu, zatim ode na drugu stranicu i tamo ode na jednu podstranicu, te zatim na još jednu, imat će dvije hrpe stranica, jednu sa dvije stranice na prvoj kartici, i drugu sa tri stranice na drugoj kartici. Zatim ako ode sa druge kartice na prvu, vratit će se na vrh hrpe stranica prve kartice. To je važno kako se korisnik ne bi vratio na prvu stranicu kartice, što bi bilo čudno korisničko iskustvo. [7]

3.4.3. Cordova

Cordova omotava izvorni kod aplikacije (Swift, Objective-C, Java) u JavaScript sučelje koje djeluje kao most do koda Ionic aplikacije. Putem ovog mosta se mogu pozvati dodatci direktno iz JavaScript koda, bez doticanja jezika za izvorni kod. Ionic aplikacije se izvode samo unutar web prikaza, te je potrebno imati pristup značajkama uređaja kao kamera ili žiroskop. S Ionic se mogu direktno koristiti Cordova dodatci ili, kao što je preporučeno, može se koristiti paket pod nazivom *Ionic Native*. Ovaj paket omogućuje jednostavnije korištenje dodataka s Angular-om. *Ionic Native* paket nudi mnogo dodataka, ali u ovom projektu je bilo potrebno dodati još jedan dodatak u paket. Aplikacija nudi mogućnost pregleda najnovijih vijesti o kriptovalutama. Kada se otvori određena vijest, korisniku se otvara pretraživač u aplikaciji, koji prikazuje stranicu sa vijestima. Dodatak je uveden zbog toga što je otvaranje posebne aplikacije pretraživača za prikaz vijesti loše korisničko iskustvo. Dodatak se naziva *InAppBrowser*, i kako bi se dodao u aplikaciju potrebno je prvo instalirati Cordova dodatak, koji će zatim biti dodan u izvorni iOS ili Android projekt kada ih izgradimo. Drugo što je potrebno uraditi jest instalirati dodatak koji je potrebno dodati u *Ionic Native* paket. Na slici 3.3. se može vidjeti način instalacije Cordova dodatka, te *InAppBrowser* dodatka.

```
ionic cordova plugin add cordova-plugin-inappbrowser
npm install @ionic-native/inappbrowser
```

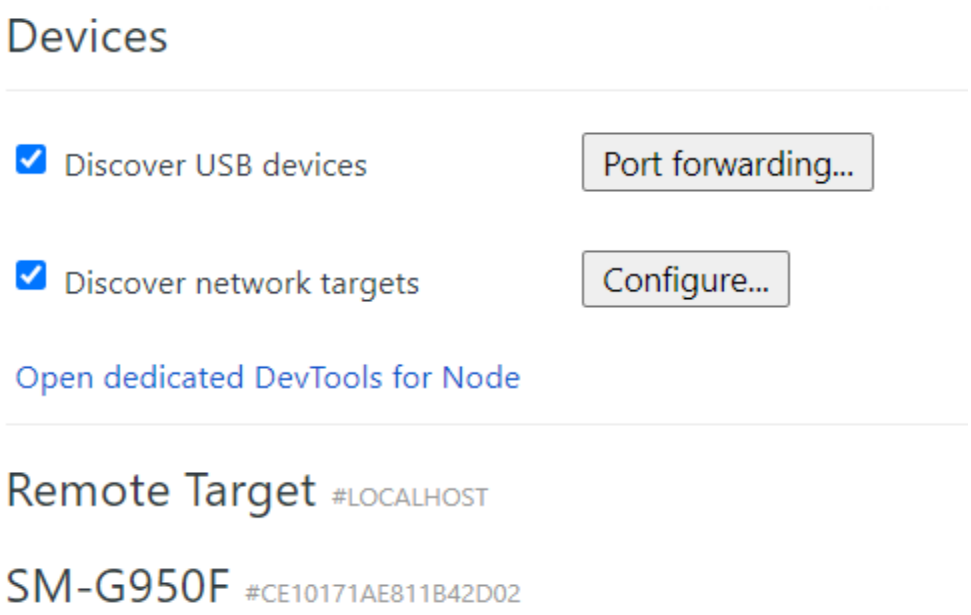
Slika 3.3. Naredbe za instalaciju *InAppBrowser* dodatka

Kako bi se dodatak koristio potrebno ga je dodati u *module* datoteku stranice koja će ga pozivati, u ovoj aplikaciji je to stranica *news*. Cordova dodatci se ne mogu testirati u pregledniku. Kako bi se testirali potrebno je instalirati aplikaciju na simulator ili fizički uređaj. Cordova most se

ubrizgava tek kada se pozove naredba *build* za aplikaciju, a datoteka nije dostupna dok se testira unutar preglednika. Ako dođe do greške prilikom korištenja ili instaliranja dodataka moguće je pozvati naredbu *ionic repair –cordova* za oporavljanje. [8]

3.4.4. Otklanjanje pogrešaka na Android i iOS uređaju

Za otklanjanje pogrešaka u Ionic aplikaciji može se koristiti daljinsko otklanjanje pogrešaka s chrome dev alatima na Androidu ili Safari Web Inspector na iOS-u za otklanjanje pogrešaka u JavaScript-u, HTML-u i CSS-u. Otklanjanje pogrešaka se može koristiti sa udaljenim uređajem ili simulatorom. Ako se koristi fizički uređaj potrebno je provjeriti je li omogućeno uklanjanje pogrešaka i na uređaju i na pregledniku. U Androidu je potrebno prvo otići u postavke i omogućiti razvojni način, a zatim USB otklanjanje pogrešaka u izborniku *Opcije za programere*. Taj izbornik se prvo mora omogućiti, a to je moguće tako što se u postavkama ode na opciju *O telefonu* zatim *Podatci o programskoj podršci*, zatim je potrebno pronaći *build* broj i pritiskati ga sve dok se ne dobije potvrda kako su omogućene razvojne opcije. Tada će se izbornik *Opcije za programere* omogućiti na istoj razini kao i opcija *O telefonu*. Kako bi se omogućilo daljinsko otklanjanje pogrešaka za android potrebno je otvoriti novu karticu u Chrome pregledniku i upisati adresu *chrome://inspect* koja će pokazati sve web prikaze dostupne za ispravljanje pogrešaka. Na slici 3.4. se može vidjeti izgled *chrome://inspect* adrese kada je priključen android uređaj.



Slika 3.4. Izgled *chrome://inspect* adrese kada je priključen Android uređaj

iOS je vrlo sličan. Za otklanjanje pogrešaka potrebno je otvoriti Safari te pronaći izbornik na kojem piše *Razvoj*. Zatim je potrebno pronaći uređaj na kojem se otklanjaju pogreške. Ako Safari ne prikazuje izbornik *Razvoj*, potrebno je otići u Safari postavke i omogućiti napredni način rada iz naprednih postavki. Ako se koristi Cordova ili Capacitor sa Ionic aplikacijom, neke od izvornih mogućnosti se neće moći testirati u pregledniku, nego samo na fizičkom uređaju ili simulatoru. [9]

3.5. Angular okvir 10.0.14

U ovom projektu je uz Ionic korišten Angular okvir. Za programiranje web aplikacija sa Ionicom može se koristiti čisti JavaScript, ali Angular uvelike olakšava razvoj. Osim Angular okvira mogu se koristiti i drugi okviri kao Vue.js ili React. Kada se grade složenije aplikacije gdje postoji složena logika, mnogo se vremena izgubi na razmišljanje o sitnim detaljima poput ispravnog ažuriranja DOM-a, ispravnog upravljanja stanjem aplikacije itd. Takve stvari su teške i dosadne programerima i žele se usredotočiti na temeljnu logiku koja pokreće aplikaciju. Dakle, sve te stvari su relativno teške kako bi ih programer sam implementirao pa se koristi okvir poput Angular-a. On daje jasno definirana pravila o tome kako se gradi aplikacija s komponentama i

