Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira

Sinković, Iva

Undergraduate thesis / Završni rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:582310

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-17**

Repository / Repozitorij:

University North Digital Repository

Sveučilište
Sjever

Završni rad br. 599/MM/2018

Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira

Iva Sinković, 0797/336

Varaždin, rujan 2018. godine

Sveučilište
Sjever

Odjel za Multimediju, oblikovanje i primjenu

Završni rad br. 599/MM/2018

Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira

Student

Iva Sinković, 0797/336

Mentor

Vladimir Stanisavljević, mr.sc.

Varaždin, rujan 2018. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR - 42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za multimediju, oblikovanje i primjenu

PRISTUPNIK: Iva Sinković | MATIČNI BROJ: 0797/336

DATUM: 19.9.2018 | KOLEGIJ: Programski alati 3

NASLOV RADA: Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira

NASLOV RADA NA ENGL. JEZIKU: Hybrid mobile application development using Ionic Framework

MENTOR: Vladimir Stanisavljević | ZVANJE: mr.sc.

ČLANOVI POVJERENSTVA:
1. doc.dr.sc. Dejan Valdec - predsjednik
2. dr.sc. Andrija Bernik, pred. - član
3. mr.sc. Vladimir Stanisavljević, v.pred. - mentor
4. dr.sc. Robert Logožar, v. pred. - zamjenski član
5.

Zadatak završnog rada

BROJ: 599/MM/2018

OPIS:

Izrada mobilnih aplikacija primjenom web tehnologija raširen je trend. Na taj način lakše se izvode aplikacije na više, prije svega mobilnih, platformi. U tu svrhu posebno je pogodno korištenje Node.js-a i odgovarajućih dodataka. Oblikovanje konačnog izgleda aplikacija lakše je ostvariti primjenom dodatnih razvojnih okvira poput Ionica.

U ovom radu potrebno je:
* opisati web pristupe izrade mobilnih aplikacija i navesti njihove prednosti i nedostatke
* opisati osnove Ionic razvojnog okvira te njegovu ulogu u izradi aplikacija
* usporediti Ionic s drugim sličnim razvojnim okvirima
* osmisliti i izraditi osnovnu mobilnu aplikaciju na kojoj će biti prikazane mogućnosti Ionic razvojnog okvira te demonstrirana većina mogućnosti
* po potrebi instalirati aplikaciju na javni poslužitelj ili izvesti kao mobilnu aplikaciju

Detaljno opisati sve korištene tehnologije i korake u radu potrebne da bi se ostvarilo traženo te opisati stečena iskustva i postignute rezultate.

ZADATAK URUČEN: 21.09.2018 | POTPIS MENTORA:

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER
KOPRIVNICA

Predgovor

Zahvaljujem se profesorima, kolegama na pomoći i mentoru Vladimiru Stanisavljeviću na razumijevanju i strpljenju. Zahvaljujem se prijateljima zbog neprestane pružane podrške, a najveću zahvalnost dugujem roditeljima radi pružanja emotivne, ali i financijske podrške tijekom cijelog mog školovanja. Na kraju bih se zahvalila svima koji su na bilo koji način doprinijeli izradi ovog rada.

Sažetak

U današnje vrijeme Internet je nešto bez čega ne možemo zamisliti naš život. Gdje god krenemo uvijek nas prati i u svakom trenutku možemo pogledati i pretražiti stvari koje nas zanimaju. Mobilne aplikacije u zadnjih nekoliko godina doživljavaju pravi procvat i potražnja za programerima tih aplikacija je sve veća. Mobilne aplikacije dijelimo na izvorne (nativne), web aplikacije i hibridne aplikacije. Nativne aplikacije razvijene su po mjeri za podržanu platformu te pisane programskim jezicima kao što su C++ i Java, web aplikacije su jednostavno responzivne web stranice. Hibridne mobilne aplikacije su kombinacija tih dvaju pristupa jer pružaju mogućnosti nativnih aplikacija, a pisane su pomoću web tehnologija (HTML5, CSS, JavaScript) te omogućuju razvoj na više platformi. U ovom radu opisana je izrada hibridne mobilne aplikacije korištenjem jednog od razvojnih okruženja – Ionic Framework-a. Ionic Framework je programski okvir koji omogućuje izradu hibridnih mobilnih aplikacija korištenjem jednostavnih tehnologija i resursa. U teorijskom dijelu rada opisana je korištena radna tehnologija i mogućnosti koje ona pruža za razvoj višeplatformskih mobilnih aplikacija. Uz to, navedene su dvije slične razvojne platforme, Apache Cordova i AngularJS, koje se mogu koristiti umjesto ili u suradnji s Ionic-om. U praktičnom dijelu bit će prikazana izrada aplikacije „iNorth“, namijenjena za studente Sveučilišta Sjever. Aplikacija ima svoj jedinstveni dizajn korisničkog sučelja te su navedeni elementi korišteni prilikom dizajniranja. Važnu ulogu u izradi aplikacije ima JSON koji sadrži sve podatke koji se ispisuju na stranicama aplikacije. Aplikacija sadrži komponentu Ionicons da bi se uključio set željenih ikona, komponentu social-sharing koja se koristi kako bi se omogućilo dijeljenje sadržaja u glavnim društvenim mrežama. Osim tih komponenti, koristi se i Google Maps za kojeg je potreban Google Maps API, dobiven prijavom na Google račun i izradom API ključa.

Ključne riječi: hibridne aplikacije, HTML5, CSS, JavaScript, Ionic Framework, JSON, API

Abstract

Nowadays, the Internet is something we cannot imagine our lives without. Wherever we go it follows us and we can at any time look up or search for the things that interest us. Mobile applications in the last few years are experiencing a real boom and the demand for developers of this kind is increasing. Mobile applications can be split into native, web and hybrid applications. Native applications are made exactly for a supported platform and programming languages, such as C++ and Java, and web applications are simply responsive web pages. Hybrid mobile applications are a combination of these two approaches because they provide native application capabilities, are written using web technologies (HTML5, CSS, JavaScript) and allow cross-platform development. This paper describes hybrid mobile application development using one of the platforms for building cross-platform apss – Ionic Framework. Ionic Framework is an open source framework for developing hybrid mobile applications using simple technologies and resources. The theoretical part of the paper describes used technology and the capabilities it provides for the development of cross-platform mobile applications. In addition, two similar framework platforms are mentioned, Apache Cordova and AngularJS, which can be used with or without Ionic. The practical part will show how was application, „iNorth“, designed for Universty North students, developed. The application has its own unique user interface design. Elements used in the design process are also included. An important function in creating the application has JSON, which contains all the data that is printed on the application pages. Application contains Ionicons component for including a set of desired icons, a social-sharing component that is used to allow content sharing in major social networks. In addition to these components, application uses Google Maps. This requires Google Maps API, obtained by signing in to your Google Account and creating an API key.

Key words: hybrid applications, HTML5, CSS, JavaScript, Ionic Framework, JSON, API

Popis korištenih kratica

HTML (HyperText Markup Language) – opisni internet jezik koji služi za izradu web stranica
CSS (Cascading Style Sheets) – stilski jezik koji opisuje izgled HTML dokumenta
SASS (Syntactically Awesome Style Sheets) – CSS ekstenzija s dodatnim posebnim značajkama poput varijabli, ugniježđenih pravila itd.
JSON (JavaScript Object Notation) – otvoreni standardni format koji koristi čitljiv tekst za prijenos objekt podataka koji se sastoje od parova atribut-vrijednost
UI (User Interface) – korisničko sučelje; mjesto komunikacije korisnika i nekog sustava, stroja ili uređaja
UX (User Experience) – korisničko iskustvo; osjećaj koji korisnik doživi prilikom uporabe određenog sustava, stroja ili uređaja
URL (Uniform Resource Locator) – jedinstveni lokator resursa
DOM (Document Object Model) – omogućuje pristup objektima stranice
SDK (Software Development Kit) – skup računarskih alata koje programer koristi kako bi napravio računarski program za određeni softverski paket, operativni sustav i slično
API (Application Programming Interface) – skup programskih pravila koji se programeri moraju držati da bi ostvarili željene rezultate kod programa
CLI (Command Line Interface) – sučelje, odnosno sredstvo interakcije s računalnim programom gdje korisnik izdaje naredbe programu u obliku uzastopnih linija teksta kako bi pregledavao i upravljao računalnim datotekama

Sadržaj

1. Uvod 1
2. Mobilne aplikacije 3
   2.1. Nativne aplikacije 4
   2.2. Web aplikacije 5
   2.3. Hibridne aplikacije 5
3. Ionic Framework 8
   3.1. Razvoj 8
   3.2. Kontrola stila Ionic korisničkog sučelja 9
   3.3. Pozadinski servisi kao usluga 10
      3.3.1. BaaS usluge Ionic platforme 10
   3.4. Ionic CLI 13
   3.5. Struktura Ionic aplikacije 14
   3.6. Instalacija Ionic Framework-a 15
4. AngularJS 16
   4.1. TypeScript 16
5. Apache Cordova 17
6. Praktični dio 18
   6.1. Dizajn 19
   6.2. Programiranje i kodiranje 21
      6.2.1. Stranica „Vijesti“ 23
      6.2.2. Stranica „Lifestyle“ 28
      6.2.3. Stranica „Događaji“ 30
      6.2.4. Stranica „Ponude“ 33
   6.3. Testiranje aplikacije 35
7. Zaključak 38
8. Literatura 40
9. Popis slika 41

1. Uvod

Vrijeme kada su jedine funkcije mobilnog telefona bile telefoniranje i slanje tekstualnih poruka odavno je iza nas. Brz razvoj tehnologije danas nam omogućava pristup informacijama gdje god bili uz pomoć pametnih telefona, krcatih raznim aplikacijama pomoću kojih radimo stvari koje prije nismo mogli ni zamisliti. Iako ne možemo predvidjeti budućnost, vrlo je vjerojatno da će uređaji nove generacije biti napredniji, odnosno brži, tanji, lakši, s većim brojem opcija nego danas, a možda će doći i do izuma uređaja koji će potpuno zamijeniti „pametne“ telefone te time pokrenuti novu revoluciju u tehnologiji. [1]

Kada govorimo o mobilnim aplikacijama, možemo reći da su one više od uobičajenih web aplikacija. Za razliku od web preglednika, mobilni uređaji sadrže senzore i razne funkcije (fotoaparat, vibracija i dr.). Upravo mogućnost pristupa tim funkcijama daje aplikacijama znatnu posebnost i prednost nad običnim responzivnim web stranicama. Razvoj aplikacija namijenjenih za „pametne“ telefone može se zasnivati na različitim pristupima. Tako možemo razlikovati nativne, web i hibridne aplikacije, a svaki od pristupa ima svoje prednosti i mane, što će biti dodatno opisano u radu.

Ionic Framework programski je okvir za izradu hibridnih mobilnih aplikacija. Ionic omogućuje izradu aplikacija podržanih na različitim platformama, upotrebom klasičnih Internet tehnologija (HTML5, CSS3, JavaScript). Navedeni okvir koristit ću u ovom radu, što će obuhvaćati pojmove vezane za spomenute tehnologije pri izradi hibridnih mobilnih aplikacija. Teorijski dio rada će, uz pristupe pri izradi mobilnih aplikacija, sadržavati opis programskog okvira Ionic, njegovu primjenu, komponente koje sadrži, podržane platforme, suradnju s Apache Cordova i AngularJS te će biti opisan postupak same instalacije Ionic Framework-a, uz primjere nekih osnovnih naredbi koje su upisuju u naredbenom prozoru. Nakon teorijskog dijela slijedi praktični dio u kojem ću opisati tijek izrade jednostavne hibridne aplikacije „iNorth“, namijenjene za studente Sveučilišta Sjever. Aplikacija će se sastojati od četiri kategorije („Vijesti“, „Lifestyle“, „Događaji“ i „Ponude“) koje bi studentima omogućile brzi pristup informacijama, bez da pretražuju dodatne internetske stranice. Tako će studentima biti dostupne vijesti vezane za Sveučilište, moći će čitati zanimljiva iskustva i savjete od drugih studenata, pronaći buduće događaje u blizini i okolici te vidjeti ponude koje se nude. Praktični dio obuhvaćat će tijek izrade dizajna same aplikacije, a zatim i proces kodiranja i programiranja pojedinih stranica.