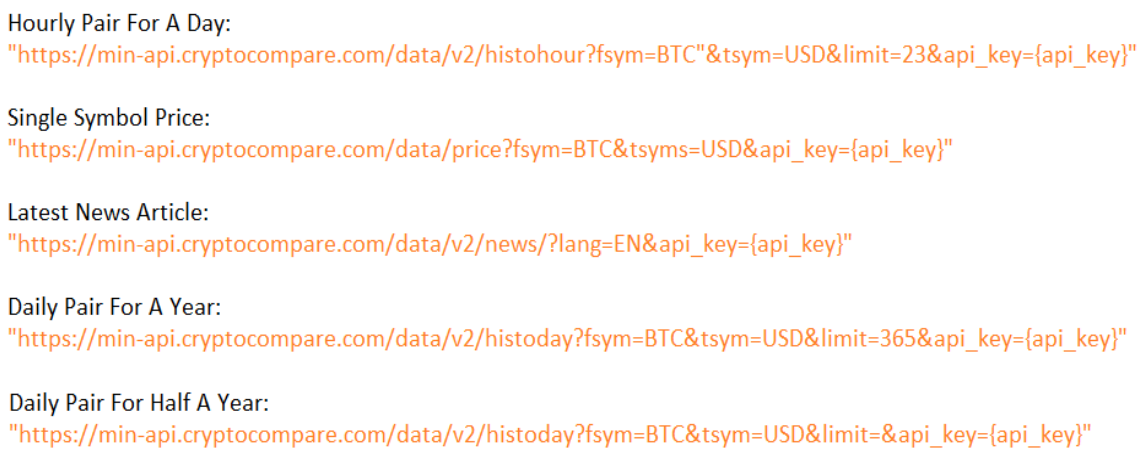
direktivama i brine se o iscrtavanju DOM-a kako bi se programeri mogli usredotočiti na logiku koja mijenja stanje aplikacije, stanje podataka, ažurira DOM itd. Daje programerima servise i druge načine prijenosa podataka od točke A do točke B, od komponente do komponente, te također pomaže pri navigaciji i slanju podataka prilikom promjene stranice. To je jedan od razloga zašto se koristi Angular sa Ionicom, umjesto samog Ionica. [10]

3.6. CryptoCompare API

API (Application Programming Interface) znači sučelje aplikacijskog programiranja, što je skup definicija i protokola za izgradnju i integraciju aplikacijske programske podrške. API-ji omogućuju proizvodu ili usluzi komuniciranje s drugim proizvodima i uslugama, dok im nisu poznate njihove implementacije. To može pojednostaviti razvoj aplikacija, uštedjeti vrijeme i novac. U ovom projektu se koristi CryptoCompare API jer je jako jednostavan za korištenje, uspostavljanje i daje sve podatke potrebne za aplikaciju. Korištenje CryptoCompare API-ja se plaća, ali je moguće koristiti besplatnu verziju koja omogućava samo 250 tisuća poziva, što je dovoljno za izgradnju i testiranje manje aplikacije kao što je aplikacija u ovom radu. Ima ogroman niz podataka o tržištu digitalne imovine, upakiranih u 73+ robusnih API krajnjih točaka. Nudi podatke o:

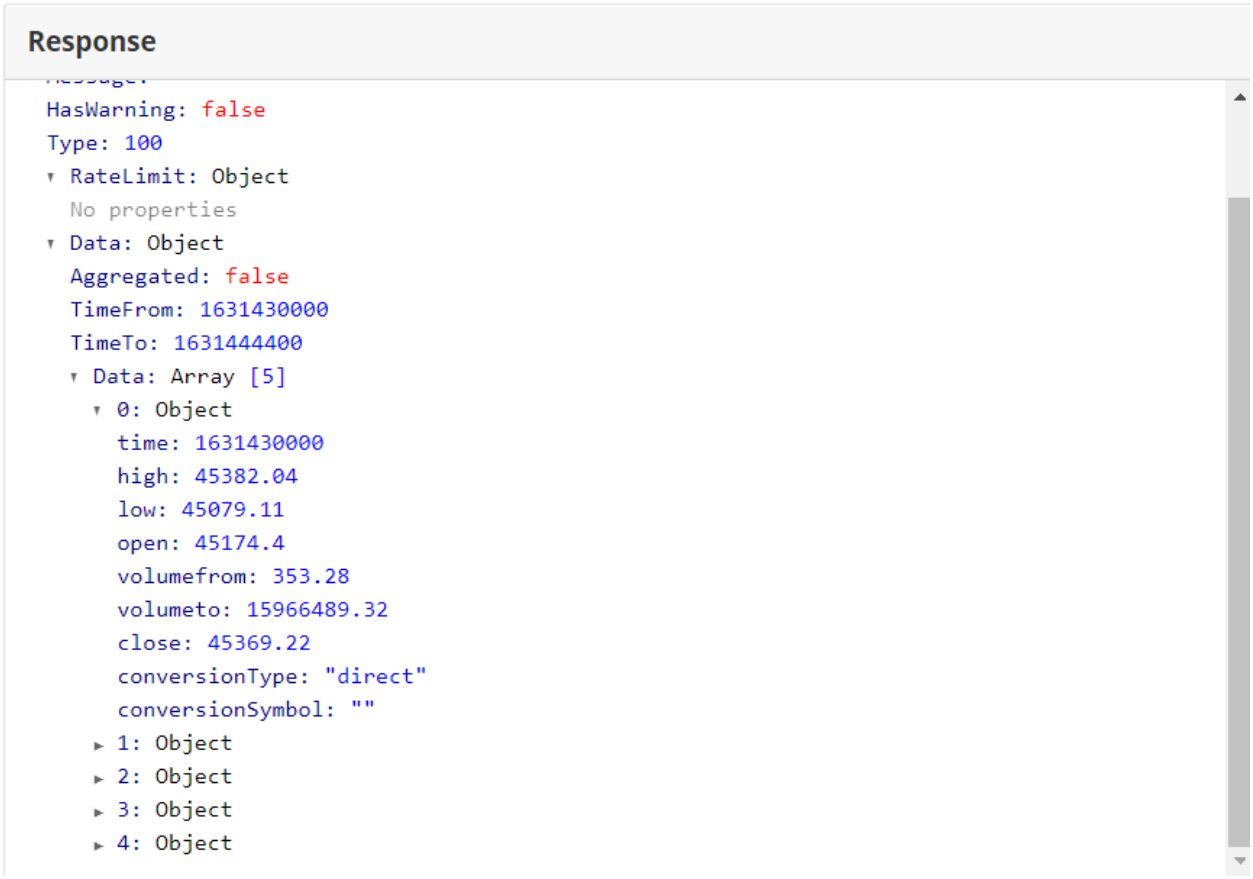
- Trenutnoj cijeni jedne ili više kriptovaluta
- OHLCV (Open, High, Low, Close, Volume) jednog para, te se mogu birati iznosi po danu, satu ili minuti.
- Podatci o blockchainu
- Najnoviji podatci o signalima za trgovanje koje uzima od tvrtke IntoTheBlock koja koristi strojno učenje i naprednu statistiku kako bi izvukla signale za trgovanje valutama.
- Top listama parova za najveći obujam trgovanja u zadnja 24 sata, najveći broj transakcija u zadnja 24 sata, zatim podatke o top listama najboljih parova za međusobno trgovanje itd.
- Društvenoj statistici određene valute, odnosno koliko je trenutno popularna, te popularnost kroz protekle dane ili sate.
- Najnovijim vijestima u svijetu kriptovaluta

U ovom projektu se poziva OHLCV određenog para kriptovaluta po satu kako bi se mogao prikazati njihov međusobni odnos na grafu. Zatim se poziva krajnja točka koja vraća trenutnu vrijednost određene valute. Ona se poziva kako bi se prikazala trenutna vrijednost svake valute na listi u aplikaciji. Osim toga još se poziva krajnja točka koja vraća najnovije vijesti iz svijeta kriptovaluta, te vrijednosti cijena kroz godinu ili pola godine ovisno o izboru korisnika. Korisnik u aplikaciji ima mogućnost odabira valute kako bi vidio vrijednost u dolarima koju bi imao ako je uložio tisuću dolara prije godinu ili pola godine, ovisno o izboru. Na slici 3.5. se može vidjeti kako izgledaju API pozivi za krajnje točke koje se koriste u ovoj aplikaciji. Za primjer u svakom API pozivu je uzet Bitcoin čija se skraćenica BTC nalazi u pozivu. [11]



Slika 3.5. Pozivi API krajnjih točaka aplikacije

Kada se pozove API poziv, rezultat se vraća aplikaciji u JSON obliku. Na slici 3.6. se može vidjeti primjer dohvaćanja *OHLCV Hourly Pair* Bitcoina u odnosu na američki dolar, tj. njihov međusobni odnos cijena kroz 5 sati. Zbog prevelikog rezultata prikazane su vrijednosti samo jednog sata od pet. Vrijeme je prikazano u milisekundama, odnosno predstavlja *unix timestamp*. To je vrijeme koje je proteklo od 01.01.1970. godine. Osim vremena rezultat vraća za određeni sat najveću vrijednost (High), najmanju vrijednost (Low), vrijednost koju je valuta imala u početku tog sata (Open), vrijednost koju je valuta imala na kraju tog sata (Close).



Slika 3.6. Rezultat API poziva *OHLCV Hourly Pair* za 5 sati

Kako bi se dobio rezultat na slici 3.6. potrebno je pozvati API poziv koji izgleda kao na slici 3.7.:

„https://min-api.cryptocompare.com/data/v2/histohour?fsym=BTC&tsym=USD&limit=5&api_key={api_key}“.

Slika 3.7. API poziv za *OHLCV Hourly Pair* za 5 sati

Na mjestu poziva u slici 3.7. gdje piše *{api_key}* potrebno je ubaciti API ključ koji je generiran kada korisnik napravi računa na CryptoCompare stranici.

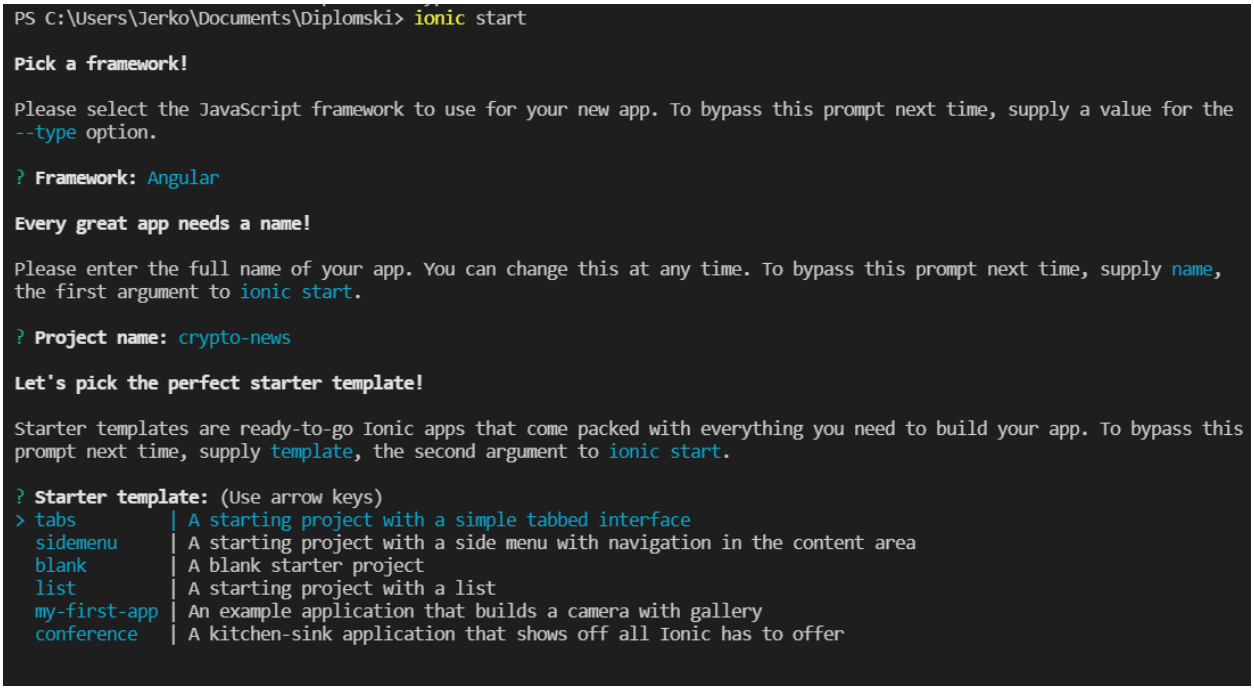
4. PROCES RAZVOJA APLIKACIJE

Prvo što je potrebno uraditi prije razvoja aplikacije jest instalirati Ionic. Kako bi se instalirao potrebno je imati Node.js instaliran kako bi se mogao koristiti npm. To je standardni paket za upravljanje ovisnostima o Javascriptu, ne samo u NodeJS projektima, nego u bilo kojem projektu koji koristi JavaScript. Bez obzira radi li se o front-end ili back-end projetku, pa i ovaj projekt također koristi npm za upravljanje svojim ovisnostima i ovisnostima poput Angular i Ionic okvira. Nakon instalacija Node.js moguće je instalirati i Ionic. To se radi tako što se pokrene naredba prikazana na slici 4.1..

```
npm install -g ionic
```

Slika 4.1. Naredba za instalaciju Ionic okvira

Na Linuxu i Mac-u je potrebno dodati *sudo* ispred naredbe, dok se na Windowsu to može učiniti tako što se konzola pokrene u administratorskom načinu. Kada se instalira moguće je napraviti novi projekt. Novi projekt se pravi pokretanjem naredbe *ionic start*. Potrebno je otići u direktorij u kojem se želi napraviti projekt te tamo pokrenuti naredbu. Zatim konzola čeka dok se odabere okvir koji želimo koristit. Nudi tri opcije i to su Angular, React i Vue. Nakon toga konzola traži upis naziva projekta. U ovom projektu naziv je *crypto-news*. Kada je naziv upisan, konzola traži izbor početnog predloška. Nudi opcije *tabs, sidemenu, blank, list, my-first-app, conference*. U ovom projektu odabrana je opcija *blank*. Na slici 4.2. se može vidjeti postavljanje projekta u konzoli.



Slika 4.2. Postavljanje projekta u konzoli

4.1. Ionic stranice

Ionic stranice se koriste kako bi se prikazao jedan dio aplikacije. Stranica upravlja registracijom i prikazom određenih stranica na temelju URL-ova. Za razliku od tradicionalnih web aplikacija, URL-ovi ne diktiraju navigaciju u Ionic aplikacijama. Umjesto toga, URL-ovi pomažu u povezivanju određenih dijelova aplikacije. Za kretanje kroz aplikaciju koristi se NavController koji koristi *push* i *pop* za hodanje po stranicama. U ovoj aplikaciji početna stranica se naziva *home* i u njoj se nalaze dvije kartice, jedna za prikaz liste i prikaz podataka o kriptovaluti, te druga za prikaz najnovijih vijesti. Osim *home* stranice u aplikaciji se koriste još tri stranice. Jedna je za prikaz vijesti u kartici i naziva se *news*. Jedna od stranica je za prikaz liste kriptovaluta i njihovih trenutnih vrijednosti i naziva se *currencies*, dok se u *currencies* stranici nalazi druga stranica koja prikazuje detalje o valuti i naziva se *currency-detail*. Na slici 4.3. se može vidjeti HTML dio *home* stranice. Na njoj se može vidjeti kako su dodane kartice u aplikaciju, nazivi kartice, te njihove ikone.

```
<ion-tabs>
  <ion-tab-bar slot="bottom">
    <ion-tab-button tab="currencies">
      <ion-label>Cryptocurrencies</ion-label>
      <ion-icon name="logo-bitcoin"></ion-icon>
    </ion-tab-button>
    <ion-tab-button tab="news">
      <ion-label>News</ion-label>
      <ion-icon name="newspaper-outline"></ion-icon>
    </ion-tab-button>
  </ion-tab-bar>
</ion-tabs>
```

Slika 4.3. HTML dio *home* stranice

4.2. Navigacija u aplikaciji

Navigacija u Ionicu koristi rute, koje preusmjeravaju aplikaciju na određenu stranicu. U ovoj aplikaciji postoje dvije kartice i potrebno je napraviti navigaciju koja će voditi korisnika kroz te dvije kartice. Na slici 4.4. se može vidjeti kako je uspostavljenja navigacija, odnosno rute, u *home* stranici, koja upravlja karticama.

```
const routes: Routes = [
  {
    path: 'tabs',
    component: HomePage,
    children: [
      {
        path: 'currencies',
        loadChildren: () => import('./currencies/currencies.module').then( m => m.CurrenciesPageModule)
      },
      {
        path: 'news',
        loadChildren: () => import('./news/news.module').then( m => m.NewsPageModule)
      },
      {
        path: '',
        redirectTo: '/home/tabs/currencies',
        pathMatch: 'full'
      }
    ]
  },
  {
    path: '',
    redirectTo: '/home/tabs/currencies',
    pathMatch: 'full'
  }
];
```