Slika 1.1 Ionic Framework

2. Mobilne aplikacije

Mobilne aplikacije možemo definirati kao aplikacijske softvere koji su razvijeni kako bi radili na širokom spektru uređaja („pametnim“ mobilnim uređajima, tabletima i slično). One prvenstveno vrše interakciju s korisnikom i operativnim sustavom, ali i drugim aplikacijama. Osim što često koriste Internet, često se koriste i mobilni senzori poput GPS-a, žiroskopa, kamere ili mikrofona. Također, mobilne aplikacije većinu vremena provode radeći u pozadini bez da ih korisnik direktno koristi i u tom stanju najčešće šalju notifikacije ili prikupljaju podatke o lokaciji i slično. Većina osnovnih mobilnih aplikacija već je instalirana na uređajima te na taj su način dostupne korisniku. Također, aplikacije se mogu preuzeti online preko online trgovina po imenu app store (App Store, Google Play, Windows Phone Store). Prvobitna upotreba aplikacija bila je neprimjetna, ali velika potražnja dovela je do njihove ekspanzije. Danas se koriste u različite svrhe i postale su dio svakodnevice. Tržište mobilnih aplikacija nalazi se u stalnom porastu i danas je ono veoma razvijeno. Veliki značaj aplikacija očituje se u njihovom korisničkom sučelju (UI). UI predstavlja prostor gdje se ostvaruje interakcija između čovjeka i stroja. Primarni cilj korisničkog sučelja je lakša, efikasnija i ugodnija interakcija s uređajem u cilju postizanja kvalitetnog korisničkog iskustva (UX). Razumljivo, lako i prepoznatljivo korisničko sučelje želja je svakog korisnika, zbog učestale interakcije sa uređajem kojeg posjeduje. Aplikacije s lošim korisničkim sučeljem imaju malu vrijednost jer kao takve ne ostavljaju dobar utisak, a samim time smanjuju očekivano korisničko iskustvo. Prilikom razvoja mobilnih aplikacija važno je znati razliku između pojedinih vrsta aplikacija, kako bismo odabrali pristup koji ćemo koristiti. Mobilne aplikacije se tako dijele na: izvorne aplikacije (*engl. native*), web aplikacije i hibridne mobilne aplikacije.

2.1. Nativne aplikacije

Nativne mobilne aplikacije su aplikacije razvijene pomoću SDK (Software Development Kit) i programskog jezika određene mobilne platforme (npr. xCode/Objective-C – iOS, Eclipse/Java – Android, Visual Studio/C# – Windows Phone i drugih). Njihova prednost se odražava u tome što su napravljene po mjeri za podržanu platformu te tako ostvaruju najbolje rezultate koje je moguće ostvariti na toj platformi. Iskustvo kod ovog pristupa veće je od internetske ili hibridne opcije, s obzirom na čimbenike kao što su performanse, bogato korisničko iskustvo i sigurnost, što je uvelike prilagođeno nativnim aplikacijama. Nativna aplikacija je pouzdanija i po samom dizajnu jer korisnik upravlja i navigira aplikacijom, njenim sadržajem, dok su vizualni elementi i struktura već na uređaju kojeg korisnik posjeduje, što korisniku dodatno pruža neometan doživljaj. Pristup hardveru ili softveru uređaja omogućuje višu razinu pristupa razvojnim programerima da u potpunosti iskoriste prednosti koje nude Apple, Android i Microsoft kako bi bili sigurni da njihove aplikacije uključuju ažurirane značajke tih operacijskih sustava. Nativne aplikacije su samodostatne jer su podatci povezani s nativnom aplikacijom pohranjeni na uređaju, ali se mogu pohraniti na daljinu i pristupiti im u aplikaciji. [2] Osnovno ograničenje ovakvih mobilnih aplikacija je nemogućnost prenošenja aplikacije na drugu platformu, bez da se cijeli programski kod ne piše ispočetka. Nedostatak bi bio i veliki trošak za razvoj i održavanje nekoliko različitih verzija aplikacije. Izrada aplikacije za vodeće platforme podrazumijevala bi rad nekoliko različitih stručnjaka/programera za svaki operativni sustav. Sve navedeno pokazuje da je cijena i vrijeme razvoja nativnih aplikacija velika. [3]

2.2. Web aplikacije

Ako trebamo jednostavnu aplikaciju koja nema puno funkcionalnosti i poslovne logike te ne zahtijeva određene funkcionalnosti mobilnog uređaja, možemo odabrati jednostavno rješenje – web stranicu. Jedna responzivna web stranica može pružiti sve funkcionalnosti koje su korisniku potrebne, a razvoj traje kraće te je samim time jeftiniji. Ne moramo se brinuti o distribuciji preko trgovina aplikacija, a i samo ažuriranje aplikacija je krajnje jednostavno. Ovaj pristup se temelji na dokazanim standardnim web tehnologijama. To predstavlja najjednostavniji pristup razvoja aplikacija za mobilne uređaje, bez puno ulaganja. [4] Iako je ova opcija fleksibilna i atraktivna, korisničko iskustvo je ograničeno na skup osnovnih funkcija, bez mogućnosti pristupa nativnim funkcionalnostima mobilnog uređaja. Web mobilne aplikacije su nastale kao moguće rješenje nedostatka nativnih aplikacija (potreba za smanjenjem troškova i transparentnost u odnosu na platformu mobilnog uređaja). Web aplikacija lako pokriva više platformi za manji trošak, a na taj se način brže širi na tržištu. Nedostatak ovakvih aplikacija je u tome što nemaju pristup svim funkcionalnostima uređaja, a jedan od nedostataka je pojavljivanje specifičnih problema koji su uzrok sporijeg i lošijeg funkcioniranja aplikacije na određenim platformama.

2.3. Hibridne aplikacije

Između prethodno nabrojana dva pristupa nalaze se hibridne aplikacije. One podrazumijevaju pisanje jedne aplikacije koja će se moći koristiti na više platformi. Glavna ideja ovakvog pristupa je izraditi aplikaciju, korištenjem naprednih alata, koja će se kompilirati u nativnu aplikaciju i zatim biti distribuirana preko odgovarajuće trgovine aplikacijama. Ovakve aplikacije predstavljaju dobar kompromis, budući da ih je jednostavnije i brže izraditi, a pružaju većinu mogućnosti nativnih aplikacija. Hibridne aplikacije osmišljene su na način da se kombiniraju nativni i web pristup, a glavna prednost ove kombinacije je mogućnost razvoja aplikacija na više platformi. Najčešće korištene tehnologije su: HTML5 (engl. HyperText Markup Language), CSS (engl. Cascading Style Sheets) i JavaScript koje se pokreću preko web preglednika mobilnog uređaja (engl. web view) – UIWebView za iOS, WebView za Android i ostalih. [5] Stoga, nema potrebe za instaliranjem novog softvera. Ažuriranja i drugi popravci izvršavaju se samo na web poslužitelju bez intervencije korisnika u usporedbi s nativnim aplikacijama gdje korisnik mora ručno ažurirati aplikaciju. Za korištenje nativnih funkcionalnosti kao što su: senzori, kamera i drugo, koriste se

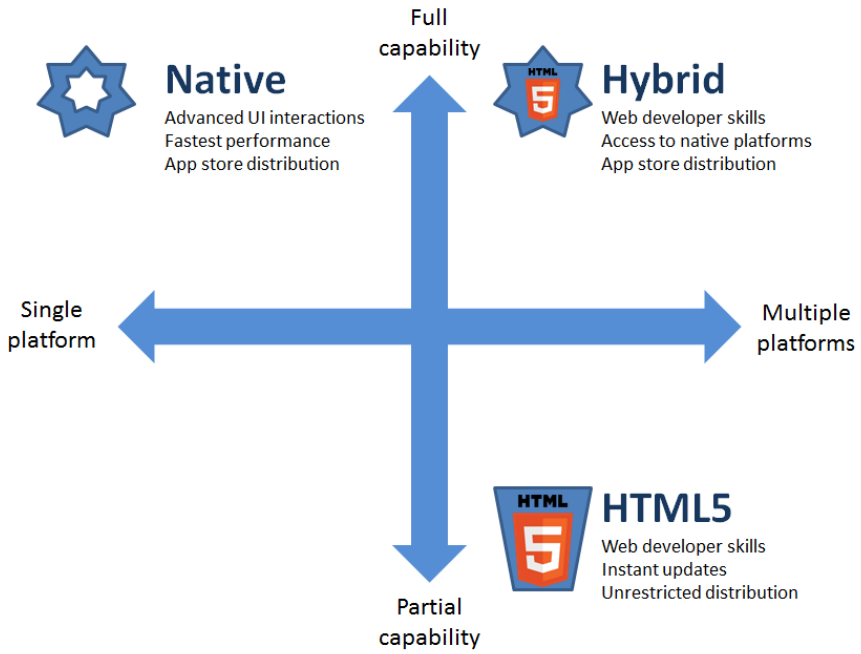
biblioteke koje su namijenjene razvoju hibridnih aplikacija. Nastanak hibridnih aplikacija izvedeno je uspostavljanjem apstraktnog sloja koji spaja nativni API (engl. Application Programming Interface) s JavaScript API-em. Problem ovog pristupa je u tome što za svaku platformu treba biti razvijen poseban kontejner, odnosno biblioteka, koja se nadograđuje za svaku novu funkcionalnost uređaja. Pored mogućnosti da se hibridne mobilne aplikacije razviju za relativno kratko vrijeme upotrebom standardiziranih web tehnologija, pri čemu se isti izvorni kod može ponovo koristiti za distribuciju aplikacije na više različitih platformi, one imaju određene prednosti i nedostatke. Pisanje programskog koda samo jednom, a da se automatski dobiju verzije za svaku od željenih platformi zvuči idealno, međutim hibridne aplikacije sa sobom nose određena ograničenja.

Ključ za uspješno korisničko iskustvo aplikacije jest njezina izvedba. Broj zahtjeva poslužitelja i zahtjeva za ravnotežu tereta dva su problema koji utječu na ukupnu izvedbu aplikacije. Dok je većina funkcija hibridnih aplikacija internetski utemeljena zbog opsežne upotrebe web-koda, moguće je pristupiti dijelovima programa napisanom u izvornom kodu izvan mreže. Tako se hibridne aplikacije mogu upotrebljavati na mreži i izvan mreže.