Slika 4.4. Rute *home* stranice

Kao što se vidi na slici 4.1. nalazi se glavna ruta *tabs* koja se koristi za grupiranje ruta za kartice. Ona sadrži u sebi tri druge rute, jedna za *currencies* stranicu koja izgleda kao */home/tabs/currencies*, jedna za *news* stranicu koja izgleda kao */home/tabs/news*, te jedna prazna ruta koja će preusmjeriti korisnika na *currencies* stranicu. Često se može dogoditi pristup praznoj ruti umjesto pristupa *currencies* ili *news* te se zbog toga mora preusmjeriti korisnika kako ne bi imao loše iskustvo sa korištenjem aplikacije. Ako korisnik ode na rutu za *currencies* stranicu, učitat će se njen modul koji se nalazi u projektu u datoteci *./currencies/currencies.module*. Isto tako ako korisnik ode na rutu za *news* stranicu, učitat će se modul stranice koji se nalazi u projektu u datoteci *./news/news.module*. Odlaskom na praznu putanju, korisnik je preusmjeren na *currencies* stranicu te se tada isto tako učitava njen modul. Kada korisnik uđe na stranicu može vidjeti listu kriptovaluta sa njihovim trenutnim vrijednostima u dolarima. Korisnik zatim može odabrati bilo koju valutu te će se učitati nova stranica koja se naziva *currency-detail* koja pokazuje detalje o toj valuti. Za tu stranicu je također potrebno namjestiti navigaciju. U *rounting* datoteci *currencies* stranice nalazi se konfiguracija za navigaciju između tih dviju stranica. Na slici 4.5. se mogu vidjeti rute *currencies* stranice.

```
const routes: Routes = [
  {
    path: '',
    component: CurrenciesPage
  },
  {
    path: ':currencyName',
    loadChildren: () => import('./currency-detail/currency-detail.module').then( m => m.CurrencyDetailPageModule)
  }
];
```

Slika 4.5. Rute *currencies* stranice

Kao što se vidi na slici 4.5., u rutama *currencies* stranice ima jedna ruta naziva *:currencyName*. Dvotočka ispred naziva rute predstavlja dinamičku rutu čije će ime biti ovisno o odabranoj kriptovaluti. Ako korisnik odabere Bitcoin sa liste, otvoriti će se *currency-detail* stranica sa rutom */home/tabs/currencies/BTC*. Umjestno *:currencyName*, u putanju će se postaviti skraćeni naziv odabrane valute, a u ovom primjeru je to BTC. Nakon toga će se učitati *currency-detail* modul koji se nalazi u projektu u datoteci *./currency-detail/currency-detail.module*.

4.3. Pozivanje API-ja

Za pozivanje API-ja napravljen je novi servis koji se naziva *data*. Servisi su dio Angular-a i to su singleton objekti koji se pokreću samo jednom tijekom života aplikacije. Sadrže metode koje održavaju podatke tijekom cijelog vijeka trajanja aplikacije, tj. podatci se ne osvježavaju i dostupni su cijelo vrijeme. Glavni cilj usluge je organiziranje i dijeljenje poslovne logike, modela ili podataka i funkcija s različitim komponentama aplikacije. Jedan primjer korištenja servisa bi bio prijenos podataka s jednog dijela aplikacije na drugi. *Data* servis ima jednu privatnu varijablu *api_key* koju koristi u svakom API pozivu. CryptoCompare zahtjeva od programera unos API ključa prilikom poziva krajnjih točaka, te je zbog toga dodana varijabla *api_key*. Kako bi Ionic koristio API potrebno je servisu dodati *HttpClient* koji programeru nudi korištenje REST metoda. REST metode su GET, POST, PUT i DELETE. U ovom projektu se koristi samo GET, jer se samo dohvaćaju podatci sa CryptoCompare-a. Na slici 4.6. je prikazan izgled *data* servisa, te njegove privatne varijable, konstruktora i metoda koje se koriste kroz cijelu aplikaciju.

```
export class DataService {
  private api_key: string = "e50d12f2b7bc6d157cbf53701d72a3e0842034251f040d2dde0a5cfaeb90deb4";

  constructor(private http: HttpClient) {}

  getHourlyPair(currency: string) {
    return this.http.get("https://min-api.cryptocompare.com/data/v2/histohour?fsym=" + currency + "&tsym=USD&limit=23&api_key={" + this.api_key + "}");
  }

  getSingeSymbolPrice(currency: string) {
    return this.http.get("https://min-api.cryptocompare.com/data/price?fsym=" + currency + "&tsyms=USD&api_key={" + this.api_key + "}");
  }

  getLatestNewsArticles() {
    return this.http.get("https://min-api.cryptocompare.com/data/v2/news/?lang=EN&api_key={" + this.api_key + "}");
  }

  getDailyPairForYear(currency: string) {
    return this.http.get("https://min-api.cryptocompare.com/data/v2/histoday?fsym=" + currency + "&tsym=USD&limit=365&api_key={" + this.api_key + "}");
  }

  getDailyPairForHalfYear(currency: string) {
    return this.http.get("https://min-api.cryptocompare.com/data/v2/histoday?fsym=" + currency + "&tsym=USD&limit=183&api_key={" + this.api_key + "}");
  }
```

Slika 4.6. Izgled *data* servisa

U neke od poziva API-ja potrebno je staviti skraćenicu kriptovalute za koju želimo dohvatiti podatke. Zbog toga metode getHourlyPair, getSingleSymbolPrice, getDailyPairForYear i getDailyPairForHalfYear imaju parametar *currency* koji predstavlja skraćenicu odabrane kriptovalute. Također se u poziv ubacuje privatna varijabla *api_key*. Vrijednosti svih cijena kroz

aplikaciju su prikazane u američkim dolarima te zbog toga nikada ne treba mijenjati varijablu *tsym* u API pozivu i uvijek je postavljena na USD,

4.4. Prikaz liste kriptovaluta

Za prikaz liste kriptovaluta koristi se *currencies* stranica. Lista se zapravo sastoji od Currency modela, koji u sebi sadrži četiri varijable. Varijablu *name* koja predstavlja ime, varijablu *value* koja predstavlja vrijednost, varijablu *iconPath* koja predstavlja putanju do ikone, te varijablu *abbr* koja predstavlja skraćenicu. Na slici 4.7. se može vidjeti Currency model koji se koristi za listu.

```
export class Currency {
    constructor(
        public name: string,
        public value: number,
        public iconPath: string,
        public abbr: string
        ) {}
}
```

Slika 4.7. Model za listu kriptovaluta

Lista kriptovaluta se nalazi u servisu *currencies* stranice. Dakle servis *currencies* stranice daje listu koja se sastoji od Currency modela, odnosno objekta. Jedan dio liste može se vidjeti na slici 4.8.. Vrijednosti u dolarima se dohvaćaju pomoću API poziva. U glavnom dijelu projektnog direktorija se nalazi direktorij *assets* u koji se spremaju slike korištene za aplikaciju. U tom direktoriju se nalazi drugi direktorij pod nazivom *icon*, te su u njega spremljene sve ikone prikazanih valuta, te i ikona Bitcoina koja se koristi kao favicon. Ikone su preuzete sa stranice *cryptoicons.co* gdje su dostupne za besplatno preuzimanje.

```
export class CurrenciesService {
  private _currencies: Currency[] = [
    new Currency("Bitcoin", 0.00, "\\assets\\icon\\btc.svg", "BTC"),
    new Currency("Ethereum", 0.00, "\\assets\\icon\\eth.svg", "ETH"),
    new Currency("XRP", 0.00, "\\assets\\icon\\xrp.svg", "XRP"),
    new Currency("Stellar", 0.00, "\\assets\\icon\\xlm.svg", "XLM"),
    new Currency("Cardano", 0.00, "\\assets\\icon\\ada.svg", "ADA"),

    new Currency("Dogecoin", 0.00, "\\assets\\icon\\doge.svg", "DOGE"),
    new Currency("Polkadot", 0.00, "\\assets\\icon\\dot.svg", "DOT"),
    new Currency("Chainlink", 0.00, "\\assets\\icon\\link.svg", "LINK"),
    new Currency("Tether", 0.00, "\\assets\\icon\\usdt.svg", "USDT"),
    new Currency("Litecoin", 0.00, "\\assets\\icon\\ltc.svg", "LTC"),

    new Currency("Binance Coin", 0.00, "\\assets\\icon\\bnb.svg", "BNB"),
    new Currency("BitTorrent", 0.00, "\\assets\\icon\\btt.svg", "BTT"),
    new Currency("Solana", 0.00, "\\assets\\icon\\sol.svg", "SOL"),
    new Currency("Polygon", 0.00, "\\assets\\icon\\matic.svg", "MATIC"),
```

Slika 4.8. Lista kriptovaluta u servisu

Servis također ima jednu metodu koja vraća sve kriptovalute sa liste, stranica zatim dohvaća listu pozivajući tu metodu i prikazuje ju korisniku. Metoda koja vraća listu kriptovaluta se može vidjeti na slici 4.9.

```
get currencies() {
  return [...this._currencies];
}
```

Slika 4.9. Izgled metode koja vraća listu kriptovaluta

Metoda sa slike iznad se poziva u glavnoj klasi za *currencies* stranicu koja se zove *CurrenciesPage*. Ona implementira *OnInit* koji zahtjeva implementaciju metode ngOnInit() koja se poziva prije prikaza stranice korisniku. To je korisno jer se u toj metodi mogu pripremiti podatci. Također koristi *data* servis i *currencies* servis. Iz *currencies* servisa dohvaća listu kriptovaluta te zatim ažurira vrijednost svake valute u listi. A ažuriranje vrši tako što poziva metodu getSingleSymbolPrice iz *data* servisa koja poziva API. Svaka kriptovaluta u listi ima spremljenu svoju skraćenicu u varijablu *abbr*, te se zbog toga ta varijabla predaje metodi getSingleSymbolPrice kako bi API znao od koje kriptovalute traži podatke. Na slici 4.10. se može vidjeti izgled *CurrenciesPage* klase, te metode ngOnInit().

```
export class CurrenciesPage implements OnInit {
  loadedCurrencies: Currency[];
  currencyValue: any;

  constructor(private dataService: DataService, private currenciesService: CurrenciesService) { }

  ngOnInit() {
    this.loadedCurrencies = this.currenciesService.currencies;
    for(let currency of this.loadedCurrencies) {
      this.dataService.getSingeSymbolPrice(currency.abbr)
        .subscribe(result => {
          this.currencyValue = result;
          currency.value = this.currencyValue.USD;
        })
    }
  }
```

Slika 4.10. Izgled CurrenciesPage klase

Varijabla *loadedCurrencies* sa slike iznad jednaka je listi kriptovaluta sa servisa, te ima ažurirane vrijednosti cijene. Zbog toga se ona koristi u HTML datoteci *currencies* stranice za prikaz liste. Stranica ima naslov na kojemu piše *Cryptocurrencies* te naravno listu kriptovaluta. Za prolazak kroz list koristi se **ngFor* od Angular-a što uvelike olakšava razumijevanje i čitljivost koda. Za prikaz liste koristi se <ion-list> u koji se postavlja <ion-item> koji predstavlja svaki red liste. Ionic uvelike olakšava prikaz liste na aplikaciji. U <ion-item> je postavljen routerDirection na *forward* što govori Ionicu kako se nova stranica dodaje na hrpu stranica, zatim na *routerLink* postavljena je ruta na koju aplikacija ide nakon odabire bilo kojeg reda na listi. *QueryParam* predstavlja vrijednost koja će se predati ruti na mjestu *:currencyName* koje je spomenuto u poglavlju 4.2. Navigacija u aplikaciji. Ime kriptovalute i vrijednost cijene se uzimaju za svaki red iz odgovarajućeg reda liste. Odnosno Angular prolazi kroz listu, i za svaki red pravi novi <ion-item> te postavlja naziv i cijenu. Na slici 4.11. se može vidjeti HTML kod *currencies* stranice.

```
<ion-header [translucent]="true">
  <ion-toolbar>
    <ion-title>Cryptocurrencies</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <ion-list>
    <ion-item
    *ngFor="let currency of loadedCurrencies"
    routerDirection="forward"
    [routerLink]="['/', 'home', 'tabs', 'currencies', currency.name]"
    [queryParams]="{abbr : currency.abbr}"
    detail>
      <ion-icon slot="start" [src]="currency.iconPath"></ion-icon>
      <ion-label>
        {{ currency.name }}
      </ion-label>
      <ion-label class="ion-text-right">
        ${{ currency.value }}
      </ion-label>
    </ion-item>
  </ion-list>
</ion-content>
```

Slika 4.11. HTML *currencies* stranice

4.5. Prikaz detalja kriptovalute

Za prikaz detalja kriptovalute koristi se *currency-detail* stranica. Prikazuje graf odnosa vrijednosti cijene i vremena, te vrijednost koju bi korisnik imao u dolarima ako je uložio tisuću dolara prije godinu dana. Glavni dio *currency-detail* stranice se odvija u *CurrencyDetailPage* klasi. Klasa također implementira *OnInit* koji joj omogućuje poziv metode ngOnInit. Klasa koristi *data* servis kako bi koristila metode koje pozivaju API, te *ActivatedRoute* kako bi dohvatila informaciju o valuti koja je predana ruti prilikom odabira sa liste na stranici *currencies*. Dohvaćanje informacije o predanoj valuti se odvija u ngOnInit metodi. Osim toga poziva se i getHourlyPair metoda iz *data* servisa, ažuriraju se podatci u grafu, te se poziva metoda getDailyPairForYear iz koje se dohvaćaju podatci o vrijednosti prije godinu dana, te nakon toga računa iznos koji bi korisnik danas imao ako je uložio tisuću dolara prije godinu dana. Na slici 4.12. se može vidjeti računanje iznosa koji korisnik ima nakon ulaganja.

```
this.dataService.getDailyPairForYear(this.currencyAbbr)
.subscribe(result => {
  this.cryptos = result;
  let dailyPair = this.cryptos.Data.Data;

  let yearAgoValue = dailyPair[0];
  let currencyAmount = this.depositMoney / yearAgoValue.close;
  let todayValue = dailyPair[364];
  this.depositToday = todayValue.close * currencyAmount;
  this.depositToday = Math.round(this.depositToday * 1e4) / 1e4;
});
```

Slika 4.12. Izgled metode koja računa iznos koji bi korisnik imao ako je uložio tisuću dolara u kriptovalutu prije godinu dana

Korisnik ima mogućnost birati želi li računati iznos koji bi imao ako je uložio novac prije godinu dana ili prije pola godine. Metoda koja računa iznos koji bi korisnik imao ako je uložio novac prije pola godine je slična metodi na slici 4.12.. Korisniku je omogućen izbor između cijele godine i pola godine pomoću <ion-radio-group> koji pruža Ionic, za jednostavan izbor između dvije opcije. Na slici 4.13. se može vidjeti HTML kod *currency-detail* stranice koji omogućuje korisniku izbor između pola godine i čitave godine.

```
<ion-text color="primary">
  If you put a deposit of <b>1000$</b> into <b>{{ currencyAbbr }}</b> a:
</ion-text>
<ion-list style="justify-content: center;display: flex;align-items: center;">
  <ion-radio-group value="year" (ionChange)="radioGroupChange($event)">
    <ion-item>
      <ion-label>Year ago</ion-label>
      <ion-radio slot="start" value="year"></ion-radio>
    </ion-item>

    <ion-item>
      <ion-label>Half a year ago</ion-label>
      <ion-radio slot="start" value="halfYear"></ion-radio>
    </ion-item>
  </ion-radio-group>
</ion-list>
<ion-text color="primary">
  you would have: <b>{{ depositToday }}$</b> today.
</ion-text>
```

Slika 4.13. HTML kod *currency-detail* stranice

4.5.1. ApexCharts

ApexCharts je moderna biblioteka grafikona koja pomaže programerima u stvaranju lijepih i interaktivnih vizualizacija za web stranice. Kako bi se mogla koristiti sa Angular-om potrebno ju je prvo instalirati, a to se može učiniti tako što se pokrene naredba *npm install apexcharts ng-apexcharts –save*. Nakon toga potrebno je u angular.json datoteku unutar Ionic projekta, ispod *scripts* dijela dodati *node_modules/apexcharts/dist/apexcharts.min.js*. Nakon toga ApexCharts biblioteka je instalirana i može se koristiti u Ionic aplikaciji. Graf se koristi u *currency-detail* stranici pa je potrebno u njene module dodati *NgApexchartsModule*. Na slici 4.14. se mogu vidjeti opcije koje su predane grafu svijeća (engl. Candle graph). Pod *data* dio se predaju OHLCV vrijednosti preuzete sa CryptoCompare API-ja, ali ih ima previse kako bi stali na sliku. [12]

```
this.chartOptions = {
  series: [
    {
      name: "candle",
      data: [ ···
      ]
    }
  ],
  chart: {
    height: 350,
    type: "candlestick"
  },
  title: {
    text: "" + this.currencyAbbr + " chart",
    align: "left"
  },
  xaxis: {
    type: "datetime"
  },
  yaxis: {
    tooltip: {
      enabled: true
    }
  }
```