Alati koji omogućuju komunikaciju između web pogleda i izvorne platforme omogućuju rad hibridnih aplikacija. Ti potrebni alati nisu dio iOS, Android ili bilo koje druge mobilne platforme, nego su to drugi alati, kao što je Ionic, koji će se i koristiti u praktičnom dijelu ovog rada. Kada se hibridna aplikacija prevodi u jezik koji računalo, odnosno mobilni uređaj, razumije, web aplikacija prevodi se u izvornu aplikaciju za traženi uređaj. Hibridne aplikacije, naspram izvornih aplikacija i mobilnih web stranica imaju nekoliko prednosti, a to su: mogućnost razvoja jedne aplikacije za jednu platformu te je tada uz minimalna ulaganja moguće razviti istu aplikaciju za sve druge podržane platforme, omogućavanje razvoja mobilnih aplikacija sa znanjem tehnologija korištenih u razvoju web stranica i web aplikacija, pristup svim resursima i mogućnostima kao i kod izvornih aplikacija, jednostavnost i brzina razvoja. Hibridne aplikacije pružaju otpornu osnovu za razvoj mobilnih aplikacija, ali ipak omogućuju korištenje web platforme. Osim velikog broja prednosti hibridnih aplikacija postoje i neki nedostaci pri korištenju i razvoju hibridnih mobilnih aplikacija, kao što: ograničenja web pogleda – aplikacija može biti pokrenuta samo onoliko dobro koliko i sam web pogled, što znači da je izvođenje aplikacije vezano i ograničeno na kvalitetu izvođenja preglednika na pojedinoj platformi, pristupanje izvornim mogućnostima pomoću programskih dodataka – potreban pristup izvornim API-jima možda trenutno nije dostupan pa je u takvom slučaju potrebno prvo razviti programski dodatak koji će podržavati takav pristup, nedostatak izvornog korisničkog sučelja – bez alata. Najveći problem hibridnih aplikacija su njene performanse. Kod starijih verzija operativnih sustava telefona javlja se problem zbog nedovoljno

brzog renderiranja korisničkog sučelja i zastajanja efekata animacije ili tranzicije. Za neke funkcionalnosti (npr. widget na Android platformi) hibridne web aplikacije i dalje nemaju adekvatno rješenje. [6]

Apache Cordova uz saradnju sa AngularJS i Ionic framework-om, omogućava kreiranje hibridne mobilne aplikacije. Na slici 2.1 prikazane su osnovne razlike između nativnih, web i hibridnih aplikacija:



Slika 2.1 Nativne, hibridne i web aplikacije

3. Ionic Framework

Ionic je besplatan programski okvir koji omogućuje razvoj hibridnih mobilnih aplikacija, pomoću HTML-a, JavaScript-a, CSS-a [7] i u osnovi je kao i bilo koji drugi mobilni okvir. Ionic na jednom mjestu pruža set gotovih funkcionalnosti za brzu izradu mobilnih aplikacija koje su podržane na više platformi (zato i naziv hibridne). Izradom aplikacija koristeći Ionic okvir, isti kod se (poslije kompiliranja) može koristi za iOS, Android i Windows Phone verziju aplikacije. Upravo to ga svrstava u jedan jako koristan alat, zato što nije potrebno razvijati posebnu aplikaciju za svaku mobilnu platformu. Ionic okvir je otvorenog koda (engl. *Open Source*), što znači da mu je kod javno dostupan te ga programeri mogu modificirati kako žele.

3.1. Razvoj

Samu tvrtku osnivaju Ben Sperry i Max Lynch 2012. godine, dok se alfa verzija Ionic razvija 2013. godine. Konačna verzija (1.0) objavljena je 2015. godine. Iako nema veliku povijest iza sebe, dobro je prihvaćen u svijetu te su iste godine programeri diljem svijeta izradili preko 1,3 milijuna aplikacija. [8] Ionic verzija 1 ili Ionic V1 za razvoj hibridne mobilne aplikacije koristi AngularJS 1.x verzije te podržava izvorno korisničko sučelje ili web pogled za iOS 7 i novije verzije iOS mobilnog operacijskog sustava te Android 4.1. i novije verzije. Ionic 2 za razvoj hibridne mobilne aplikacije koristi AngularJS 2.0, koji nad starijom verzijom ima osjetnu prednost u brzini izvođenja, što je vrlo bitno za razvoj hibridnih mobilnih aplikacija. Jedini bitniji nedostatak prelaska na 2.0 inačicu jest ta što je sintaksa u AngularJS 2.0 inačici potpuno ili velikim dijelom drugačija te prelazak s Ionic V1 na V2 nije bio sasvim jednostavan. Ionic V2 pri razvoju hibridnih mobilnih aplikacija podržava iOS 8+ operacijske sustave, Android 4.4.+ operacijske sustave te Windows 10 mobilni operacijski sustav. Najnovija verzija Ionic V3 objavljena je 2017. godine te je korištena prilikom izrade praktičnog dijela ovog rada. Važno je napomenuti da Ionic V3 koristi AngularJS 4.0. Nadogradnja na najnoviju verziju donosi nove značajke, manje i brže aplikacije, podršku za noviju verziju TypeScript-a i ostalo.

3.2. Kontrola stila Ionic korisničkog sučelja

Ono što Ionic čini pravim okvirom je jako velika količina unaprijed dostupnih resursa. Tako su dostupne razne mobilne komponente [9], tipografija, interaktivna ponašanja elemenata i lako proširiv osnovni izgled. Tako da je Ionic od početka opremljen sa setom gotovih funkcija koje omogućuju kontrolu toka programa. Ionic Framework razvijen je u ekosustavu koji uključuje AngularJS (ili samo Angular) i Cordova-u za razvoj strukture i logike aplikacije te omogućuje pristupanje mogućnostima uređaja na kojem se aplikacija koristi. Ionic u navedenom ekosustavu omogućuje korisniku osjećaj izvorne mobilne aplikacije. Ti dodatni alati olakšavaju korištenje i integraciju Cordova-e i njenih programskih dodataka potrebnih za razvoj hibridne mobilne aplikacije. Ono što Ionic čini jedinstvenim je i podrška za SASS (*Syntactically Awesome Style Sheets*) CSS ekstenziju. Kako je pomoću ugrađenih CSS klasa i automatskog prepoznavanja platforme moguće urediti korisničko sučelje, tako je uz pomoć JavaScript-a moguće urediti i prilagoditi izgled i interakciju korisničkog sučelja.

Ionic znatno pojednostavljuje *front-end* dio aplikacije jer kroz primjenu klasa platforme, upotrebom JavaScript-a ili dinamičnih predložaka kontrolira izgled i UI mobilne aplikacije. Ionic Framewrok automatski dodjeljuje klase platforme u element stranice. Klasa se dodjeljuje s obzirom na uređaj koji pokreće aplikaciju. Ionic se tako pridržava preporučenih uputa za stiliziranje korisničkog sučelja ovisno o platformi.

Postoje dvije vrste klasa:

- klase platforme uređaja – koriste se za dobavljanje informacija o uređaju i dodjeljuju klase elementu stranice

  Klase ove vrste se koriste se s namjerom da se aplikacija što bolje prilagodi specifičnom izgledu i interakciji platforme. U klase platforme uređaja spadaju: *platform-browser*, *platform-cordova*, *platformwebview*, *platform-ios*, *platform-android* itd.

- klase OS verzije platforme – koriste se za prilagođavanje aplikacije za određenu verziju operativnog sustava

  Neke od ovakvih klasa su: *platform-ios8*, *platformios8_4*, *platform-android4*, *platform-android4_4* itd.

3.3. Pozadinski servisi kao usluga

Pozadinski servisi kao usluga ili BaaS (*engl. Backend as a Service*), drugim imenom MBaaS (*engl. Mobile Backend as a Service*), su načini spajanja web i mobilnih aplikacija sa servisima zasnovanima u oblaku (*engl. cloud*). BaaS umjesto korištenja međuprograma (*engl. middleware*) i samostalnog razvijanja pozadinskog sustava, stvara API i SDK, a oni omogućuju komunikaciju web i mobilnih aplikacija sa spremnikom u oblaku. Takav pristup razvijanju aplikacije programeru omogućuje fokusiranje na korisničko sučelje, bez trošenja vremena na razvoj samostalnog servisa. Usluge koje uobičajeni BaaS sustav nudi su: obavijesti (*engl. push notifications*), integracija sa socijalnim medijima (*engl. social networking integration*), lokacijske usluge (*engl. location services*) i upravljanje korisnikom (*engl. user management*). [10]

3.3.1. BaaS usluge Ionic platforme

- Ionic Push – omogućuje objavu notifikacija, odnosno obavijesti, korisnicima Ionic aplikacije. Obavijesti korisniku mogu dati informacije o aplikaciji, iako korisnik u tom trenutku možda ne koristi pametni mobilni uređaj. Ionic platformom moguće je stvoriti notifikacije koje će biti poslane korisniku kada on ispuni neki određeni kriterij. Slanje notifikacija se na Android i iOS platformi obavlja putem GCM *(Coogle Cloud Messaging*) i APNs (*Apple Push Notification Service*) servisa. Navedeni servisi zahtijevaju određeno vrijeme za njihovo postavljanje i konfiguraciju te se razlikuju prema načinu upotrebe i funkcionalnosti. Ionic Push servis djeluje kao posrednik između GCM-a, APNs-a i Ionic aplikacije. Ono što servis nudi je: API, izvještaj notifikacija, spremnik za informacije o uređaju i dodatne mogućnosti koje se ne mogu ostvariti koristeći APNs i GCM servise.
- Ionic Users – omogućuje registraciju i verifikaciju korisnika. Svaka aplikacija u kojoj je moguće stvoriti korisnički račun i komunicirati s korisnicima, sadrži način registracije te verifikacije tih korisnika. Navedeni servis, osim API-a, koji olakšava upravljanje registracijom i verifikacijom korisnika, omogućuje i web sučelje kojim programer pristupa informacijama o pojedinom korisniku.
- Ionic Deploy – omogućuje ažuriranje aplikacije na zahtjev, bez dugotrajnih procesa kroz Google Play usluge za Android platformu te App Store usluge za iOS platformu.

Ažuriranje aplikacije na zahtjev omogućeno je samo za promjene HTML, CSS, JavaScript i multimedijskih datoteka smještenih u *www* direktoriju projekta. Ako se želi dodati novi programski dodatak, potrebno je koristiti uobičajeni proces za ažuriranje mobilne aplikacije. Pomoću navedenog servisa, osim ažuriranja na noviju verziju, moguće je aplikaciju vratiti na starije verzije.

- Ionic Package – omogućuje izgradnju aplikacije za distribuciju. Glavna namjena ovog servisa je slanje aplikacije drugima, izgradnja aplikacije za platforme koje računalo programera ne podržava (npr. izgradnja iOS aplikacije na Windows platformi) te dohvaćanje APK i IPA datoteka koje je moguće predati Google Play-u za Android (APK) ili App Store-u za iOS (IPA) platformu.
- Ionic Analytics – omogućuje praćenje ponašanja korisnika Ionic aplikacije. Moguće je pratiti koliko je aktivnih korisnika kojeg dana te je ta mogućnost ugrađena kao zadana mogućnost. Također, mogu se dodati vlastiti događaji (*engl. event*) su praćeni od strane servisa.
- Security Profiles – omogućuje grupiranje Android i iOS akreditiva u jedan profil. Ti podatci se tada mogu koristiti od strane drugih servisa.
- API – omogućuje standardne i dodatne mogućnosti koje u nekim servisima nisu podržane. Putem Ionic platforme pristupa se tokenima na kojima je zasnovana verifikacija aplikacije. API također zahtjeva i korištenje HTTP metoda, omogućene AngularJS-om.

Ionic Framework, osim navedenih servisa, koristi neke dodatne alate koji omogućuju olakšani razvoj hibridne mobilne aplikacije. Neki od alata su:

- Ionic Lab – desktop aplikacija za Mac, Windows i Linux operacijske sustave. Pomoću jednostavnog korisničkog sučelja omogućuje lako korištenje nekih od navedenih Ionic servisa. Uz to omogućuje i velik dio CLI funkcionalnosti, na primjer, pokretanje aplikacije na emulatoru ili uređaju.
- Ionic View – omogućuje testiranje Ionic aplikacija na više uređaja, bez potrebe izgradnje i distribucije aplikacije. Ionic aplikaciju moguće je poslati bilo kome s iOS ili Android uređajem te instaliranom Ionic View aplikacijom. Bitno je napomenuti da Ionic View aplikacija ne podržava sve programske dodatke omogućene sa strane Cordova-e.