Slika 4.14. Opcije predane grafu svijeća (engl. candle graph)

4.6. Prikaz novosti iz svijeta kriptovaluta

Jedna od dvije kartice dostupne u aplikaciji omogućuje korisniku listu najnovijih vijesti iz svijeta kriptovaluta. Kako bi se prikazale vijesti potrebno ih je dohvatiti iz CryptoComapre API-ja. Zbog toga se koristi *data* servis koji ima metodu getLatestNewsArticles(). Glavni dio ove stranice odvija se u klasi *NewsPage*. Ona također implementira *OnInit* kako bi se dobila metoda ngOnInit. U metodi ngOnInit se poziva metoda getLatestNewsArticles kako bi se dohvatile vijesti prije učitavanja stranice. Zatim se uzima 20 vijesti iz dohvaćenih vijesti i stavlja u listu koja će se prikazati na stranici. Dohvaćene vijesti su u JSON obliku, te je vrijeme kada su objavljene dohvaćeno u milisekundama. Te milisekunde je potrebno pretvoriti u format koji je lako čitljiv korisniku, odnosno u mjesece i dane u tjednu. Zbog toga se koristi metoda timestampToDate() koja pretvara vrijeme iz milisekundi u mjesece i dane u tjednu. Također se u njoj koriste dva niza iz *NewsPage* klase koja se zovu *weekday* i *month*. Kako bi korisnik mogao čitati vijesti u aplikaciji i kako ne bi morao otvarati novu aplikaciju za učitavanje vijesti koristi se *InAppBrowser*. Zbog toga je napravljena metoda goToLink koja koristi taj *InAppBrowser* kako bi otvorio vijest u aplikaciji. Na slici 4.15. se mogu vidjeti metode *NewsPage* klase.

```
export class NewsPage implements OnInit {
  private news: any;
  private loadedNews: any[];
  private weekday: string[] = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
  private month: string[] = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul",
    "Aug", "Sep", "Oct", "Nov", "Dec"]

  constructor(private dataService: DataService) {

   }

  ngOnInit() {
    this.loadedNews = [];
    this.dataService.getLatestNewsArticles()
      .subscribe(result => {
        this.news = result;
        for(let x = 0; x < 20; x++) {
          this.loadedNews.push(this.news.Data[x]);
        }
      })
  }

  timestampToDate(timestamp: number) {
    const date = new Date((timestamp + 7200)*1000);
    return this.weekday[date.getDay()] + ", "
    + " " + date.getDate() + " " + this.month[date.getMonth()];
  }

  goToLink(url: string) {
    InAppBrowser.create(url);
  }
```

Slika 4.15. Metode *NewsPage* klase

Za prikaz liste u HTML datoteci *news* stranice korišten je <ion-list> te u njoj jedan <ion-item> koji koristi Angular-ov **ngFor* koji prolazi kroz sve vijesti i redom ih prikazuje na stranici. Prolazi kroz svaku vijest te postavlja sliku, naslov, izvor vijesti i vrijeme. Na slici 4.16. se može vidjeti HTML kod *news* stranice.

```
<ion-header>
  <ion-toolbar>
    <ion-title>News</ion-title>
  </ion-toolbar>
</ion-header>

<ion-content padding>
  <ion-list>
    <ion-item
    *ngFor="let news of loadedNews"
    (click)="goToLink(news.guid)"
    detail>
      <ion-thumbnail slot="start">
        <ion-img [src]="news.imageurl"></ion-img>
      </ion-thumbnail>
      <ion-label style="white-space: normal;">
        <h2> {{ news.title }} </h2>
        <p> <b>{{ news.source_info.name }}</b> | {{ timestampToDate(news.published_on) }} </p>
      </ion-label>
    </ion-item>
  </ion-list>
</ion-content>
```

Slika 4.16. HTML kod *news* stranice

5. KRIPTOVALUTE

Kriptovaluta je vrsta digitalnog novca stvorenog iz koda. One funkcioniraju autonomno, izvan tradicionalnih bankovnih i državnih sustava. Koriste kriptografiju za osiguranje transakcija i reguliranje stvaranja dodatnih jedinica. Bitcoin, izvorna i daleko najpoznatija kriptovaluta, pokrenuta je u siječnju 2009. Danas ih je na internetu dostupno preko 1000. Značajno se razlikuju od tradicionalnih fiat valuta. Bez obzira na to, i dalje se mogu kupiti i prodati kao i bilo koja druga imovina. Spadaju pod digitalne valute, alternativne valute i virtualne valute. U početku su bile osmišljene kao alternativni način plaćanja za internetske transakcije. Međutim, tvrtke i potrošači ih još uvijek nisu široko prihvatili, a trenutno su previše nestabilne kako bi bile prikladne kao način plaćanja. Kao decentralizirana valuta, razvijena je kako bi bila slobodna od državnih utjecaja, a ekonomiju kriptovaluta umjesto toga prati internetski protokol *peer-to-peer*. Bitcoin je zaslužan kao prva decentralizirana kriptovaluta. Kao i sve druge, njime se upravlja putem blockchaina. Bitcoin je stvorio Satoshi Nakamoto, ali se ne zna odnosi li se ime na pojedinca ili skupinu. Značajka većine kriptovaluta je to što su osmišljene tako sa ciljem smanjivanja proizvodnje. Posljedično, samo će ograničeni broj jedinica valute ikada biti u opticaju, slično kao i zlato. Na primjer, broj Bitcoina neće premašiti 21 milijun. S druge strane, kriptovalute poput ethereuma rade malo drugačije. Izdavanje je ograničeno na 18 milijuna ethereum žetona godišnje. Ograničavanje broja valute osigurava „rijetkost“ što joj povećava vrijednost. Neki tvrde kako je tvorac bitcoina zapravo modelirao kriptovalutu na plemenitim metalima. Zbog toga rudarstvo postaje sve teže s vremenom, jer se rudarska nagrada svakih nekoliko godina prepolovljuje sve dok ne dosegne nulu. [13]

5.1. Razlike fiat i kripto valuta

Fiat je konvencionalniji oblik novca. Svoju vrijednost crpi iz središnjeg tijela, najčešće bankarskog sustava koji kontrolira vlada. Kovanice ili novčanice stvorene su od banaka. One također mogu ispisati više valuta, povezujući vrijednost valute s odlukama središnje vlade. USD, GBP, EUR, HRK i u biti svaka glavna papirna valuta povezana s nacionalnom državom je fiat valuta. Jedna od glavnih razlika između fiat i kripto valute je ta što ne postoji „središnja“ kontrola kriptovalute. Kriptovaluta umjesto toga dolazi iz blockchaina. Nove kovanice potrebno je rudariti pomoću algoritama kao što su *Proof-of-Work, Proof-of-Stake* ili *Proof-of-Authority*. Svaki novčić je povezan sa blockchainom. To znači kako ne postoji način „tiskanja“ više

kriptovaluta, pogotovo jer se radi samo o digitalnoj valuti. U većini država se i dalje koriste fiat valute jer su stabilnije i lakše je njima upravljati. [13]

5.2. Blockchain

Blockchain je temeljna tehnologija iza Bitcoina i tisuće drugih kriptovaluta te ima veliki potencijal. Blockhain je javni digitalni zapis transakcija koji bilježi informacije na način koji otežava hakiranje ili promjenu. Omogućuje siguran način na koji pojedinci mogu izravno komunicirati jedni s drugima, bez posrednika poput vlade, banke ili nekog trećeg. Rastući popis zapisa, nazvan blokovi, povezan je kriptografijom. Svaka transakcija se provjerava *peer-to-peer* povezanom mrežom računala i dodaje rastućem lancu podataka. Nakon spremanja, podatci se ne mogu mjenjati. Koristeći Bitcoin sustav kao primjer, blockchain funkcionira tako što kupovina i prodaja bitcoina ulazi i prenosi se mrežom računala, poznatih kao čvorovi. Zatim ta mreža od tisuće čvorova diljem svijeta potvrđuje transakcije pomoću računalnih algoritama. To je poznato kao rudarenje Bitcoina. Rudar koji uspješno završi novi blok nagrađen je Bitcoinom za svoj rad. Nagrade se plaćaju mrežnim naknadama koje se prenose na kupca i prodavatelja. Naknade mogu rasti ili padati ovisno o količini transakcija. Nakon što je kupnja kriptografski potvrđena, prodaja, odnosno blok se dodaje u blokchain. Blok je trajno vezan za sve prethodne blokove Bitcoin transakcija. [13]

5.3. Graf svijeća

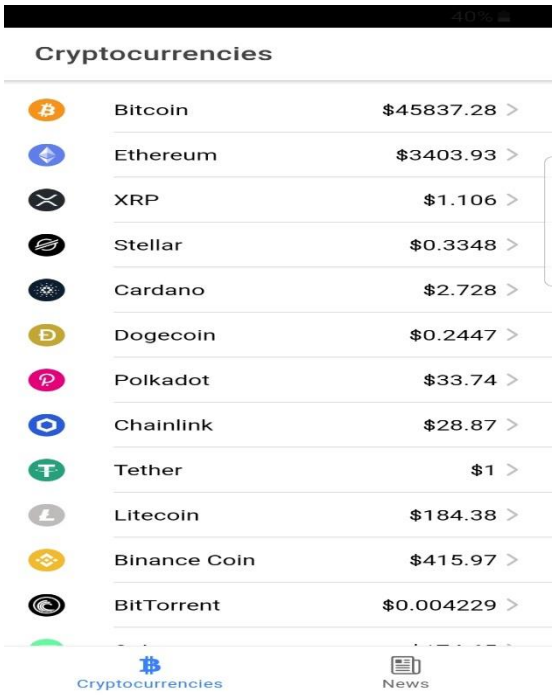
U aplikaciji se koristi graf svijeća (engl. candle graph) jer je mnogo korišten među trgovcima kriptovaluta i lako ga mogu razumjeti. Nude mnogo informacija i imaju jednostavan dizajn lagan za čitanje. Ovaj stoljetni stil grafikona razvijen je na japanskim tržištima riže. Naziv stila odnosi se na način na koji je svako vremensko razdoblje predstavljeno pravokutnikom s linijama koje izlaze s vrha i dna. Na grafu svaki svijećnjak predstavlja najvišu, najnižu, cijenu otvaranja i cijenu zatvaranja. U aplikaciji je graf prikazan u odnosu na vrijeme po satima, te svaki svijećnjak predstavlja jedan sat. Gornji ili donji dio svijećnjaka pokazat će otvorenu cijenu, ovisno o tome kreće li se imovina više ili niže tijekom razdoblja od jedan sat. Ako se cijena podigne, zatvarajući se više nego što se otvorila, otvoreno mjesto predstavlja donji dio tijela, a zatvaranje gornji dio. Ako se kreću cijene prema dolje, zatvarajući se niže nego što se otvorilo, otvoreno se prikazuje kao vrh svijećnjaka. Svijećnjaci koji se zatvaraju više su ispunjeni zelenom bojom, dok su svijećnjaci koji se zatvaraju niže ispunjeni crvenom bojom. Najveća cijena koja se pojavila

kroz sat vremena označena je crtom iznad svijeće, osim ako je cijena otvaranja ili zatvaranja bila najveća. Najniža cijena koja se pojavila kroz sat vremena označena je crtom ispod svijeće, osim ako je cijena otvaranja ili zatvaranja bila najniža. Cijena zatvaranja predstavlja cijenu koju je valuta imala kada je prošao sat vremena. Kako se cijena kreće se može lako pročitati iz grafa, gledajući boje i smjer svijeća. Ako su svijeće crvene to označava pad cijene, a ako su zelene označava rast cijene. Na slici 5.1. se može vidjeti primjer Bitcoin grafa iz aplikacije.

Slika 5.1. Izgled Bitcoin grafa svijeća iz aplikacije

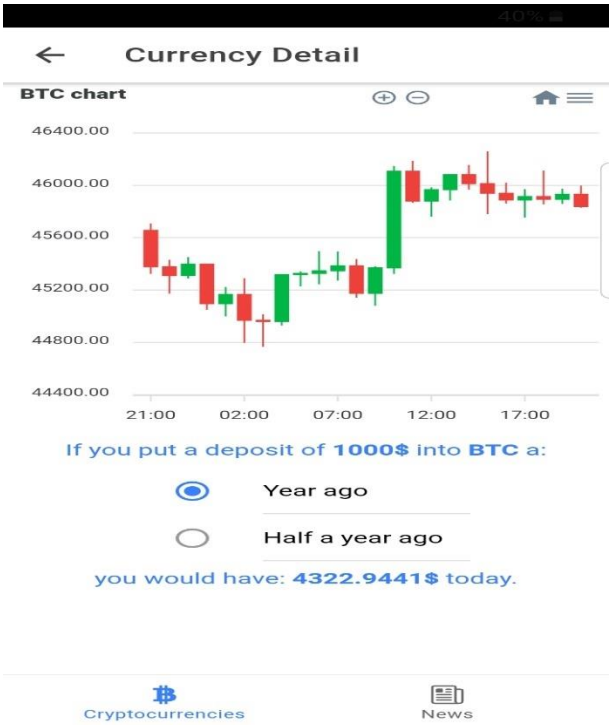
6. NAČIN KORIŠTENJA APLIKACIJE

Prvo što korisnik vidi kada otvori aplikaciju jest lista kriptovaluta sa njihovim trenutnim vrijednostima u dolarima, te dvije kartice, jednu koja sadrži spomenutu listu, a jednu koja sadrži vijesti iz svijeta kriptovaluta. Na slici 6.1. se može vidjeti snimka zaslona sa android uređaja.



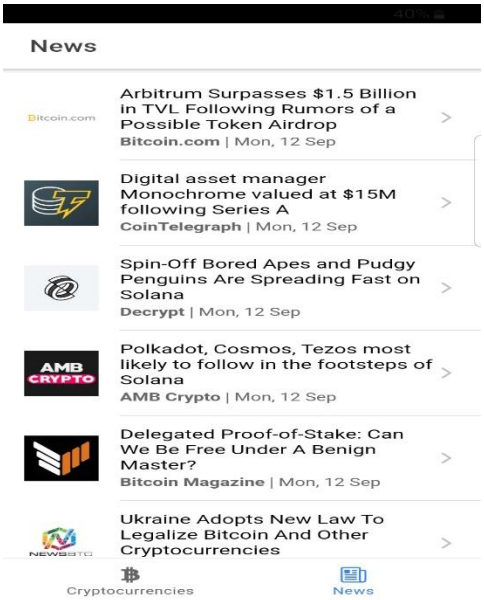
Slika 6.1. Izgled dijela aplikacije koji prikazuje listu kriptovaluta

Korisnik zatim može odabrati bilo koju od ponuđenih kriptovaluta kako bi vidio neke detalje o njoj ili može otići na karticu *News* kako bi pogledao najnovije vijesti. Na slici 6.2. se može vidjeti izgled aplikacije ako korisnik uđe u Bitcoin kako bi pogledao detalje. U detaljima je prikazan graf svijeća (engl. candle graph) za zadnja 24 sata, te količina novca u dolarima ako je korisnik uložio tisuću dolara u Bitcoin prije godinu dana.



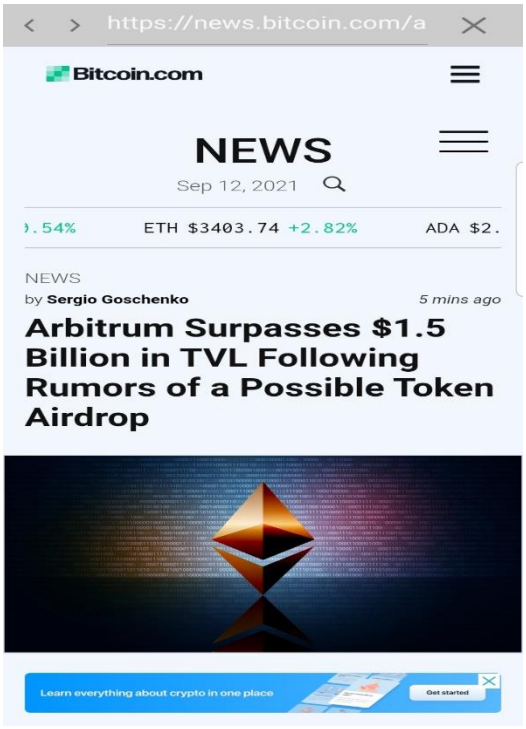
Slika 6.2. Izgled dijela aplikacije koji pokazuje detalje o kriptovaluti

Korisnik sa stranice koja prikazuje detalje kriptovalute može otići na vijesti ili se može vratiti natrag na listu kriptovaluta. Ako ode na vijesti prikazat će mu se lista vijesti kao što se vidi na slici 6.3.