- Ionic Serve – omogućuje pokretanje Ionic aplikacije lokalno u pregledniku uređaja. Ionic Serve uz Ionic View može poslužiti kao alternativa testiranju hibridnih mobilnih aplikacija na fizičkim uređajima. Takav pristup nije preporučen, ali u situacijama s malo sredstava i vremena može biti vrlo dobra i korisna zamjena. Pokretanjem aplikacije u pregledniku moguće je i pratiti rad aplikacije u stvarnom vremenu. Izvođenjem Ionic Serve usluge s dodatkom -lab, dobije se aplikacija koja se lokalno izvodi u pregledniku, ali ujedno se simulira rad aplikacije na Android i iOS platformi.
- Ionic Creator – web aplikacija koja pomoću sučelja s postupcima povlačenja i puštanja željenih elemenata (*engl. drag and drop*) omogućuje razvoj mobilnih aplikacija.

Ionic Native – skup ES5/ES6/TypeScript omotača za Cordova programske dodatke koji omogućuju dodavanje izvorne funkcionalnosti, bez direktnog korištenja Cordova programskih dodataka.

3.4. Ionic CLI

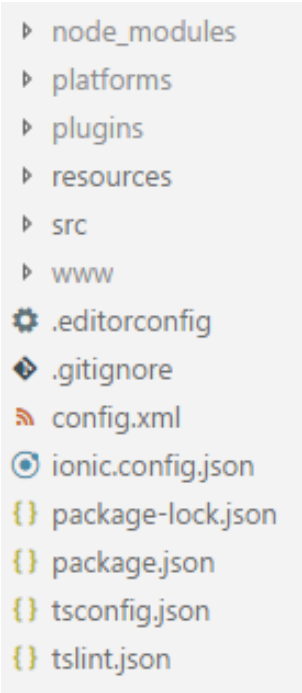
Još jedan vrlo koristan alat iz Ionic obitelji jest Ionic CLI (*engl. Command Line Utility*), odnosno Ionic komandna linija. Ionic CLI je primarni alat pri razvoju hibridne mobilne aplikacije pomoću Ionic Framework-a. CLI sadrži vrlo bitne naredbe kojima se programer može služiti tijekom razvoja aplikacije. Ionic CLI potrebno je instalirati na vlastito računalo te se naredbama pristupa preko komandne linije uređaja.

Neke od bitnijih naredbi su:

- start – naredba koja pokreće novi Ionic projekt sa svim potrebnim integracijama Cordova-e i AngularJS-a. Prilikom pokretanja moguće je izabrati gotove predloške ili prazan projekt te željenu verziju Ionic Framework-a.
- build – priprema i prevodi Ionic projekt za određenu platformu
- serve – pokreće lokalni server u kojem se pokreće aplikaciju u pregledniku računala
- platform – dodaje resurse potrebne za razvoj aplikacije za određenu platformu
- info – daje informacije o verzijama potrebnih programskih paketa (Cordova CLI-a, Ionic CLI-a, Ionic Framework-a, Node-a)

3.5. Struktura Ionic aplikacije

S obzirom da su Ionic aplikacije napravljene pomoću Cordova-e, koristi se Cordova struktura podataka. Prikaz uobičajene strukture Ionic aplikacije prikazana je na slici 3.1.



Slika 3.1 Prikaz strukture Ionic aplikacije

Neki od najvažnijih direktorija prikazanih na slici 3.1 su:

- plugins – ovdje se spremaju svi programski dodaci koje programer dodaje u projekt
- src – u ovom direktoriju se nalaze najbitnije datoteke za razvoj aplikacije, a to uključuje: *.scss* datoteku u kojoj se nalazi SASS aplikacije, *assets* mapa s korištenim resursima (slike, fontovi itd.), mapa sa stranicama (ekiranima) aplikacije
- www – ovdje se odvija sam razvoj mobilne aplikacije, preporučena struktura aplikacije automatski je postavljena od strane Ionic-a

3.6. Instalacija Ionic Framework-a

Ionic se treba instalirati u lokalni sistem (Local Disc (C:)). Pošto se pokreće na Node.js platformi, prvo je potrebno instalirati Node.js [11]. Zatim, s obzirom na kojem se operativnom sustavu prvenstveno testira aplikacija, potrebno je instalirati odgovarajuće alate za razvoj aplikacija. U ovom slučaju, fokus mi je bio operativni sustav Android.

1. korak – besplatno preuzimanje i instalacija Node.js-a (https://nodejs.org/en/)
2. korak – besplatno preuzimanje i instalacija Android Studio-a (potrebno zbog dodavanja programske zbirke Cordova; (https://developer.android.com/studio/)
3. korak – instalacija Ionic-a pomoću npm-a (node package manager)

```
npm install -g ionic
```

4. korak – početak izrade aplikacije, generiranje ime direktorija u kojem se nalazi projekt, biranje predložaka ili prazne stranice

```
ionic start myApp blank

ionic start myApp tabs

ionic start myApp sidemenu
```

4. AngularJS

AngularJS je strukturalni okvir (*engl. framework*) za dinamične web aplikacije. Angular okvir napisan je u programskom jeziku TypeScript, razvijenom od strane Microsofta. Njegova je svrha nadomjestiti nedostatke JavaScript-a. Kompatibilan je i preporučen *framework* za razvoj jednostraničnih web stranica, što mu je osnovna namjena, te hibridnih mobilnih aplikacija, zajedno s Cordova-om.

Glavni dijelovi Angular-a su [12]:

- moduli – logičko grupiranje komponenti, servisa, kontrolera i ostalih dijelova aplikacije u cjeline
- rute – ovisno u URL-u određeni HTML predložak, kojim upravlja određeni kontroler, se prikazuje kao klasična višestranična aplikacija, iako se sve odvija u istom dokumentu
- kontroleri – prvotna zadaća im je vezanje određene varijable za predočaj nad kojim upravljaju, te da vrše pozive prema servisima koji posežu u ogled, dohvaćaju podatke, vrše nekakve radnje nad njima te ih vraćaju kontroleru
- direktive – omogućava programerima da stvaraju vlastite DOM elemente
- servisi – služe za komunikaciju sa serverom, obradu podataka, formatiranje informacija i sl.

4.1. TypeScript

TypeScript je besplatan jezik otvorenog koda, razvijen od strane Microsoft-a [13]. Podržava pisanje programskog koda u JavaScript-u, korištenje biblioteka za JavaScript i pozivanje TypeScript koda iz samog JavaScript-a. Svrha mu je nadomjestiti nedostatke JavaScript-a pa se može nazvati i njegovim nadskupom. Svaki JavaScript program je zapravo i TypeScript program. TypeScript se pomoću specijalnog programa, tzv. prevoditelja (*engl. transpiler*) prevodi u jednostavan JavaScript programski kod koji se lako pokreće u bilo kojem internet pregledniku. TypeScript se koristi za razvoj klijentskih JavaScript aplikacija, ali i serverskih (Node.js) aplikacija. Glavne prednosti programskog jezika TypeScript su to što je objektno orijentiran te što omogućava strogo tipiziranje (*engl. strong typing*) nad JavaScript kodom, tj. možemo deklarirati tipove podataka. Time je omogućena provjera nad tipovima podataka prilikom prevođenja koda kako ne bi došlo do greške prilikom izvršavanja koda, što uvelike olakšava razvoj srednjih i velikih aplikacija.

5. Apache Cordova

Apache Cordova (ili samo Cordova) je besplatan *framework* za izgradnju *cross-platform* nativnih aplikacija upotrebom internet tehnologija: HTML5, CSS3 i JavaScript-a. Framework je nastao od strane Apache Software Foundation-a kao rezultat jednostavnije izrade *cross-platform* mobilnih aplikacija, zasnovan na kombinaciji nativne i web aplikacije. Primarna korist Cordova-e je omogućavanje upotrebe nativnih funkcionalnosti izvan mobilnog pretraživača. Kako bi se zaobišla ova ograničenja, Cordova implementira paket dodataka koji se protežu na nativne mogućnosti uređaja (npr. kamera, akcelerometar, kontakti) pokrećući web aplikaciju unutar nativnog kontejnera. Razvoj internet aplikacije koja ima interakciju sa nativnim hardverom i nativnom funkcionalnošću je bilo nemoguće, do pojavljivanja Cordova-e. Cordova aplikacija sastoji se od nekoliko komponenti.

Najvažniji dijelovi u arhitekturi aplikacije su [14]:

- web pogled *(engl. Web View*) – aplikaciji može pružiti cjelokupno korisničko sučelje
- web aplikacija (*engl. WebApp*) – mjesto gdje je smješten aplikacijski kod, aplikacija je implementirana kao web stranica koja se zadano sastoji od lokalne datoteke (*index.html*), koja pokazuje na CSS pravila, JavaScript kod te ostale datoteke potrebne za njeno pokretanje
- programski dodaci (*engl. plugins*) – sastavni dio Cordova ekosustava koji pružaju sučelje za komunikaciju između Cordova-e i izvornih komponenti te veze do standardnih API-a uređaja

6. Praktični dio

Za razvoj hibridne mobilne aplikacije korišteno je Visual Studio Code razvojno okruženje. Visual Studio Code besplatan je program za razvoj aplikacija u više programskih jezika [15], a brojne mogućnosti olakšavaju i ubrzavaju korisnikov rad. Program je razvijen od strane Microsoft –a za operativne sustave Windows, Linux i MacOS.

U ovom poglavlju bit će prikazan postupak izrade jednostavne mobilne aplikacije pomoću Ionic Framework-a. Bitno za napomenuti je da je u praktični dijelu ovog rada prikazan *frontend* dio, odnosno nedostaju neke značajke koje bi omogućile potpuno funkcionalnu mobilnu aplikaciju.

Ideja rada je stvoriti aplikaciju za studente Sveučilišta Sjever na kojoj će se nalaziti informativne stvari i stvari zabavnog karaktera. Tako je osmišljena aplikacija „iNorth“ koja to i pruža. Aplikacija je strukturirana jednostavno, kako bi korisnici imali kvalitetno korisničko iskustvo. Podijeljena je na četiri kategorije, odnosno stranice, koje su jasno prikazane na izborniku u obliku kartica (engl. tabs). Kategorije aplikacije su: „Vijesti“, „Lifestyle“, „Događaji“ i „Ponude“.

Stranica „Vijesti“ sadržavat će popis članaka objavljenih na službenoj web stranici Sveučilišta Sjever, koji će se moći pristupiti u aplikaciji. „Lifestyle“ će sadržavati članke s raznim savjetima i ostalim zabavnim temama vezanih za studentski život. Na stranici „Događaji“ moći se pregledati događanja u blizini, ali i okolici te pregledati pojedinosti o svakom događaju. Stranica „Ponude“ služit će kao oglasnik gdje se prikupljaju oglasi i objave korisne studentima, npr. oglasi za posao, instrukcije, najam stanova.

6.1. Dizajn

Prva stvar nakon razrade ideje aplikacije bila je izrada njenog vizualnog identiteta. Kako se ne bi koristila zadana Ionic ikona aplikacije, bilo je potrebno izraditi novu. Ikona je izrađena u Adobe Illustrator-u kako bi odgovarala svim rezolucijama ikona za Android i iOS. Pri izradi je korištena jedinstvena crvena boja Sveučilišta Sjever.

Da bi ikona bila uspješno implementirana u aplikaciju, potrebno ju je definirati u globalnoj konfiguracijskoj datoteci.

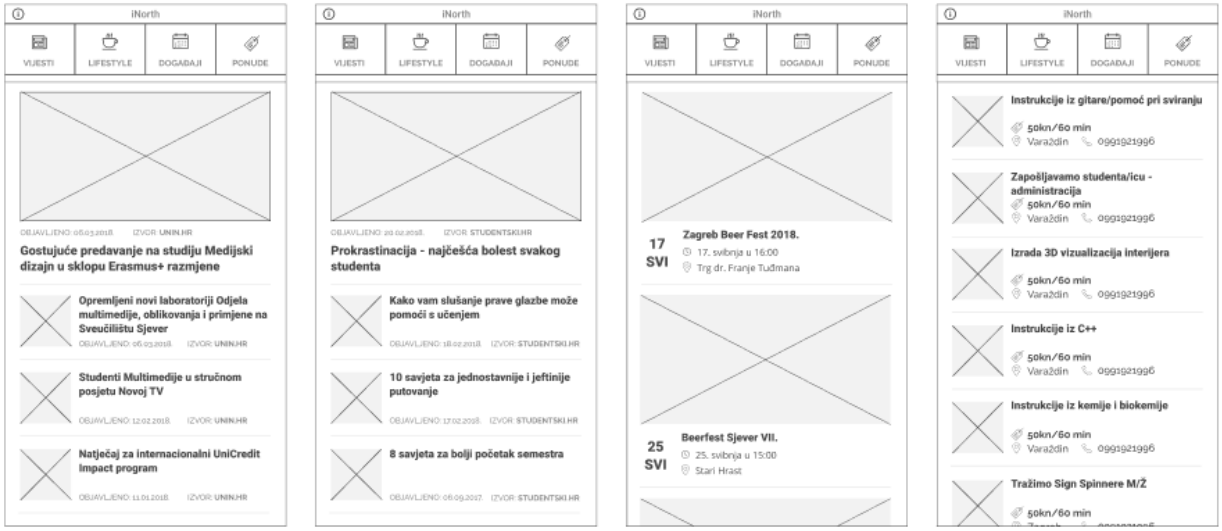
Slika 6.1 Ikona aplikacije

```
<platform name="android">
    <allow-intent href="market:*" />
    <icon density="ldpi" src="resources/android/icon/drawable-ldpi-icon.png" />
    <icon density="mdpi" src="resources/android/icon/drawable-mdpi-icon.png" />
    <icon density="hdpi" src="resources/android/icon/drawable-hdpi-icon.png" />
    <icon density="xhdpi" src="resources/android/icon/drawable-xhdpi-icon.png" />
    <icon density="xxhdpi" src="resources/android/icon/drawable-xxhdpi-icon.png" />
    <icon density="xxxhdpi" src="resources/android/icon/drawable-xxxhdpi-icon.png" />
</platform>
```

Slika 6.2 Implementacija ikone u aplikaciju

Prije same izrade, odnosno programiranja aplikacije, ona najprije treba imati svoj kostur, što znači da treba napraviti skice (*engl. wireframe*) pojedinih ekrana kako bi se odredio izgled i razmještaj elemenata stranice. Skice stranica (*slika 6.3*) aplikacije izrađivala sam pomoću alata Figma, baziranog u web pregledniku, koji služi za dizajniranje korisničkog sučelja i stvaranje prototipa. Osim toga, navedeni alat omogućuje kolaborativno okruženje i interakciju s članovima tima, ali i ostalim korisnicima.

Najvažnija stvar kod dizajna aplikacije je da je dizajn uniformni, a to znači da svi elementi koji se pojavljuju nekoliko puta u aplikaciji svugdje izgledaju isto. Prilikom osmišljavanja dizajna treba obratiti pozornost na današnje standarde dizajna mobilnih aplikacija, a samim time na odabir boja, tipografije i smislenih grafika. Kod dizajna aplikacije „iNorth“ koristila sam pet boja, korištenih kroz cijelu aplikaciju, koje sam odabrala zbog lakog snalaženja i korištenja iste. Također, pomno birani fontovi uvelike poboljšavaju korisničko iskustvo.



Slika 6.3 Skice osnovnih stranica aplikacije

Slika 6.4 Korištena paleta boja

Slika 6.5 Korišteni fontovi

6.2. Programiranje i kodiranje

U prvom dijelu rada opisana je instalacija Ionic-a i kreiranje novog projekta. Od tri predloženih mogućih projekta, za ovu aplikaciju odabran je rad s predloškom izbornika u obliku kartica (*engl. tabs*). Generirana mapa projekta otvorena je u navedenom odabranom razvojnom okruženju. Najvažniji direktorij za izradu aplikacije je „src“ direktorij u kojem se nalazi glavna *index.html* stranica gdje su dodani dodaci poput CSS stilova i TypeScript datoteka. U „src“ direktoriju nalaze se i ostale mape unutar kojih se odvija svo programiranje i potrebne izmjene, kao što su: *app* mapa s glavnom *.scss* datotekom u kojoj se nalazi CSS/SASS kod, *assets* mapa s resursima upotrijebljenih u aplikaciji (slike, fontovi, .json datoteke), mapa *pages* sa stranicama (ekranima) aplikacije.

Kao što je spomenuto prije, aplikacija se sastoji od četiri stranice. Kako bi one bile ispunjene sadržajem, potrebno je stvoriti navigacijsku traku koja će sadržavati poveznice na pojedine stranice. U prostor kartica dodane su ikone koje ih predstavljaju, a one su preuzete s Ionic Framework službene web stranice kao dio komponente Ionicons, koja sadrži set raznovrsnih ikona za svaku svrhu [16]. Stvorene stranice potrebno je definirati u .ts datoteci (slika 6.7), a zatim ih deklarirati u glavnoj datoteci s modulima kako bi ih bilo moguće prikazati i kliknuti (slika 6.8). U tabs.html osim izbornika dodana je ikona koja je ujedno i poveznica na stranicu Impressum. Stranica je napravljena u svrhu prikaza informacija o samoj aplikaciji i programeru.

Slika 6.6 Primjer Ionicons dokumentacije

```
@Component({
  templateUrl: 'tabs.html'
})
export class TabsPage {

  tab1Root = VijestiPage;
  tab2Root = LifestylePage;
  tab3Root = DogadajiPage;
  tab4Root = PonudePage;
  impressumPage = ImpressumPage;

  constructor() {

  }

}
```

Slika 6.7 Definiranje stranica u tabs.ts datoteci

```
@NgModule({
  declarations: [
    MyApp,
    LifestylePage,
    DogadajiPage,
    VijestiPage,
    PonudePage,
    TabsPage,
    ImpressumPage
  ],
```

Slika 6.8 Deklaracija u app.module.ts datoteci

```
<ion-tabs tabsPlacement="top">
  <ion-tab [root]="tab1Root" tabTitle="Vijesti" tabIcon="ios-paper" md="md-paper" id="tab-t0-0"></ion-tab>
  <ion-tab [root]="tab2Root" tabTitle="Lifestyle" tabIcon="ios-cafe" md="ios-cafe-outline" id="tab-t0-1"></ion-tab>
  <ion-tab [root]="tab3Root" tabTitle="Događaji" tabIcon="ios-calendar" md="md-calendar" id="tab-t0-2"></ion-tab>
  <ion-tab [root]="tab4Root" tabTitle="Ponude" tabIcon="ios-pricetag" md="md-pricetag" id="tab-t0-3"></ion-tab>
</ion-tabs>
<ion-header class="header header-md">
  <ion-navbar class="toolbar toolbar-md">
    <ion-title class="toolbar-title toolbar-title-md title title-md">iNorth</ion-title>
    <ion-buttons left>
      <button ion-button icon-only [navPush]="impressumPage">
        <ion-icon name="ios-information-circle" md="ios-information-circle-outline"></ion-icon>
      </button>
    </ion-buttons>
  </ion-navbar>

</ion-header>
```

Slika 6.9 tabs.html

6.2.1. Stranica „Vijesti“

Stranica je osmišljena na način da sadrži članke s vijestima i obavijestima objavljenih na službenoj stranici Sveučilišta Sjever. Pošto web stranica nema omogućen RSS, podatci su prikazani preko .json datoteke spremljene lokalno u mapi projekta.

JSON (JavaScript Object Notation) je sintaksa, odnosno notacija, zapisa objekata u JavaScript-u. Koristi se kao zamjena XML-u jer je jednostavniji i čitljiviji. Sastoji se od parova naziva (ključeva) i podataka.

Za prikaz članaka na stranici, u .json datoteci bilo je potrebno pohraniti sljedeće podatke: izvor članka, naslov, datum objave i URL fotografije koja predstavlja članak. Podatci su upisivani ručno, a kao primjer je uzeto nekoliko članaka sa stranice Sveučilišta.


```
[
    {
    "izvor1": "IZVOR: UNIN.HR",
    "title1": "Najava novih studija na Danu otvorenih vrata u Sveučilišnom centru Varaždin",
    "urlToImage1": "assets/imgs/picture1.jpg",
    "publishedAt1": "OBJAVLJENO: 22.03.2018."
    },
    {
    "izvor": "IZVOR: UNIN.HR",
    "title2": "Sveučilište Sjever partner na Danu otvorenih vrata Tehnološkog parka Varaždin",
    "urlToImage": "assets/imgs/picture2.png",
    "publishedAt": "OBJAVLJENO: 21.03.2018."
    },
    {
    "izvor": "IZVOR: UNIN.HR",
    "title3": "'Dobar Film' traži suradnike na projektu kratkometražnog filma",
    "urlToImage": "assets/imgs/picture3.png",
    "publishedAt": "OBJAVLJENO: 19.03.2018."
    },
    {
    "izvor": "IZVOR: UNIN.HR",
    "title4": "Dan otvorenih vrata Tehnološkog parka Varaždin",
    "urlToImage": "assets/imgs/picture4.png",
    "publishedAt": "OBJAVLJENO: 14.03.2018."
    },
    {
    "izvor": "IZVOR: UNIN.HR",
    "title5": "Opremljeni novi laboratoriji Odjela multimedije, oblikovanja i primjene na Sveučilištu Sjever",
    "urlToImage": "assets/imgs/picture5.png",
    "publishedAt": "OBJAVLJENO: 06.03.2018."
    }
]
```


Slika 6.10 .json datoteka za stranicu "Vijesti"

Da bi se podatci iz napravljene .json datoteke prikazali na stranici, potrebno je kreirati uslugu (*engl. service*). Usluga obuhvaća bilo koju funkciju ili vrijednost nekog tipa podataka dostupna nad svim dijelovima aplikacije. Unutar aplikacije, usluge su najčešće korištene od strane komponenti. Cilj je uslugama prepustiti sve složenije zadatke, kao što je dohvaćanje podataka sa servera ili validacija korisničkog računa, a da komponente budu što jednostavnije. Ideja usluga bazirana je nad konceptom ubrizgavanja ovisnosti (*engl. dependency injection*). Što se tiče ovisnosti, uobičajeno jedna klasa treba drugu klasu za rad. Mehanizam ubrizgavanja ovisnosti brine o tome da sve ovisnosti sadržane unutar opskrbljivača budu registrirane, ali i da ovisnosti mogu biti ubrizgane tamo gdje su potrebne [17]. Kako bi injektor znao koje su mu usluge dostupne i kako ih kreirati, potrebno je registrirati pružatelja usluge (*engl. provider*) za svaku uslugu. Tako je za potrebe prikaza članaka na stranici registriran provider *article-service.* Registrirani provider potrebno je uključiti u TypeScript datoteci stranice na kojoj se usluga izvađa (slika 6.12).

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import 'rxjs/add/operator/map';
import 'rxjs/add/operator/do';
import 'rxjs/add/operator/catch';
import { Observable } from 'rxjs/Observable';

@Injectable()
export class ArticleService {

  configUrl = 'assets/articles.json';
  constructor(private http: HttpClient) {

}

  getConfig() {
  return this.http.get(this.configUrl);
}
}
```

Slika 6.11 provider article-service

```
@Component({
  selector: 'page-home',
  templateUrl: 'vijesti.html'
})
export class VijestiPage implements OnInit{

  public articles: any;

  constructor(private articleService: ArticleService) {

  }

  ngOnInit(){
    this.articleService.getConfig().subscribe(data => {
          console.log(data);
          this.articles = data;
    });
}

}
```

Slika 6.12 vijesti.ts

Na samoj HTML stranici (*vijesti.html*) korištena je posebna direktiva ***ngFor** koja generira sadržaj na stranici (slika 6.13). Direktiva je, tehnički gledano, TypeScript klasa koja sadrži članove kojima utječemo na svojstva DOM-a – strukturu, ponašanje i izgled. Jednako kao i komponenta, direktiva također ima svoj selektor. Direktivu koristimo tako da ju "pričvrstimo" na oznaku HTML elementa kojim želimo manipulirati.

Direktive se dijele u tri kategorije:

- komponente
- strukturalne direktive
- atributne direktive

Za ovaj primjer najvažnije su strukturalne direktive, kao što je spomenuta *n**gFor** direktiva. Strukturalne direktive mogu duplicirati, ukloniti ili premjestiti neki element unutar DOM-a, a te se radnje izražavaju na sadržaj unutar web preglednika. Strukturalna direktiva ***ngFor** se koristi kako bismo dobili više instanci nekog HTML predloška te ona govori Angular-u da ispiše onoliko elemenata koliko lista ima članova, u ovom slučaju lista s člancima.

```
<ion-content padding>
  <div class="status0"></div>
  <ion-list>
    <ion-item *ngFor="let article of articles">
    <img src="{{article.urlToImage1}}" [navPush]="article1Page" class="article-pic1"/>
    <img src="{{article.urlToImage}}" class="article-pic2"/>

    <div class="naslov-clanka" [navPush]="article2Page">{{article.title2}}</div>
    <div class="naslov-clanka" [navPush]="article3Page">{{article.title3}}</div>
    <div class="naslov-clanka" [navPush]="article4Page">{{article.title4}}</div>
    <div class="naslov-clanka" [navPush]="article5Page">{{article.title5}}</div>

    <div class="container1">
      <div class="datum-clanka1">{{article.publishedAt1}}</div>
      <div class="izvor-clanka">{{article.source1}}</div>
      <div class="izvor-clanka1"><a href="https://www.unin.hr/"><b>{{article.izvor1}}</b></a></div>
    </div>

    <div class="naslov-clanka1" [navPush]="article1Page">{{article.title1}}</div>

    <div class="container">
      <div class="datum-clanka">{{article.publishedAt}}</div>
      <div class="izvor-clanka">{{article.source}}</div>
      <div class="izvor-clanka1"><a href="https://www.unin.hr/"><b>{{article.izvor}}</b></a></div>
    </div>

  </ion-item>
  </ion-list>
</ion-content>
```

*Slika 6.13 vijesti.html i *ngFor direktiva*

Kako bi stranica bila upotpunjena, osim generiranog HTML dokumenta, potrebno je rasporediti elemente i urediti izgled stranice pomoću CSS-a. Svaka stranica aplikacije sadrži vlastitu .scss datoteku zbog lakšeg snalaženja. Elementima su dodijeljene klase i uređene prema osmišljenom dizajnu mobilne aplikacije.

```
page-home {
.naslov-clanka1 {
    font-family: "Roboto-Black";
    color: #474747;
    font-size: 12pt;
    white-space: normal;
}
.naslov-clanka {
    font-family: "Roboto-Black";
    color: #474747;
    font-size: 11pt;
    height: auto;
    white-space: normal;
    display: -webkit-inline-box;
    position: absolute;
    padding-left: 10px;
}
.datum-clanka {
    font-family: "Raleway-Medium";
    color: #828282;
    font-size: 8pt;
    display: inline-block;
}
.datum-clanka1 {
    font-family: "Raleway-Medium";
    color: #828282;
    font-size: 8pt;
    display: flex;
}
.izvor-clanka {
    font-family: "Raleway-Medium";
    font-size: 8pt;
    padding-left: 5px;
    display: flex;
}
a {
    color: #C22929;
}
a:link {
    text-decoration: none;
}
.article-pic1 {
    padding-bottom: 5px;
}
.article-pic2 {
    width: 65px;
    height: 65px;
}
.container {
    display: inline-flex;
    position: absolute;
    padding-left: 78px;
    margin-top: -25px;
}
.container1 {
    display: flex;
    padding-bottom: 10px;
}

}
```

Slika 6.14 vijesti.scss

Pojedinačne stranice članaka rađene su HTML-om, a njihov izgled oblikovan CSS-om. Na stranice članaka dodani su gumbi za dijeljenje sadržaja na društvenim mrežama. Za to je bilo potrebno preuzeti dodatnu biblioteku *social-sharing* [18]. Instalacija je bila jednostavna jer je na Ionic Framework web stranici opisana instalacija biblioteke i sama njena primjena za sve glavne društvene mreže.

```
<div class="social">
  <h3 class="share">Podijeli s prijateljima</h3>
  <img src="assets/imgs/facebook.png" (click)="facebookShare()" class="ikona"/>
  <img src="assets/imgs/twitter.png" (click)="twitterShare()" class="ikona"/>
  <img src="assets/imgs/whatsapp.png" (click)="whatsappShare()" class="ikona"/>
  <img src="assets/imgs/viber.png" (click)="viberShare()" class="ikona"/>
  <img src="assets/imgs/instagram.png" (click)="instagramShare()" class="ikona"/>
</div>
```

Slika 6.15 dio koda HTML stranice članka – social-sharing

```
export class Article1Page{

  message: string;
  image: string;
  url: string;
  appName: string;
  subject: string;
  pasteMesageHint: string;

  constructor(public navCtrl: NavController, private socialSharing: SocialSharing) {

  }

  whatsappShare() {
    this.socialSharing.shareViaWhatsApp("Najava novih studija na Danu otvorenih vrata u Sveučilišnom centru Varaždin",
        "https://www.unin.hr/wp-content/uploads/DSC_0370-300x200.jpg",
        "https://www.unin.hr/2018/03/dan-otvorenih-vrata-u-sveucilisnom-centru-varazdin/").then(() => {
      console.log("shareViaWhatsApp: Success");
    }).catch(() => {
      console.error("shareViaWhatsApp: failed");
    });
  }
```

Slika 6.16 dio TypeScript koda – primjer za društvenu mrežu WhatsApp

6.2.2. Stranica „Lifestyle“

Stranica je osmišljena da sadrži članke zabavnog karaktera sa studentskog portala [19]. To podrazumijeva članke sa savjetima od drugih studenata te razne zanimljivosti o studentskom životu. Struktura stranice je jednaka kao i stranica „Vijesti“ pa su vrste podataka na njoj iste (izvor članka, naslov, datum objave, URL fotografije) , a samim time i njen postupak izrade. Razlika je u sadržaju članaka koji se na stranici prikazuju. Ponovno je bilo potrebno napraviti .json datoteku s odgovarajućim člancima te ju pomoću provider-a i usluga prikazati na stranici. Također, kao i članci sa stranice „Vijesti“, članci pripadajuće stranice „Lifestyle“ sadrže gumbove za dijeljenje sadržaja na društvenim mrežama.


```
[
    {
    "izvor1": "IZVOR: STUDENTSKI.HR",
    "title1": "5 načina na koja putovanja pomažu u razvoju karijere",
    "urlToImage1": "assets/imgs/picture6.jpg",
    "publishedAt1": "OBJAVLJENO: 14.04.2018."
    },
    {
    "izvor": "IZVOR: STUDENTSKI.HR",
    "title2": "Jednostavan vodič za rješavanje stresa",
    "urlToImage": "assets/imgs/picture7.jpg",
    "publishedAt": "OBJAVLJENO: 04.04.2018."
    },
    {
    "izvor": "IZVOR: STUDENTSKI.HR",
    "title3": "10 stvari koje vam mladi profesori neće reći",
    "urlToImage": "assets/imgs/picture8.jpg",
    "publishedAt": "OBJAVLJENO: 31.03.2018."
    },
    {
    "izvor": "IZVOR: STUDENTSKI.HR",
    "title4": "Osam načina kako unijeti više sreće u svoj život",
    "urlToImage": "assets/imgs/picture9.jpg",
    "publishedAt": "OBJAVLJENO: 30.03.2018."
    },
    {
    "izvor": "IZVOR: STUDENTSKI.HR",
    "title5": "Savjeti za postavljanje ostvarivih ciljeva",
    "urlToImage": "assets/imgs/picture10.jpg",
    "publishedAt": "OBJAVLJENO: 28.03.2018."
    }
]
```


Slika 6.17 .json datoteka za stranicu "Lifestyle"

```
<ion-content padding>
  <div class="status1"></div>
  <ion-list>
    <ion-item *ngFor="let advice of advices">
      <img src="{{advice.urlToImage1}}" [navPush]="advice1Page" class="article-pic1"/>
      <img src="{{advice.urlToImage}}" class="article-pic2"/>

      <div class="naslov-savjeta" [navPush]="advice2Page">{{advice.title2}}</div>
      <div class="naslov-savjeta" [navPush]="advice3Page">{{advice.title3}}</div>
      <div class="naslov-savjeta" [navPush]="advice4Page">{{advice.title4}}</div>
      <div class="naslov-savjeta" [navPush]="advice5Page">{{advice.title5}}</div>

      <div class="container2">
      <div class="datum-savjeta1">{{advice.publishedAt1}}</div>
      <div class="izvor-savjeta">{{advice.source1}}</div>
      <div class="izvor-savjeta1"><a href="http://studentski.hr/"><b>{{advice.izvor1}}</b></a></div>
      </div>

      <div class="naslov-savjeta1" [navPush]="advice1Page">{{advice.title1}}</div>

      <div class="container3">
          <div class="datum-savjeta">{{advice.publishedAt}}</div>
          <div class="izvor-savjeta">{{advice.source}}</div>
          <div class="izvor-savjeta1"><a href="https://www.unin.hr/"><b>{{advice.izvor}}</b></a></div>
      </div>
    </ion-item>
  </ion-list>
</ion-content>
```

Slika 6.18 lifestyle.html

```
.naslov-savjeta1 {
    font-family: "Roboto-Black";
        color: #474747;
        font-size: 12pt;
        white-space: normal;
    }
.naslov-savjeta {
    font-family: "Roboto-Black";
        color: #474747;
        font-size: 11pt;
        height: auto;
        white-space: normal;
        display: -webkit-inline-box;
        position: absolute;
        padding-left: 10px;
    }
.datum-savjeta {
    font-family: "Raleway-Medium";
        color: #828282;
        font-size: 8pt;
        display: inline-block;
    }
.datum-savjeta1 {
    font-family: "Raleway-Medium";
    color: #828282;
    font-size: 8pt;
    display: flex;
    }
.izvor-savjeta {
    font-family: "Raleway-Medium";
    color: #828282;
    font-size: 8pt;
    padding-left: 10px;
    display: inline-block;
    }
.izvor-savjeta1 {
    font-family: "Raleway-Medium";
    color: #828282;
    font-size: 7pt;
    padding-left: 5px;
    display: flex;
    }
a {
    color: #219653;
}
a:link {
    text-decoration: none;
    }
.article-pic1 {
        padding-bottom: 5px;
    }
.article-pic2 {
    width: 65px;
    height: 65px;
    }
.container3 {
    display: inline-flex;
    position: absolute;
    padding-left: 75px;
    margin-top: -25px;
    }
```

Slika 6.19 lifestyle.scss

6.2.3. Stranica „Događaji“

Prema stvorenom predlošku aplikacije, ova stranica ima drugačiji izgled i raspored elemenata. Kao što sam naziv govori, stranica je osmišljena kako bi sadržavala razne događaje (koncerte, kulturna događanja, sportske aktivnosti itd.) u blizini i okolici. Za primjer je uzeto nekoliko događanja u ovom dijelu regije. Isto kao i za prethodno navedene stranice, izrađuje se .json datoteka s potrebnim podatcima koji će se ispisati na ekranu pametnog telefona (naziv događaja, URL slike koja predstavlja isti, dan, mjesec i vrijeme događanja, lokacija). Generirane podatke potrebno je urediti i razmjestiti po stranici. Uz podatke na stranici, ponovno je korištena komponenta Ionicons kako bi se stranica upotpunila i poprimila zanimljiv izgled.


```
[
    {
    "title1": "Zagreb Beer Fest 2018.",
    "urlToImage": "assets/imgs/picture11.png",
    "day": "17",
    "month": "SVI",
    "time": "17. svibnja u 16:00",
    "location": "Trg dr. Franje Tuđmana"
    },
    {
    "title2": "Beerfest Sjever VII.",
    "urlToImage": "assets/imgs/picture12.png",
    "day": "25",
    "month": "SVI",
    "time": "25. svibnja u 15:00",
    "location": "Stari Hrast"
    },
    {
    "title3": "INmusic festival #13",
    "urlToImage": "assets/imgs/picture13.png",
    "day": "25",
    "month": "LIP",
    "time": "25. lipnja u 16:30",
    "location": "Otok hrvatske mladeži"
    }
]
```


Slika 6.20 .json datoteka za stranicu "Događaji"

```
<ion-content padding class="ion-content">
  <div class="status2"></div>
  <ion-list>
    <ion-item *ngFor="let event of events">
      <img src="{{event.urlToImage}}"/>

      <div class="container4">
      <div class="datum-dogadaja" id="dan">{{event.day}}</div><br>
      <div class="datum-dogadaja">{{event.month}}</div>
      </div>

      <div class="container">
        <div class="naslov-dogadaja" [navPush]="event1Page">{{event.title1}}</div>
        <div class="naslov-dogadaja" [navPush]="event2Page">{{event.title2}}</div>
        <div class="naslov-dogadaja" [navPush]="event3Page">{{event.title3}}</div>

        <div class="vrijeme-dogadaja">
          <ion-icon name="time" md="ios-time-outline"></ion-icon>
          {{event.time}}
        </div>

        <div class="lokacija-dogadaja">
          <ion-icon name="pin" md="ios-pin-outline"></ion-icon>
          {{event.location}}
        </div>
      </div>
    </ion-item>
  </ion-list>

</ion-content>
```

Slika 6.21 dogadaji.html

```
.naslov-dogadaja {
    font-family: "Roboto-Black";
    color: #474747;
    font-size: 12pt;
}
.datum-dogadaja {
    font-family: "Roboto-Black";
    color: #2C6FC9;
    font-size: 16pt;
    float: left;
}
#dan {
    color: #474747;
    margin-left: 5px;
}
.vrijeme-dogadaja {
    font-family: "OpenSans-Regular";
    color: #474747;
    font-size: 10pt;
    padding-bottom: 3px;
    margin-top: 5px;
}
.lokacija-dogadaja {
    font-family: "OpenSans-Regular";
    color: #474747;
    font-size: 10pt;
}
```

```
.container {
    float: left;
    margin-left: 20px;
    margin-top: 10px;
}
.container4 {
    float: left;
    margin-left: 15px;
    margin-top: 15px;
    line-height: 17pt;
}
```

Slika 6.22 dogadaji.scss

Pojedinačne stranice događaja imaju istu strukturu kao i pojedinačni članci, međutim sadrže dodatne elemente, potrebne kako bi se upotpunile potrebne informacije o nekom događaju. Pri kraju svake stranice s događajem, osim gumbova za dijeljenje, implementirana je komponenta Google Maps. Kako se hibridna mobilna aplikacija izvodi kao web stranica u pregledniku, s dodatnim omotačima koji joj daju izvorni (nativni) izgled i ponašanje, za dohvaćanje lokacije uređaja korisnika koristi se HTML geolokacijski API. HTML geolokacijski API, osim GPS podatcima uređaja, ima mogućnost pristupa i drugim API-jima za dohvaćanje lokacije korisnika. Ukoliko GPS podatci nisu dostupni, preglednik može koristiti Google Maps API [20], koji je korišten prilikom izrade ove mobilne aplikacije.

Da bi se Google Maps API mogao koristiti, potrebno je prijaviti se s odgovarajućim Google korisničkim imenom i lozinkom i kreirati API ključ. On se uključuje u glavnu *index.html* datoteku u mapi *src*. Nije bilo dosta samo umetnuti kartu na stranicu, već dodati marker koji pokazuje točnu lokaciju događaja. Za dodavanje markera, potrebno je upisati koordinate na kojima se želi prikazati. Koordinate su dobivene očitavanjem URL-a od željenog mjesta na Google Maps-u u internetskom pregledniku.

```
export class GoogleMapComponent {

  @ViewChild("map") mapElement;
  map: any;
  constructor() {

  }

  ngOnInit(){
    this.initMap();
  }

  initMap(){

    let coords = new google.maps.LatLng(45.8118271,15.9555403)
    let mapOptions: google.maps.MapOptions = {
      center: coords,
      zoom: 15,
      mapTypeId: google.maps.MapTypeId.ROADMAP
    }

    this.map = new google.maps.Map(this.mapElement.nativeElement,
    mapOptions)

    let marker: google.maps.Marker = new google.maps.Marker({
      map: this.map,
      position: coords
    })

  }
}
```

Slika 6.23 primjer koda za lokaciju događaja

6.2.4. Stranica „Ponude“

Ova stranica sadržavala bi korisne stvari za studente u kojima bi se pronašle ponude za studentski posao, smještaj te instrukcije. Struktura stranice i raspored elemenata osmišljen je tako da za svaku ponudu ili oglas sa strane stoje najbitnije informacije, tako da se pojedinačna stranica može, ali ne mora posebno otvarati. Napravljena je .json datoteka s potrebnim podatcima koji će se ispisivati (naslov ponude, URL slike, cijena usluge, lokacija i podatci za kontakt – broj mobilnog telefona). Pojedinačne stranice, uz glavne, sadrže dodatne i opširnije informacije o objavljenom oglasu ili ponudi.


```
[
    {
    "title1": "Tražimo instruktore za online instrukcije",
    "urlToImage": "assets/imgs/picture14.jpg",
    "priceTag": "800 kn",
    "location": "Varaždin",
    "phone": "+385 99 192 1996"
    },
    {
    "title2": "Studentica za rad u dućanu",
    "urlToImage": "assets/imgs/picture15.jpg",
    "priceTag": "20 kn/h",
    "location": "Varaždin",
    "phone": "+385 98 320 226"
    },
    {
    "title3": "Tražimo Sign Spinnere M/Ž",
    "urlToImage": "assets/imgs/picture16.jpg",
    "priceTag": "30 kn/h",
    "location": "Zagreb",
    "phone": "+385 91 603 4040"
    },
    {
    "title4": "Instrukcije iz gitare/pomoć pri sviranju",
    "urlToImage": "assets/imgs/picture17.jpg",
    "priceTag": "50 kn/h",
    "location": "Varaždin",
    "phone": "+385 91 983 9198"
    },
```


Slika 6.24 dio .json datoteke za stranicu "Ponude"

```
<ion-content padding class="ion-content">
  <div class="status3"></div>
  <ion-list>
    <ion-item *ngFor="let offer of offers">
      <img src="{{offer.urlToImage}}" class="img"/>
      <div class="naslov-ponude" [navPush]="offer1Page">{{offer.title1}}</div>
      <div class="naslov-ponude" [navPush]="offer2Page">{{offer.title2}}</div>
      <div class="naslov-ponude" [navPush]="offer3Page">{{offer.title3}}</div>
      <div class="naslov-ponude" [navPush]="offer4Page">{{offer.title4}}</div>
      <div class="naslov-ponude" [navPush]="offer5Page">{{offer.title5}}</div>
      <div class="naslov-ponude" [navPush]="offer6Page">{{offer.title6}}</div>
      <div class="cijena-ponude">
        <ion-icon name="pricetag" md="ios-pricetag-outline"></ion-icon>
        {{offer.priceTag}}
      </div>

      <div class="container">
      <div class="lokacija-ponude">
        <ion-icon name="pin" md="ios-pin-outline"></ion-icon>
        {{offer.location}}
      </div>
      <div class="kontakt-ponude">
        <ion-icon name="call" md="ios-call-outline"></ion-icon>
        {{offer.phone}}
      </div>
      </div>

    </ion-item>
  </ion-list>
</ion-content>
```

Slika 6.25 ponude.html

```
.naslov-ponude {
    font-family: "Roboto-Black";
    color: #474747;
    font-size: 12pt;
    height: auto;
    white-space: normal;
    display: -webkit-inline-box;
    position: absolute;
    padding-left: 10px;
}
.cijena-ponude {
    font-family: "Raleway-Bold";
    color: #474747;
    font-size: 11pt;
    display: inline-block;
    margin-left: 9px;
}
.lokacija-ponude {
    font-family: "Raleway-Medium";
    color: #474747;
    font-size: 9pt;
    margin-left: 78px;
}
.kontakt-ponude {
    font-family: "Raleway-Medium";
    color: #474747;
    font-size: 10pt;
    padding-left: 20px;
    padding-top: 2px;
}
```

Slika 6.26 ponude.scs

6.3. Testiranje aplikacije

Izrađena aplikacija uglavnom je testirana na Google Chrome pregledniku zbog kratkog vremena izgradnje, što je praktičnije tijekom izrade same aplikacije. Osim Chrome-a, završno testiranje sam provela na vlastitom mobilnom Android uređaju (verzija sustava 8.0.0) kako bih vidjela konačni produkt i provjerila ima li grešaka ili nedostataka. Da bi se aplikacija mogla razviti, na mobilnom uređaju je bilo potrebno omogućiti razvojne opcije (*engl. Developer Mode*) i uključiti uklanjanje programskih pogrešaka na USB-u (*engl. USB debugging*) te unijeti odgovarajuću komandnu liniju:

```
ionic cordova run android –device
```

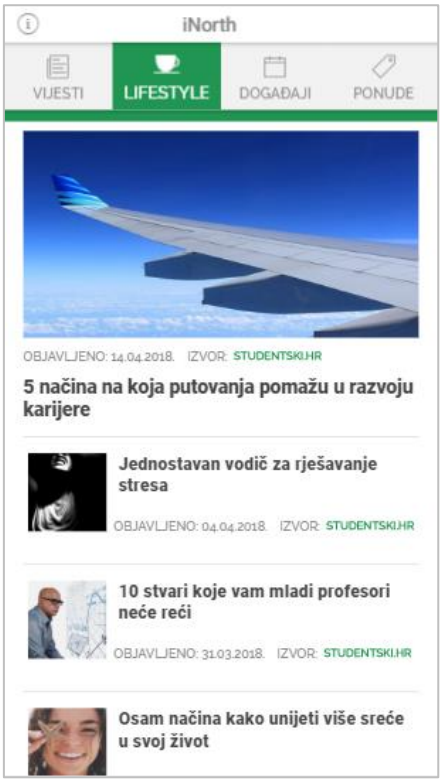
Ovaj način testiranja aplikacije je dugotrajan i otežan, stoga nije korišten često jer bi se postupak izgradnje trebao ponavljati nakon svake promjene koda.

Aplikacija se nalazi na: http://arwen.unin.hr/~isinkovic/iNorth-tabs/www/

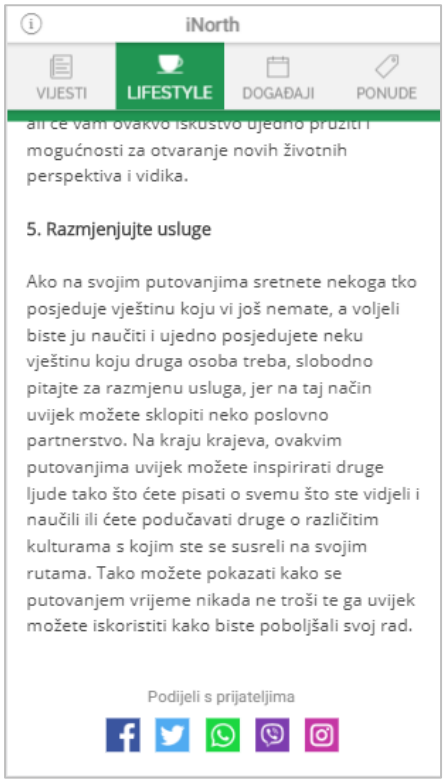
vijesti.html

pojedinačna stranica vijesti

lifestyle.html



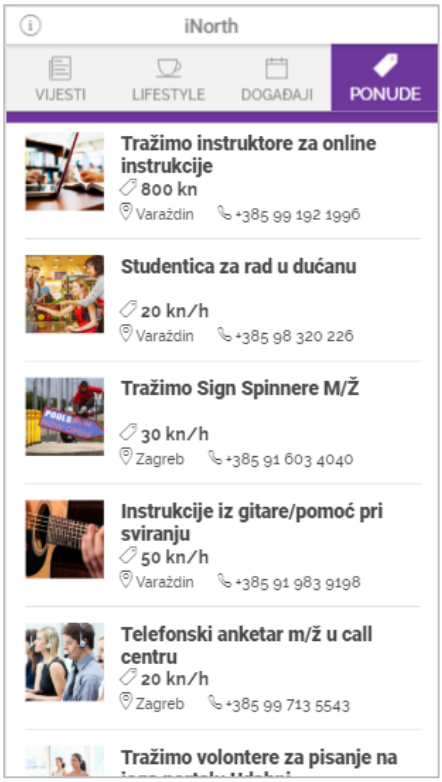
pojedinačna stranica lifestyle-a



dogadaji.html

pojedinačna stranica događaja

ponude.html



pojedinačna stranica ponuda

impressum.html

7. Zaključak

Uobičajenim načinom razvoja izvornih (nativnih) aplikacija potrebno je poznavanje više programskih jezika i tehnologija, a samim time i više sredstava. Prednost hibridnih mobilnih aplikacija nad izvornima je uporaba istog programskog koda, zasnovanog na uglavnom jednostavnijim web tehnologijama s pristupom nativnim funkcionalnostima uređaja, da bi se razvila aplikacija koja radi kao izvorna na više platformi. Razvoj višeplatformskih aplikacija za jednostavnije potrebe svakako može biti ravnopravna zamjena izvornim aplikacijama. S druge strane, takve aplikacije često imaju niže performanse i određena ograničenja u pristupu hardveru pa se pri kompleksnijim zahtjevima koriste nativne aplikacije zbog svoje brzine izvođenja i pristupa svim izvornim mogućnostima koje pruža mobilni uređaj.

Za razvoj mobilnog dijela rada korišten je Ionic okvir koji je i inače često korišten za razvoj hibridnih mobilnih aplikacija. Ionic je jednostavan za upotrebu, pošto je dostupno mnogo gotovih komponenti, a integracija sa vanjskim API-jem na kojeg se veže i dohvaća podatke nije kompliciranа. Ionic ne zahtjeva veliki tim i veliku cijenu troška rada te se ne javljaju problemi oko odabira mobilne platforme. Za rad s Ionic-om dovoljan je samo jedan programer koji sam odlučuje koje će mobilne platforme biti podržane u aplikaciji. Bitno je i napomenuti okvire s kojima Ionic surađuje – Cordova i AngularJS, bez čijih biblioteka i principa rada nije moguća izrada aplikacije. Važna stavka u izradi aplikacije je i izrada njenog vizualnog identiteta i korisničkog sučelja. Korisničko sučelje aplikacije „iNorth“ osmišljeno je pomoću alata Figma te je dizajnirano prema standardima za omogućavanje dobrog korisničkog iskustva. Aplikacija je osmišljena kao mjesto gdje se nalaze sve potrebne stvari za studente – vijesti sa Sveučilišta, vijesti i objave zabavnog karaktera, događanja u blizini i okolici te raznovrsne ponude i oglasi. Za generiranje tih stranica koristio se JSON, u kojem su ispisani svi potrebni podaci za prikaz članaka/objava, te odgovarajući provider-i. Na pojedinačnim stranicama korištena je i biblioteka social-sharing da bi se omogućilo dijeljenje sadržaja na društvenim mrežama. Važna stavka na stranici s događajima je i umetnuta komponenta Google Maps, čija je implementacija dobivena pomoću Google Maps API-a, odnosno generiranog ključa. U aplikaciji je korištena i komponenta Ionicons, kako bi se upotpunio izgled aplikacije te čiji je set ikona i njihova uporaba opisana na stranici Ionic web mjesta. Na kraju je važno napomenuti da praktični dio ovog završnog rada, iako je funkcionalan, sadrži osnove sa kojima se može prikazati demonstracija tehnologija, ali tako postoji i mogućnost za daljnji razvoj i širi broj dodatnih mogućnosti koje mogu biti integrirane, kao što je back-end ili baza aplikacije.

Iako se trendovi danas jako brzo mijenjaju i tehnologija neprestano napreduje, očekujem da će mobilne aplikacije i dalje biti jedne od najpopularnijih te da će se i dalje razvijati na svim operativnim sustavima. Također, očekujem daljnji rast popularnosti hibridnih aplikacija, jer su te tehnologije još relativno nove i nedovoljno razvijene, te osmišljavanje novih univerzalnih alata za izradu mobilnih aplikacija.

U Varaždinu, _____ _____

Iva Sinković

8. Literatura

[1] https://www.vecernji.hr/techsci/povijest-aplikacija-za-pametne-telefone-od-prvih- mobitela-do-danas-1025201, dostupno 20.08.2018.

[2] https://www.techopedia.com/definition/27568/native-mobile-app, dostupno 20.08.2018.

[3] http://www.digital-dividend.com/en/native-or-hybrid-the-difference/, dostupno 20.08.2018.

[4] https://www.maxcdn.com/one/visual-glossary/web-application/, dostupno 20.08.2018.

[5] https://www.webprogramiranje.org/uvod-u-hibridne-mobilne-aplikacije/#Cordova, dostupno 20.08.2018.

[6] http://freezeprosoftware.com/news/advantages-and-disadvantages-of-using-cross-p.aspx#.W4kyj-gzbIU, dostupno 20.08.2018.

[7] https://ionicframework.com/docs/v1/guide/preface.html, dostupno 20.08.2018.

[8] https://ionicframework.com/about, dostupno 20.08.2018.

[9] https://ionicframework.com/docs/components/, dostupno 20.08.2018.

[10] https://www.businessnewsdaily.com/4992-what-is-baas.html, dostupno 20.08.2018.

[11] https://nodejs.org/en/, dostupno 20.08.2018.

[12] https://angular.io/guide/architecture, dostupno 20.08.2018.

[13] https://www.typescriptlang.org/, dostupno 20.08.2018.

[14] https://cordova.apache.org/docs/en/latest/guide/overview/index.html, dostupno 20.08.2018.

[15] https://code.visualstudio.com/, dostupno 20.08.2018.

[16] https://ionicframework.com/docs/ionicons/, dostupno 20.08.2018.

[17] https://docs.angularjs.org/guide/di, dostupno 20.08.2018.

[18] https://ionicframework.com/docs/native/social-sharing/, dostupno 20.08.2018.

[19] http://studentski.hr/, dostupno 20.08.2018.

[20] https://developers.google.com/maps/documentation/geolocation/get-api-key, dostupno 20.08.2018.

9. Popis slika

Slika 1.1 Ionic Framework Izvor: https://ionicframework.com/press 2
Slika 2.1 Nativne, hibridne i web aplikacije Izvor: http://www.digital-dividend.com/en/native-or-hybrid-the-difference/ 7
Slika 3.1 Prikaz strukture Ionic aplikacije Izvor: vlastita slika 14
Slika 6.1 Ikona aplikacije Izvor: vlastita slika 19
Slika 6.2 Implementacija ikone u aplikaciju Izvor: vlastita slika 19
Slika 6.3 Skice osnovnih stranica aplikacije Izvor: vlastita slika 20
Slika 6.4 Korištena paleta boja Izvor: vlastita slika 20
Slika 6.5 Korišteni fontovi Izvor: vlastita slika 20
Slika 6.6 Primjer Ionicons dokumentacije Izvor: vlastita slika 21
Slika 6.7 Definiranje stranica u tabs.ts datoteci Izvor: vlastita slika 22
Slika 6.8 Deklaracija u app.module.ts datoteci Izvor: vlastita slika 22
Slika 6.9 tabs.html Izvor: vlastita slika 22
Slika 6.10 .json datoteka za stranicu "Vijesti" Izvor: vlastita slika 23
Slika 6.11 provider article-service Izvor: vlastita slika 24
Slika 6.12 vijesti.ts Izvor: vlastita slika 25
Slika 6.13 vijesti.html i *ngFor direktiva Izvor: vlastita slika 26
Slika 6.14 vijesti.scss Izvor: vlastita slika 26
Slika 6.15 dio koda HTML stranice članka – social-sharing Izvor: vlastita slika 27
Slika 6.16 dio TypeScript koda – primjer za društvenu mrežu WhatsApp Izvor: vlastita slika 27
Slika 6.17 .json datoteka za stranicu "Lifestyle" Izvor: vlastita slika 28
Slika 6.18 lifestyle.html Izvor: vlastita slika 29
Slika 6.19 lifestyle.scss Izvor: vlastita slika 29
Slika 6.20 .json datoteka za stranicu "Događaji" Izvor: vlastita slika 30
Slika 6.21 dogadaji.html Izvor: vlastita slika 31
Slika 6.22 dogadaji.scss Izvor: vlastita slika 31
Slika 6.23 primjer koda za lokaciju događaja Izvor: vlastita slika 32
Slika 6.24 dio .json datoteke za stranicu "Ponude" Izvor: vlastita slika 33
Slika 6.25 ponude.html Izvor: vlastita slika 34
Slika 6.26 ponude.scs Izvor: vlastita slika 34

Sveučilište
Sjever

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Iva Sinković *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)
Iva Sinković
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Iva Sinković *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom Izrada hibridne mobilne aplikacije korištenjem Ionic razvojnog okvira *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)
Iva Sinković
(vlastoručni potpis)