Slika 6.3. Izgled dijela aplikacije koji prikazuje listu vijesti

Korisnik može odabrati bilo koju vijest kako bi vidio njene detalje te pročitao više o navedenoj temi. Vijest se otvara u aplikaciji, odnosno ne otvara se nova aplikaciji ili pretraživač koji će prikazati vijest. Izgled jedne od odabranih vijesti se može vidjeti na slici 6.4.



Slika 6.4. Izgled dijela aplikacije koji prikazuje detalje o odabranoj vijesti

Korisnik se zatim bez problema može vratiti nazad u aplikaciju kada pročita željenu vijest.

7. ZAKLJUČAK

U ovom radu je opisana izrada Ionic aplikacije za prikaz novosti o kriptovalutama. Već postoje neke slične aplikacije. Opisano je nekoliko sličnih rješenja koji imaju po nekoliko mogućnosti više od aplikacije u ovom radu. Za razvoj je odabran Ionic okvir jer omogućuje pokretanje aplikacije na web, android i iOS platformi pisanjem jednog koda. Uz Ionic se koristi Angular okvir koji uvelike olakšava razvoj. Umjesto njega se također mogao koristiti React ili Vue.js. Aplikacija koristi razne mogućnosti Angular okvira kao stranice, navigaciju, *InAppBrowser* koji poboljšava iskustvo korisnika itd. Korisniku su prikazane razne informacije o kriptovalutama koje se dohvaćaju preko CryptoCompare API-ja. On nudi mnogo različitih informacija, ali za ovu aplikaciju je iskorišteno samo nekoliko, te bi se u budućnosti mogle iskoristiti i ostale. Također je opisano par razlika između fiat i kripto valuta, zatim blockchain koji je temelje većine kriptovaluta. Mnogi trgovci su upoznati sa grafom svijeća (engl. Candle graph), te je zbog toga odabran za prikaz cijene u aplikaciji. Lako ga je čitati i jednostavan je za razumjeti. Kao poboljšanje aplikacije mogao bi se dodati alarm koji korisnik može postaviti kako bi ga upozorio ako cijene dosegne određeni iznos. Također bi se mogao napraviti widget koji korisnik može postaviti na zaslon mobilnog uređaja kako bi u svakom trenutku mogao vidjeti cijenu određene valute.

LITERATURA

[1] Google Play Store, Crypto Market Cap – Crypto tracker, Alerts, News, dostupno na: https://play.google.com/store/apps/details?id=com.zeykit.dev.cryptomarketcap&hl=en&gl=US [02.08.2021]

[2] Google Play Store, Crypto App – Widgets, Alerts, News, Bitcoin Prices, dostupno na: https://play.google.com/store/apps/details?id=com.crypter.cryptocyrrency&hl=en&gl=US . [02.09.2021]

[3] Google Play Store, CoinGecko – Bitcoin & Cryptocurrency Price Tracker, dostupno na: https://play.google.com/store/apps/details?id=com.crypter.cryptocyrrency&hl=en&gl=US. [02.09.2021]

[4] R., Gibb, StackPath, 31.05.2016, dostupno na: https://blog.stackpath.com/web-application/ [02.09.2021]

[5] C., Clifton, A., Gerber, Learn Android Studio, Aspress, Svibanj 2015

[6] V., Beal, Webopedia, 24.05.2021, dostupno na: https://www.webopedia.com/definitions/iphone-app/ [03.09.2021]

[7] J., Wilken, Ionic in Action, Manning, Rujan 2015

[8] Cordova, Apache, dostupno na: https://cordova.apache.org/docs/en/latest/ [03.09.2021]

[9] Parham, Medium, 05.04.2021, dostupno na: https://medium.com/codex/how-to-debug-ionic-apps-during-development-19d5df51c6bf [03.09.2021]

[10] K. Williamson, Learning AngularJS, O'Reilly Media, Inc., Sebastopol, 2015

[11] CryptoCompare, dostupno na: https://min-api.cryptocompare.com/documentation [03.09.2021]

[12] ApexCharts, dostupno na: https://apexcharts.com/docs/installation/ [03.09.2021]

[13] A, Lewis, The Basics of Bitcoins and Blockchains, Mango Media, 04.10.2018

SAŽETAK

Ionic okvir omogućuje programiranje web, android i iOS aplikacije koristeći jedan kod. Uvelike smanjuje vrijeme potrebno za izradu aplikacije za sve tri platforme. Može se koristiti sa Angular, React, Vue.js okvirom ili sa čistim JavaScriptom. U ovom radu je odabran Angular okvir jer uvelike olakšava izradu web aplikacije. Aplikacija nudi korisniku listu kriptovaluta sa njihovim trenutni vrijednostima u američkim dolarima. Korisnik može odabrati bilo koju valutu sa liste kako bi vidio detalje o njoj. Prikazan je graf svijeća (engl. candle graph) sa vrijednostima cijene u posljednja 24 sata. Također korisnik može vidjeti vrijednost koju bi imao ako je uložio u određenu valutu tisuću dolara prije godinu dana. Osim vrijednosti cijene korisnik može vidjeti najnovije vijesti iz svijeta kriptovaluta. Vijesti se otvaraju u aplikaciji i korisnik se lako može vratiti nazad na listu. Za dohvaćanje informacija o kriptovalutama koristi se CryptoCompare API.

Ključne riječi: Angular okvir, CryptoCompare API, Ionic okvir, Kriptovalute

ABSTRACT

Ionic application for tracking latest cryptocurrency news.

The Ionic framework allows you to program web, android and iOS applications using a single codebase. It greatly reduces the time required to build applications for all three platforms. It can be used with Angular, React, Vue.js frameworks or with pure JavaScript. The Angular framework was chosen in this paper because it greatly facilitates the development of web applications. The applications offers users list of cryptocurrencies with their current values in US dollars. The user can select any currency to see details about it. A candle graph with price values in the last 24 hours is shown. Also, the user can see the value he would have if he deposited a value of thousand dollars a year ago. In addition to the price value, the user can see the latest news from the world of cryptocurrency. The news open in the app and the user can easily return to the list. The CryptoCompare API is used to retrieve information about cryptocurrencies.

Keywords: Angular framework, CryptoCompare API, Cryptocurrencies, Ionic framework

PRILOZI

Izvorni kod za aplikaciju je dostupan na sljedećem repozitoriju:

https://github.com/Jeckiie/diplomski-rad

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 05.10.2021.

Ime i prezime studenta:	Jerko Kosić
Studij:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1067R, 06.10.2019.
Turnitin podudaranje [%]:	5%

Ovom izjavom izjavljujem da je rad pod nazivom**: Ionic aplikacija za praćenje novosti o kriptovalutama**

izrađen pod vodstvom mentora Izv. prof. dr. sc. Ivica Lukić

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:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 16.09.2021.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime studenta:	Jerko Kosić
Studij, smjer:	Diplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	D-1067R, 06.10.2019.
OIB studenta:	15117046197
Mentor:	Izv. prof. dr. sc. Ivica Lukić
Sumentor:	
Sumentor iz tvrtke:	
Predsjednik Povjerenstva:	Izv. prof. dr. sc. Mirko Köhler
Član Povjerenstva 1:	Izv. prof. dr. sc. Ivica Lukić
Član Povjerenstva 2:	Robert Šojo
Naslov diplomskog rada:	Ionic aplikacija za praćenje novosti o kriptovalutama
Znanstvena grana rada:	**Obradba informacija (zn. polje računarstvo)**
Zadatak diplomskog rada:	Rad opisuje način izrade web, android i ios aplikacije koristeći Ionic framework sa Angularom. Aplikacija preko API-ja povlači najnovije podatke o kriptovalutama te ih prikazuje korisniku.
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
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: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	16.09.2021.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum: