Android aplikacija za organizaciju obavijesti na mobitelu

Mihić, Vlatka

Undergraduate thesis / Završni 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:200:732054

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-09-14**

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 računarstva

Android aplikacija za organizaciju obavijesti na mobitelu

Završni rad

Vlatka Mihić

Osijek, 2021. godina.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 15.09.2021.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime studenta:	Vlatka Mihić
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R4247, 26.07.2018.
OIB studenta:	52915400718
Mentor:	Doc.dr.sc. Tomislav Rudec
Sumentor:	Dr. sc. Krešimir Romić
Sumentor iz tvrtke:	Goran Luketić
Naslov završnog rada:	Android aplikacija za organizaciju obavijesti na mobitelu
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 2 bod/boda Jasnoća pismenog izražavanja: 2 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	15.09.2021.
Datum potvrde ocjene Odbora:	22.09.2021.

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

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

IZJAVA O ORIGINALNOSTI RADA

Osijek, 26.09.2021.

Ime i prezime studenta:	Vlatka Mihić
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	R4247, 26.07.2018.
Turnitin podudaranje [%]:	12

Ovom izjavom izjavljujem da je rad pod nazivom**: Android aplikacija za organizaciju obavijesti na mobitelu**

izrađen pod vodstvom mentora Doc.dr.sc. Tomislav Rudec

i sumentora Dr. sc. Krešimir Romić

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

SADRŽAJ

1. UVOD 1

1.1. Zadatak završnog rada 1

2. KRATAK POVIJESNI I AKTUALNI PREGLED ALATA ZA ORGANIZACIJU SVAKODNEVICE 2

3. POSTUPAK IZRADE APLIKACIJE 3

4. ATHINA – ANDROID APLIKACIJA ZA POMOĆ PRI ORGANIZACIJI 4

4.1. Projektna ideja 4

4.2. Izrada dizajna 5

4.2.1. Figma 5

4.2.2. Dizajn mobilnih aplikacija 5

4.2.3. Dizajn aplikacije Athina 6

4.3. Pretvaranje slike u kod 11

4.3.1. XML – izrada rasporeda 12

4.3.2. Navigacija – tok aplikacije 15

4.3.3. Spremanje podataka pomoću baze podataka 16

4.3.4. Obavijesti 18

4.3.5. Primjena aplikacije Athina 21

5. ZAKLJUČAK 24

LITERATURA 25

SAŽETAK 26

ABSTRACT 27

ŽIVOTOPIS 28

PRILOZI 29

1. UVOD

Svi su se jednom pronašli u situaciji da ne znaju gdje su stali. „Jesam li naručen na pregled u 11 sati ili 12?“, „Imamo li dovoljno rajčice doma ili bih mogao kupiti?, samo su neka od često postavljenih pitanja. Današnji svijet je sve brži i brži te je nedovoljno vremena stati i dobro zapamtiti sve što treba obaviti svakoga dana. Upravo iz tog razloga mnogi trebaju oslonac i pomoć u organizaciji svakodnevice. Tehnologija je dovoljno napredovala i napreduje ka razvitku alata koji će pomoći ljudima pri obavljanju svakodnevnih poslova, ali osim u tu svrhu potrebni su i alati u koje će se moći pohraniti sve što treba odraditi u danu. Ovaj rad bavi se upravo ovim problemom i opisuje postupak izrade aplikacije koji može poslužiti u organizaciji svakodnevice. Dotaknuti će se i organizacije izrade aplikacije gdje će biti opisani koraci po kojima se rade projekti u većini današnjih poduzeća. Svaki od koraka bit će detaljno opisan na primjeru aplikacije koja je izrađena u navedenu svrhu.

Drugim poglavljem opisuje se povijesni razvoj te aktualni pregled alata za organizaciju svakodnevice koji su potkrijepljeni referencama na izvore korištene u razradi ideje završnoga rada.

Kroz jasno navedene korake u trećem poglavlju opisuje se postupak izrade bilo kakve Android aplikacije s mogućnošću primjene i na druge vrste aplikacija.

U četvrtome poglavlju detaljnije su opisani i pojašnjeni koraci definirani u prijašnjem poglavlju na primjeru aplikacije pod nazivom Athina. Osim pojašnjenja koraka iz prijašnjeg poglavlja u ovome poglavlju opisane su i neke značajke aplikacije Athina te alati koji su korišteni u svrhu izrade ove aplikacije.

1.1. Zadatak završnog rada

Zadatak ovoga završnoga rada jest opisati postupak izrade aplikacije po kojemu je većina aplikacija u Android okruženju koje koristimo izrađena te izraditi aplikaciju za organizaciju obavijesti pod nazivom Athina.

2. KRATAK POVIJESNI I AKTUALNI PREGLED ALATA ZA ORGANIZACIJU SVAKODNEVICE

Povijest planera počinje prije više od dva stoljeća i još uvijek je iznimno rasprostranjena u današnjem svijetu o čemu se govori i u debitanskoj tezi brenda Lala Letter iz 2018. godine. Planeri su izvorno bili časopisi u koje se bilježila prošlost, tek kasnije je glavna funkcija planera postalo planiranje budućnosti. Planer kakvog danas poznajemo datira iz 1773. godine, kada je Aitken (Robert Aitken, 1735-1802, filadelfijski pisač, prodavač knjiga i izdavač) stvorio prvi osmišljeni dnevni planer za korisnike koji planiraju svoju budućnost, nazvan Aitken Register [1].

Digitalno doba došlo je kao plimni val, uništavajući gotovo sve intenzivne dužnosti papira koje ljudi svakodnevno koriste.

1975. godine indijski poduzetnik Satyan Pitroda izumio je elektronički dnevnik ili organizator što se može pronaći na njegovoj web stranici iz 2020. godine [2]. To je malo računalo veličine kalkulatora, često s ugrađenom aplikacijom dnevnika i drugim funkcijama, poput adresara i kalendara.

20. stoljeće donosi nagli razvoj tehnologije koja uključuje pametne telefone zajedno s mnogim aplikacijama koje se počinju razvijati. Uz planer aplikacije razvijaju se i takozvani podsjetnici. Podsjetnici su vrlo korisni za osobe koje lako zaboravljaju kako bi ih podsjetili na uzimanje određenih terapija i drugih dnevnih ili tjednih obveza što se spomenulo u nedavnoj studiji pod nazivom „*Intention to Use a Drug Reminder App: A Case Study of Diabetics and High Blood Pressure Patients“* iz 2020. godine [3]. Kako danas ovakvih aplikacija ima mnogo teško je odabrati onu koja će nama odgovarati, a još teže je izraditi novu aplikaciju ove vrste koja bi donijela nešto što dosadašnje aplikacije nemaju. Najnovija istraživanja Google korporacije koja datiraju iz 2016. godine pokazaju da su jednostavne, male, besplatne i aplikacije usredotočene na specifične radnje najpoželjnije [4]. Neke od takvih aplikacija koje se mogu pronaći u mobilnim trgovinama su: Serene, Todoist, Things 3, Any.do, Trello [5].

3. POSTUPAK IZRADE APLIKACIJE

Jedna od početnih i najbitnijih stvari koju je potrebno učiniti prije no što se krene s poslom jest organizacija. Kako će se nešto odraditi je jedan od ključnih dijelova bilo kojeg problema. Uvijek se kreće od ideje ili zahtjeva. Što je ona bolje i jasnije opisana to je lakše krenuti dalje. Naiđe li se u samoj ideji na poteškoće kao što su nedovoljna istraženost ili neizvedivost nekih značajki ne može se krenuti na sljedeći korak. Sama ideja iziskuje mnogo truda i vremena te upravo ona može uštedjeti na vremenu kod praktičnog dijela izrade aplikacije. Neka pitanja na koja treba odgovoriti kod razrade ideje su sljedeća [6]:

- "Koja je svrha aplikacije koja se izrađuje?"
- "Koji su ciljani korisnici aplikacije?"
- "Koji su mogući rizici do kojih se može doći pri izradi aplikacije?"
- "Koji su koraci izrade koji se mogu primijeniti u aplikaciji?"
- "Koji su alati potrebni za realizaciju?"
- "Koja su ograničenja u izradi aplikacije?"

Što se više pitanja postavi i odgovora pronađe u ovoj fazi time se manje vremena troši na neočekivana istraživanja i probleme koji mogu nastati pri izradi.
Kada je ideja razrađena i svi odgovori pronađeni može se krenuti s izradom aplikacije. Prvo što dolazi jest dizajn aplikacije koji uključuje tipografiju, boje, ikone i elemente kao što su gumbi, meniji, polja za unos i drugo. Kada su navedeni dijelovi gotovi sklapaju se u stilski vodič (engl. *style guide*). On je smjernica pri izradi dizajna koja održava konzistentnost aplikacije.

Pri završetku dizajna prelazi se na razvoj aplikacije. Dizajn je kalup po kojemu se razvija aplikacija. U ovome radu pri razvoju aplikacije korišten je Android Studio koji će biti nešto detaljnije opisan. U ovom završnom koraku se primjenjuju i koriste svi ranije navedeni koraci čijim je završetkom gotova i sama izrada aplikacije.

4. ATHINA – ANDROID APLIKACIJA ZA POMOĆ PRI ORGANIZACIJI

4.1. Projektna ideja

Projektna ideja jest postupak dokumentiranja problema, rješenja i dolaska do željenog rješenja pri izradi aplikacije. Sastoji se od nekoliko ključnih koraka [7] kojima se treba voditi, a to su:

1. Definiranje problema/ideje.
2. Potraga i predstavljanje rješenja.
3. Definiranje rezultata i kriterija uspjeha.
4. Prezentiranje ili dokumentacija plana, odnosno pristupa.

Osim ovih koraka u projektnoj ideji mogu se nalaziti i koraci poput navođenja rasporeda projekta i proračuna te pisanje prijedloga.

Početna ideja aplikacije Athina jest izrada aplikacije koja će pomoći korisnicima u boljoj organizaciji svojih planova i svakidašnjice. Kako bi se to postiglo uzeta su u obzir sljedeća rješenja:

- planer
- izrada podsjetnika/obavijesti
- upoznavanje sebe kroz opisivanje

Planer služi za bilježenje kratkoročnih i dugoročnih ciljeva, odnosno planova koje si korisnik postavi. Glavne funkcionalnosti ove značajke su stvaranje i brisanje planova. Izrada obavijesti iliti podsjetnika treba služiti kao popis stvari ili događaja kojih se treba sjetiti ili obaviti. Najvažnije kod podsjetnika [8] jest vrijeme kada se treba prisjetiti na nešto prilikom kojeg će nas aplikacija obavijestiti o tome. Stoga se pri izradi obavijesti mora dodati i vrijeme obavijesti. Kao dodatna značajka aplikacija će imati poseban dio gdje će korisnik dodati svoje ime, hobije, osobine i stvari koje voli kao što su najdraže knjige, pjesme ili autori. Svaka od značajki bit će povezana na bazu podataka gdje će se svi podaci spremati i dohvaćati. Ovako definirana rješenja su ujedno i kriteriji uspjeha kojima se teži u izradi.

U iznošenju plana ili pristupa potrebno je navesti vremenski rok izrade s definiranim fazama i određenim resursima [9].

Alati i resursi koji su se koristili u izradi aplikacije Athina jesu Figma i Android Studio te neki vanjski resursi poput besplatnih fotografija koje su preuzete s internetskih pretraživača.

4.2. Izrada dizajna

Korak koji slijedi nakon razrade projektne ideje jest dizajn. Kada kažemo dizajn zapravo mislimo na grafički dizajn aplikacije koju izrađujemo. Općenito pojam dizajn označava stvaranje, oblikovanje, izvršavanje ili konstruiranje nečega prema određenome planu. Grafički dizajn aplikacije uključuje izradu izgleda aplikacije, ali i određivanje toka aplikacije. U nastavku će biti opisan jedan od poznatih besplatnih alata koji se koristi pri izradi dizajna.

4.2.1. Figma

Figma [10] je široko rasprostranjen alat za uređivanje i izradu prototipa raznih vrsta aplikacija. Ove ideje mrežno osnovanog dizajn alata dosjetili su se Dylan Field i Evan Wallace kao studenti Sveučilišta Brown u Providenceu 2011. godine. Na ideji su krenuli raditi 2012. godine te je Figma četiri godine kasnije prvi puta objavljena 27. rujna 2016. godine. Prvenstveno je zasnovana na mreži dok su dodatne izvanmrežne značajke omogućili računalni programi na mac OS i Windows platformama. Glavna funkcionalnost, uvedena 22. kolovoza 2019. godine, uz izradu dizajna jest mogućnost suradnje više osoba u stvarnom vremenu. Ovom funkcionalnošću je pristup dizajnu omogućen ne samo dizajnerima nego i cijelome timu koji radi na projektu.

4.2.2. Dizajn mobilnih aplikacija

Dizajn mobilnih aplikacija najčešće uključuje dizajn korisničkog iskustva (engl. *user experience design*) i dizajn korisničkog sučelja (engl. *user interface design*). Dizajn korisničkog sučelja jest postupak koji dizajneri koriste za izgradnju sučelja pri čemu je naglasak na izgledu ili stilu sučelja. Ovdje dizajneri izrađuju komponente, određuju tipografiju, ikone i druge elemente kojima stvaraju određeni brend aplikacije. Osnovni elementi dizajna korisničkog sučelja, ali ne i jedini, su:

- kontrole unosa: gumbi (engl. *buttons*), tekstualna polja, potvrdni okviri, padajući izbornici, okviri s popisima, preklopnici (engl. *toggles*) i drugi okviri za unos podataka
- navigacijske komponente: klizač, polje za pretraživanje, paginacija, oznake i ikone
- informacijske komponente: opis (engl. *tooltip*), ikone, traka napretka, obavijesti, okviri za poruke, modalni prozori

Kod dizajna korisničkog sučelja osim što je bitno znati koje elemente koristiti, bitno je i paziti na to kako ih koristimo [11]. Kako bi se napravilo što bolje korisničko sučelje bitno je poznavati buduće korisnike i njihove ciljeve, sposobnosti, preference i tendencije. Neke od boljih praksi kod izrade korisničkog sučelja su sljedeće [12]:

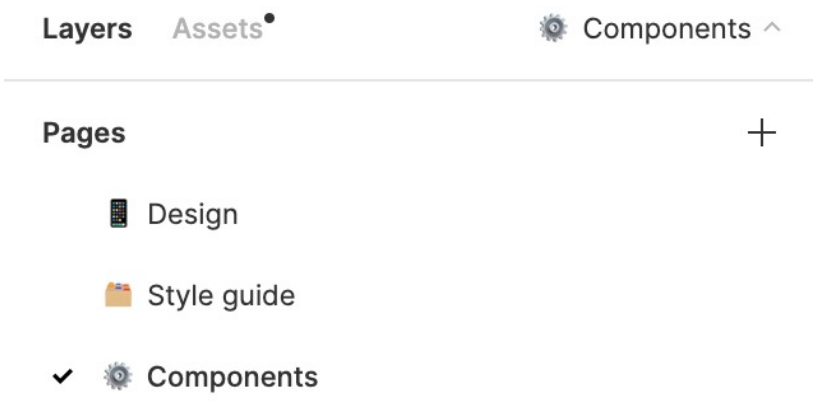
- sučelje treba biti što jednostavnije, bez nepotrebnih elemenata, čitko i jasno

- cilj je stvoriti dosljednost i koristiti uobičajene elemente korisničkog sučelja kako bi se korisnici osjećali ugodnije te kako bi mogli brže obaviti stvari
- važno je gdje postavljamo elemente na stranici kako bi se pozornost usmjerila na najvažnije dijelove što pomaže u skeniranju i čitljivosti
- strateški koristiti boje i teksturu pomoću kojih možemo usmjeriti ili preusmjeriti boju s predmeta
- upotreba tipografije neka služi za stvaranje hijerarhije i jasnoće
- obratiti pozornost na obavijesti koje korisnik primi od aplikacije, odnosno paziti da sustav što bolje komunicira s korisnikom

Dizajn korisničkog iskustva jest postupak koji dizajneri koriste za stvaranje proizvoda koji korisnicima pružaju smislena i relevantna iskustva. Ovaj proces uključuje aspekte brendiranja te dizajn upotrebljivosti i funkcije. Brendiranjem neki proizvod ili usluga postaju jedinstveni i prepoznatljivi te im se time ujedno i vrijednost na tržištu povećava. Što je bolje napravljen dizajn korisničkog sučelja time će i dizajn korisničkog iskustva biti dobar. Smjernice koje su navede pri dizajniranju korisničkog sučelja vrijede i za dizajn korisničkog iskustva. Najbitnije osobine aplikacije jesu jednostavnost, smislenost, pristupačnost, preglednost i konzistentnost.

4.2.3. Dizajn aplikacije Athina

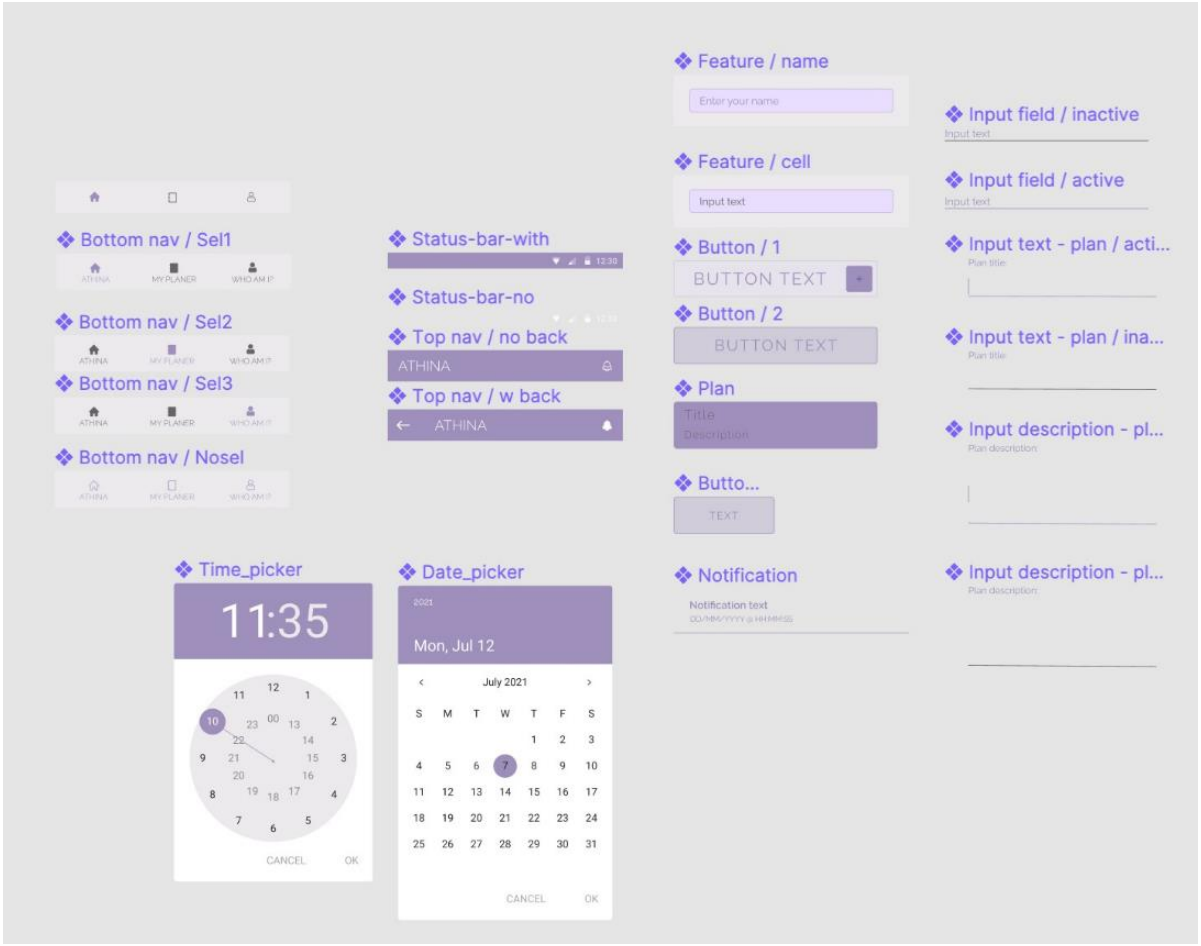
Za dizajn aplikacije Athina korišten je alat Figma koji je nešto ranije opisan. Slika 3.1 prikazuje takozvane stranice projekta Athina. One uključuju komponente (engl. *Components*), stilski vodič (engl. *Style Guide*) i dizajn (engl. *Design*). Započinje se s komponentama. Komponente su elementi u alatu Figma koji se mogu lako dohvatiti iz ploče resursa (engl. *asset panel*). Takozvane



Slika 3.1 Stranice projekta Athina u alatu Figma

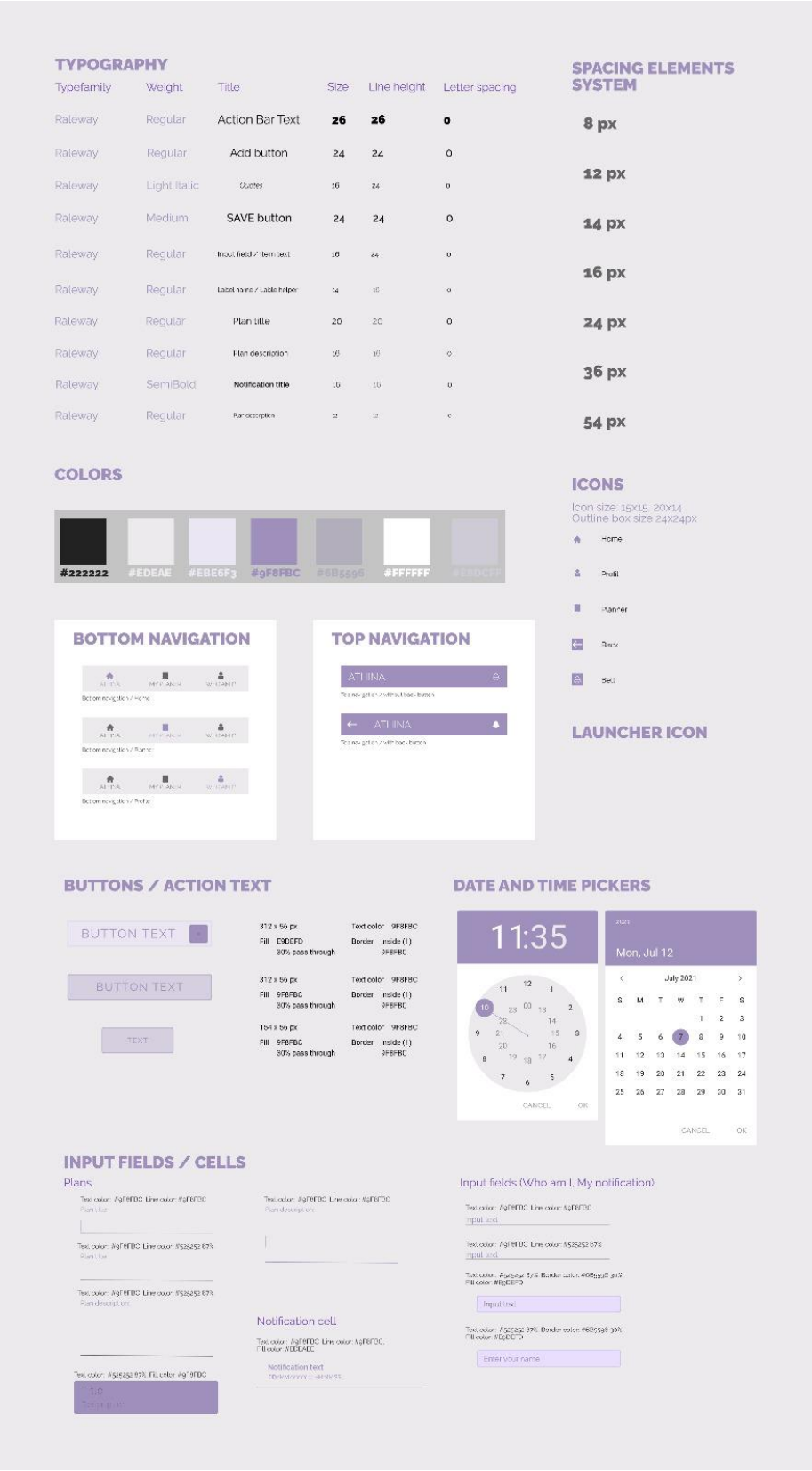
komponente su prvi elementi koji se izrađuju u dizajnu. U to su uključeni: gumbi, polja za unos, navigacije, podatkovni blokovi i drugi elementi poput birača datuma i vremena koji su korišteni u ovoj aplikaciji.

Komponente koje su korištene u izradi aplikacije mogu se vidjeti na slici 3.2 gdje je prikazana stranica komponente.



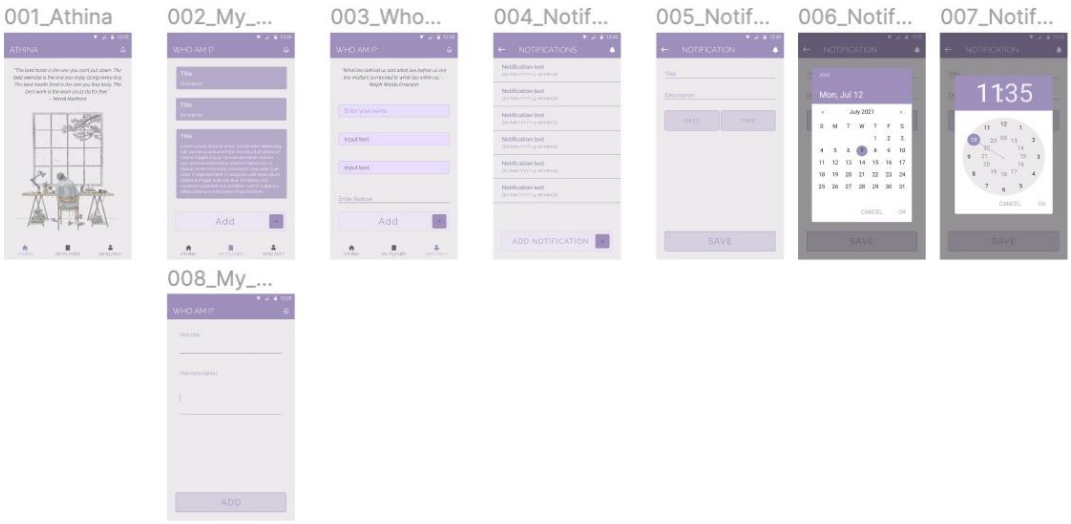
Slika 3.2 Stranica “Komponente” u alatu Figma

Kako bi se stvorio takozvani brend odnosno konzistentnost dizajna izrađen je stilski vodič po kojemu je rađena aplikacija. U njemu se mogu vidjeti fontovi, odnosno tipografija koja je korištena, boje, ikone, gumbi, polja za unos i ostale komponente koje su u stilskom vodiču pobliže opisane. Stilski vodič je prikazan na slici 3.3.



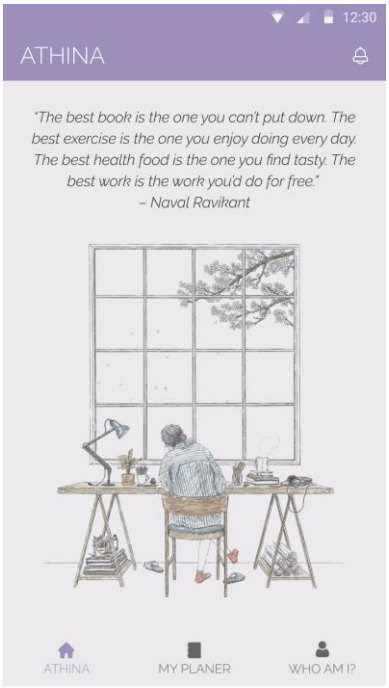
Slika 3.3 Stilski vodič aplikacije Athina

Na posljednjoj stranici alata nalazi se konačan dizajn aplikacije (Slika 3.4**Error! Reference s ource not found.**).



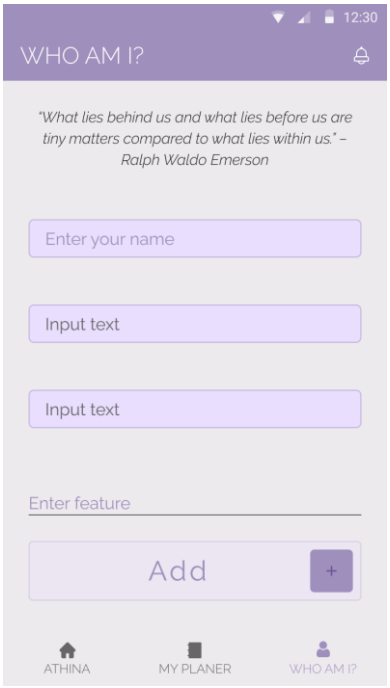
Slika 3.4 Dizajn aplikacije Athina

Aplikacija se sastoji od tri glavna zaslona: početnog zaslona, planera i profila. Na slikama 3.5 , 3.6 i 3.7 su vidljivi navedeni zasloni.



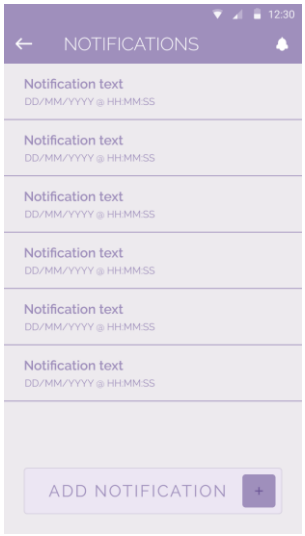
Slika 3.5 Početni zaslon

Slika 3.6 Planer

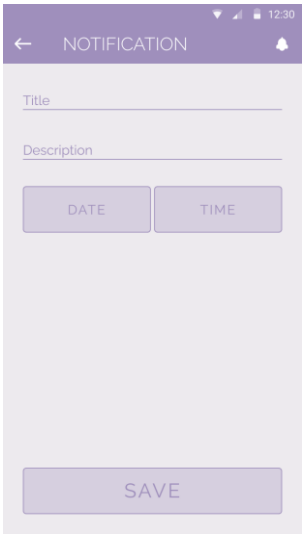


Slika 3.7 Profil

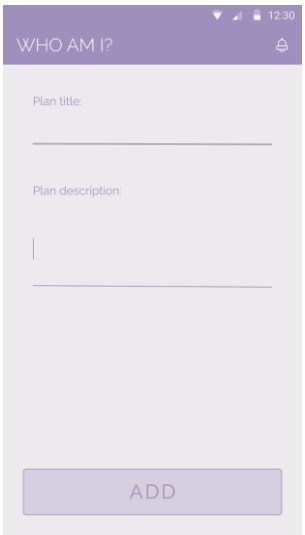
Preostali zasloni su: zaslon s obavijestima, zaslon za dodavanje planova i zaslon za dodavanje obavijesti.



Slika 3.8 Obavijesti



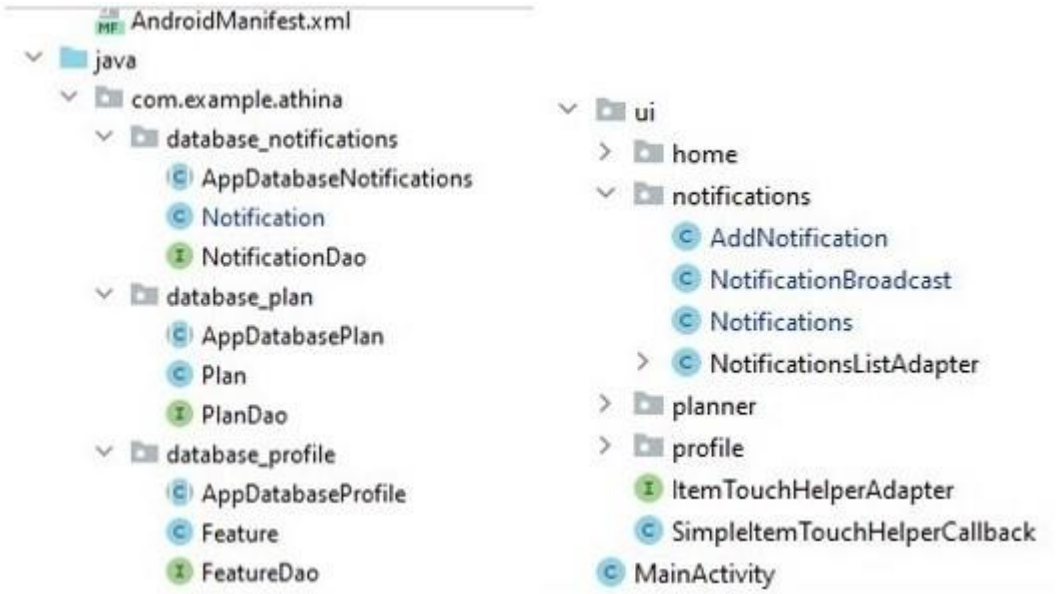
Slika 3.9 Dodavanje planova



Slika 3.10 Dodavanje obavijesti

4.3. Pretvaranje slike u kod

Kada je dizajn gotov prelazi se na programiranje. Najprije se rade pogledi, odnosno zasloni u obliku XML dokumenata te im se uz pomoć Java ili Kotlin programskog jezika dodaju funkcionalnosti. Za dovršetak ovoga koraka korišten je programski alat Android Studio. Android Studio je službeno integrirano razvojno okruženje (engl. *Integrated Development Environment*) koje služi kao alat za razvoj Android aplikacija. Prva stabilna verzija Android Studija objavljena je 2014. godine u prosincu. Kako bi ovaj program mogao raditi računalo mora zadovoljavati osnovne zahtjeve sustava koji se mogu pronaći na službenoj stranici alata. Za izradu Android aplikacije najprije je potrebno izraditi projekt u Android Studiju. Projekt sadrži sve što definira radni prostor za aplikaciju, od izvornog koda i resursa, do testiranja koda i izgradnje konfiguracija. Svaki se projekt sastoji od niza dokumenata i klasa koji definiraju aplikaciju. Kako ćemo strukturirati naš program isključivo ovisi o nama. Najbolje je podijeliti dokumente aplikacije po funkcionalnostima ili hijerarhijski po strukturi same aplikacije. Primjer strukture programa Athina može se vidjeti na slici 3.11.



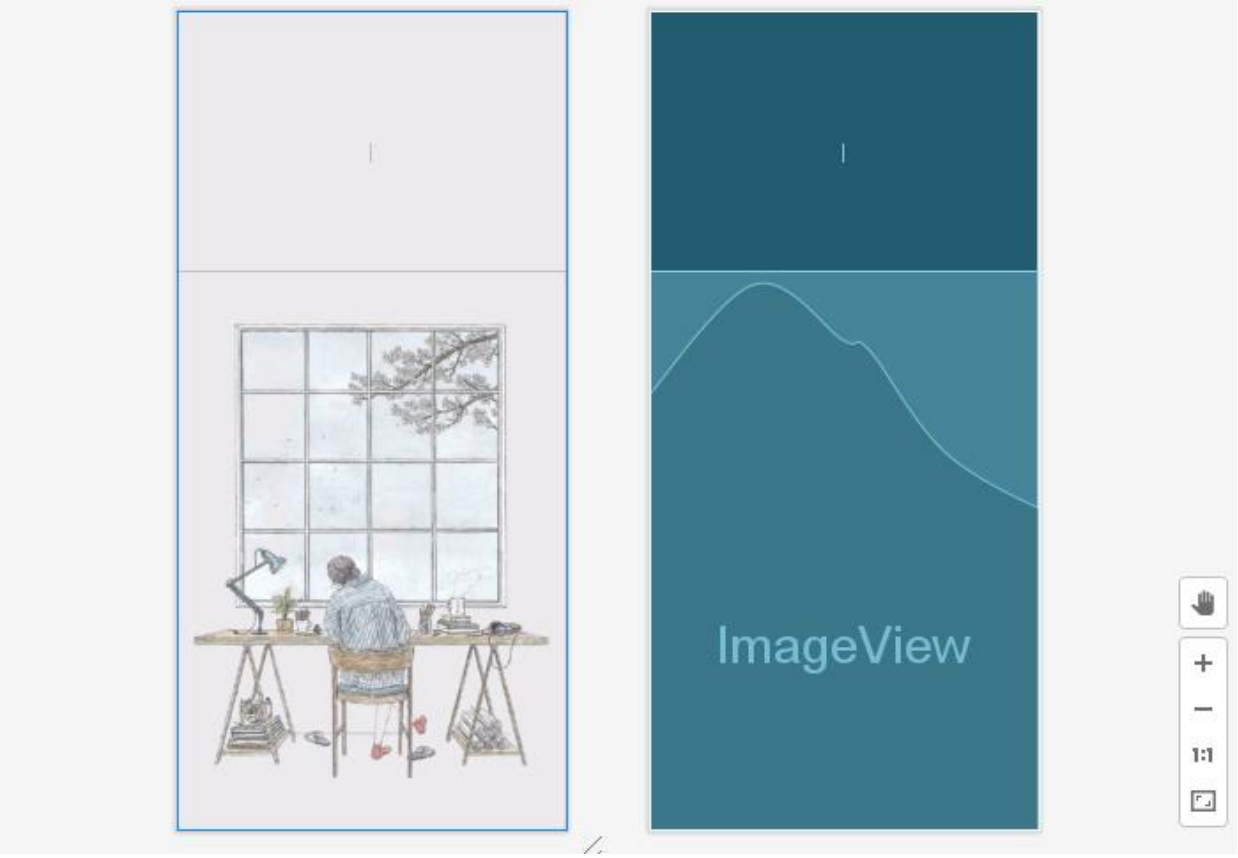
Slika 3.11 Struktura projekta Athina

Od dokumenata svaki projekt mora imati dokument pod nazivom AndroidManifest.xml koji Android alatima opisuje glavne informacije o projektu. Osim njega svaka aplikacija mora imati barem jedan aktivni prozor aplikacije koji se na priloženoj slici može vidjeti pod nazivom MainActivity. Android Studio nudi mogućnost korištenja programskih jezika Java i Kotlin. U izradi aplikacije Athina korišten je programski jezik Java.

4.3.1. XML – izrada rasporeda

XML (engl. *EXtensible Markup Language*) je softverski i hardverski neovisan alat za pohranu i transport podataka. Mnogi ga uspoređuju s HTML jezikom za izradu mrežnih aplikacija [13]. Najveća razlika im je upravo svrha. XML je dizajniran za transport podataka s naglaskom na podatke kao takve, dok je HTML dizajniran za prikaz podataka s naglaskom na izgled podataka. Usprkos tome, XML se upravo koristi kao alat za izradu "pogleda" aplikacije. U XML datoteke se pohranjuju podaci o izgledu, stilu i poziciji elemenata na zaslonu. Svaki od elemenata predstavlja klase u programskom jeziku. Inicijaliziranjem i dodavanjem izgleda nastaju objekti nad kojima vršimo određene radnje. Kao primjer postupka izrade XML dokumenta na osnovi dizajna uzet je početni zaslon.

Prvo što se može vidjeti u kodu je <?xml version = "1.0" što govori da je ovo XML standard kojem ova datoteka odgovara (Slika 3.13). Nadalje slijedi encoding = "utf-8" što označava način kodiranja datoteke koji je u ovome slučaju UTF-8 Unicode kodiranje.



Slika 3.12 Android Studio prikaz početnoga zaslona

Linija ***Kod***

```
1:      <?xml version="1.0" encoding="utf-8"?>
2:      <androidx.constraintlayout.widget.ConstraintLayout
3:      xmlns:android="http://schemas.android.com/apk/res/android"
4:      xmlns:app="http://schemas.android.com/apk/res-auto"
6:      xmlns:tools="http://schemas.android.com/tools"
7:      android:layout_width="match_parent"
8:      android:layout_height="match_parent"
9:      android:fitsSystemWindows="true"
10:     android:background="@color/background"
11:     android:background="@color/background"
12:     tools:context=".ui.home.HomeFragment">
13:     <TextView
14:     android:id="@+id/quoteHomepage"
15:     android:layout_width="wrap_content"
16:     android:layout_height="wrap_content"
17:     android:fontFamily="@font/raleway_light_italic"
18:     android:layout_marginRight="24dp"
19:     android:layout_marginLeft="24dp"
20:     android:layout_marginTop="24dp"
21:     android:textSize="14sp"
22:     android:lineHeight="20sp"
23:     android:translationZ="0dp"
24:     android:background="@color/background"
25:     android:textAlignment="center"
26:     app:layout_constraintEnd_toEndOf="parent"
27:     app:layout_constraintStart_toStartOf="parent"
28:     app:layout_constraintTop_toTopOf="parent" />
29:     <ImageView
30:     android:id="@+id/imageView"
31:     android:layout_width="fill_parent"
32:     android:layout_height="0dp"
33:     android:scrollbars="vertical"
34:     android:src="@drawable/homepage_imagee"
35:     app:layout_constraintBottom_toBottomOf="parent"
36:     app:layout_constraintEnd_toEndOf="parent"
37:     app:layout_constraintTop_toBottomOf="@id/quoteHomepage"
38:     app:layout_constraintStart_toStartOf="parent" />
39:
40:     </androidx.constraintlayout.widget.ConstraintLayout>
```

Slika 3.13 Kod izgleda početnog zaslona (fragment_home.xml)

U Android Studiju postoje tri vrste rasporeda elemenata [14] koji definiraju na koji način su složeni elementi. To su: linearni (engl. *LinearLayout*), relativni (engl. *RelativeLayout*) i raspored s ograničenjima (engl. *ConstraintLayout*).

U linearnom rasporedu elementi se slažu jedno ispod drugoga, dok se u relativnom rasporedu slažu u odnosu jedni na druge. Raspored s ograničenjima je raspored koji nudi prilagodljive i fleksibilne načine prikaza. Ovakav raspored je trenutno zadani raspored u Android Studiju. Glavna prednost mu je omogućavanje izrade velikih i složenih izgleda s hijerarhijom ravnoga prikaza. Unutar rasporeda slažu se svi elementi. Elementi koji se mogu vidjeti u kodu i na slici su: tekstualni prikaz (engl. *TextView*) i prikaz slike (engl. *ImageView*).

Svaki od elemenata mora sadržavati atribute visinu i širinu. Dobra praksa je svakom elementu dodijeliti jedinstvenu oznaku (id) pomoću koje pristupamo pojedinom elementu. Osim visine i širine ukoliko se koristi raspored s ograničenjima moraju se postaviti ograničenja svakome elementu. Ograničenja se mogu postaviti u odnosu na roditeljski element ili bilo koji susjedni element. U primjeru se može vidjeti da su sva ograničenja postavljena u odnosu na roditeljski element osim gornje strane prikaza slike koji je ograničen s donjom stranom tekstualnog prikaza.

Osim visine i širine elementu se mogu postaviti margine i uvlake, boja, veličina teksta, pozadina, poravnanje i drugi atributi koji ih opisuju.

4.3.2. Navigacija – tok aplikacije

Tok aplikacije jest način na koji korisnik prolazi kroz određene zaslone aplikacije te kako odrađuje određene radnje u aplikaciji. Aplikacija Athina je vrlo jednostavna aplikacija od šest zaslona do kojih se dolazi pomoću navigacije u podnožju, gumba u zaglavlju te gumba za dodavanje elemenata.

Donja i gornja navigacija mogu se vidjeti na slikama . Donja navigacija se sastoji od tri gumba od kojih prvi vodi na početni zaslon, drugi na planer i treći na profil. U gornjoj navigaciji nalazi se gumb u obliku zvona koje označava obavijesti. Klikom na taj gumb dolazi se na zaslon s obavijestima.

Na planer zaslonu i zaslonu s obavijestima nalazi se gumb za dodavanje planova i obavijesti. Svaki od gumba vodi na novi zaslon. Na navedenim zaslonima korisnik treba unijeti podatke o planu ili obavijesti ovisno o kojemu zaslonu je riječ. Pritiskom na gumb za spremanje podataka kreiran je plan odnosno obavijest koja je pritom vidljiva na planer zaslonu odnosno zaslonu s obavijestima.

Na profilu se nalaze dva polja za unos, jedan od njih služi za upisivanje imena korisnika dok drugi služi za dodavanje korisnikovih preferenci, hobija i drugih osobina. Kako bi osobina bila dodana osim što se mora upisati željeni sadržaj mora se pritisnuti i na gumb za dodavanje te osobine koji se nalazi ispod polja za unos. Dodana osobina nakon pritiska gumba za dodavanje bit će vidljiva na istom zaslonu ispod polja za unos imena korisnika.

4.3.3. Spremanje podataka pomoću baze podataka

Za spremanje podataka u lokalnoj bazi podataka koristi se Room baza podataka [15]. Room je biblioteka objektnog relacijskog mapiranja koja preslikava objekte naše baze podataka u Java objekte. Ona je zapravo sloj na vrhu SQLite baze podataka čime omogućava tečan pristup bazama podataka. Za korištenje ove biblioteke najprije se moraju implementirati sljedeće ovisnosti u build.gradle datoteku projekta (Slika 3.14).

Linija ***Kod***

```
1:      def room_version = "2.3.0"
2:
3:      implementation "androidx.room:room-runtime:$room_version"
4:      annotationProcessor "androidx.room:room-compiler:$room_version"
```

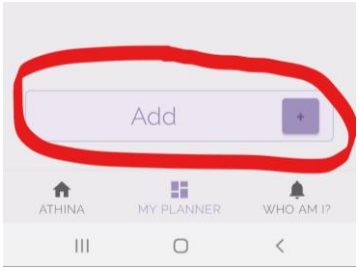
Slika 3.14 Implementacija ovisnosti u build.gradle datoteku

Postoje tri glavne komponente Room baze podataka [16]. Prva od njih jest klasa baze podataka koja drži bazu podataka i omogućava pristup trajnim podacima aplikacije. Kako je riječ o SQLite bazi podataka potrebni su entiteti podataka (engl. *Data entities*) koji predstavljaju tablice u bazi podataka aplikacije. Osim toga, potrebni su i objekti za pristup podacima koji pružaju metode pomoću kojih se šalju upiti, ažuriraju, dodaju i brišu podatci iz baze.

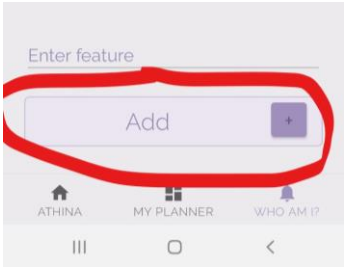
U aplikaciji Athina ovakav način pohrane koristi se kod spremanja planova, osobina i obavijesti.

Spremanje podataka događa se u onome trenutku kada se pritisne na gumb za dodavanje planova, osobina ili obavijesti. Ti gumbi mogu se vidjeti na slikama 3.15, 3.16 i 3.17.

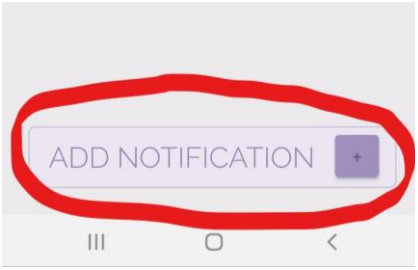
Ono što se u kodu događa prilikom pritiska na gumb je stvaranje nove instance baze podataka i



Slika 3.15 Gumb za dodavanje planova



Slika 3.16 Gumb za dodavanje osobina



Slika 3.17 Gumb za dodavanje obavijesti

dodavanje novog plana, osobine ili obavijesti u navedenu bazu podataka. Podatci koji se spremaju u bazu dohvaćaju se iz EditText elemenata te iz dijaloga za datum i vrijeme kod obavijesti. Text koji je upisan u EditText element vrlo jednostavno se dohvaća pomoću metode getText() dok se

dohvaćanje vremena i datuma vrši pomoću metoda onDateSet i onTimeSet koje su vidljive na slici 3.18.

Linija ***Kod***

```
1:       @Override
2:       public void onDateSet(DatePicker view, int year, int month, int day) {
3:             this.year = year;
4:             this.month = month;
6:             this.day = day;
7:       }
8:
9:       @Override
10:      public void onTimeSet(TimePicker view, int hourOfDay, int minute) {
11:            this.hour = hourOfDay;
12:            this.minute = minute;
13:      }
```

Slika 3.18 Metode onDateSet() i onTimeSet()

U ovim metodama u atribute klase AddNotification spremaju se godina, mjesec, dan, sat i minuta (Slika 3.19). Prije no što je obavijest dodana u bazu podataka mora se kreirati te u nju spremiti svi atributi koji su definirani u entitetu. Obavijest se naposljetku dodaje u bazu preko objekta za pristup podacima koji se poziva na stvorenoj instanci baze.

Linija ***Kod***

```
1:       public class AddNotification extends AppCompatAct...
2:             ...
3:             private int year;
4:             private int month;
6:             private int day;
7:             private int hour;
8:             private int minute;
9:             ...
```

Slika 3.19 Atributi klase AddNotification za privremeno spremanje podatak o datumu i vremenu

Na sličan način kao i za dodavanje tako i za brisanje podataka iz baze koristi se instanca baze i objekt za pristup podacima što se može vidjeti na slici 3.20.

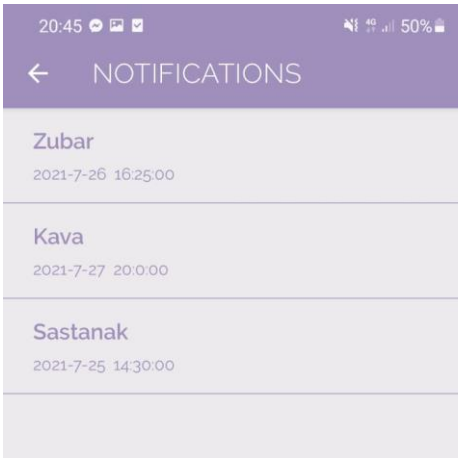
Linija Kod

```
1:      private void deleteNotification(Notification notification) {
2:            AppDatabaseNotifications databaseNotifications =
3:               AppDatabaseNotifications.getDBInstance(getApplicationContext());
4:            cancelAlarm(notification);
6:            databaseNotifications.notificationDao().delete(notification);
7:      }
8:
9:      private void cancelAlarm(Notification notification){
10:         AlarmManager alarmManager = (AlarmManager)
11:            getApplicationContext().getSystemService(Context.ALARM_SERVICE);
12:         Intent intent=new Intent(getApplicationContext(), NotificationBroadcast.class);
13:         intent.setAction("notification_action");
14:         PendingIntent pendingIntent = PendingIntent
15:            .getBroadcast(getApplicationContext(), notification.nAlarmId, intent,
16:            PendingIntent.FLAG_UPDATE_CURRENT);
17:         alarmManager.cancel(pendingIntent);
18:         pendingIntent.cancel();
19:     }
```

Slika 3.20 Metoda za brisanje obavijesti

4.3.4. Obavijesti

Glavna funkcionalnost koja će biti opisana jesu obavijesti. Aplikacija Athina omogućava izradu obavijesti na uređaju koji mogu podsjetiti na nadolazeće događaje ili druge obveze. Za svaku obavijest mora se odabrati datum i vrijeme u koje će obavijest stići na mobilni uređaj. Sve obavijesti koje su ikada izrađene nalaze se u obliku popisa u aplikaciji (Slika 3.21).



Slika 3.21 Zaslon s popisom izrađenih obavijesti

Slika 3.22 Gumb za pristup zaslonu s obavijestima

Dođe li iz nekog razloga do potrebe za brisanjem obavijesti to se čini na isti način na koji se brišu planovi, a to jest povlačenjem obavijesti u desnu stranu. Kako bi ova funkcionalnost uvijek bila dostupna tijekom uporabe aplikacije do nje se dolazi pritiskom na crno zvonce u gornjem desnom kutu aplikacije (Slika 3.22). Što zapravo leži iza ove naizgled vrlo jednostavne funkcionalnosti bit će opisano uz priloženi kod u nastavku.

Ono što se prvo dogodi pri izradi obavijesti jest spremanje obavijesti u bazu te nakon toga pomoću emitera (engl. *broadcaster*) i alarm menadžera obavijest se zakazuje u točno definirano vrijeme. Za izračun vremena u mikrosekundama koristi se metoda getDateTimeFromSelectedValues() koja pomoću instance klase Calendar vraća vrijeme proteklo od trenutka kada je obavijest izrađena sve do odabranoga datuma i vremena kada će obavijest biti prikazana na uređaju (Slika 3.23).

Linija ***Kod***

```
1:      public long getDateTimeFromSelectedValues() {
2:            Calendar cal = Calendar.getInstance();
3:            cal.set(Calendar.DAY_OF_MONTH, day);
4:            cal.set(Calendar.MONTH, month);
6:            cal.set(Calendar.YEAR, year);
7:            cal.set(Calendar.HOUR_OF_DAY, hour);
8:            cal.set(Calendar.MINUTE, minute);
9:            cal.set(Calendar.SECOND, 0);
10:           Log.d("NOTIFICATION", "addAlarm: day: " + day);
11:           Log.d("NOTIFICATION", "addAlarm: month: " + month);
12:           Log.d("NOTIFICATION", "addAlarm: year: " + year);
13:           Log.d("NOTIFICATION", "addAlarm: hour: " + hour);
14:           Log.d("NOTIFICATION", "addAlarm: minute: " + minute);
15:           return cal.getTimeInMillis();
16:     }
```

Slika 3.23 Metoda getDateTimeFromSelectedValues()

Ranije navedeno zakazivanje obavijesti događa se u metodi addAlarm() koju se poziva u 16. liniji koda sa slike 3.24. Unutar ove metode izrađuje se takozvani PendingIntent odnosno token koji se pomoću emitera prenosi vanjskoj aplikaciji pomoću koje nam dolaze obavijesti na uređaju.

Kako bi obavijest stigla u točno određeno vrijeme koristi se objekt klase AlarmManager i njegova metoda setExactAndAllowWhileIdle().

Linija Kod

```
1:      private void addAlarm(long timeToSet, String name, String description) {
2:        long alarmTime = 0L;
3:        Calendar calendar = Calendar.getInstance();
4:        calendar.setTimeInMillis(timeToSet);
6:        alarmTime = calendar.getTimeInMillis();
7:        PendingIntent alarmIntent = getAlarmIntent(alarmTime, description, name);
8:        AlarmManager alarmManager = (AlarmManager)
9:              getApplicationContext().getSystemService(Context.ALARM_SERVICE);
10:       alarmManager.setExactAndAllowWhileIdle(AlarmManager.RTC, alarmTime, alarmIntent);
11:     }
12:     private PendingIntent getAlarmIntent(long alarmId,String description, String name) {
13:          Intent intent=new Intent(getApplicationContext(), NotificationBroadcast.class);
14:          intent.putExtra("todo", description);
15:          intent.putExtra("title", name);
16:          intent.setAction("notification_action");
17:          return PendingIntent.getBroadcast(getApplicationContext(),
18:              (int) alarmId, intent, PendingIntent.FLAG_UPDATE_CURRENT);
19:     }
```

Slika 3.24 Metoda addAlarm()

Sve metode koje su do sada opisane naposljetku se koriste u metodi saveAndScheduleNotification() koja ih poziva te u potpunosti izvršava dodavanje obavijesti.

Linija Kod

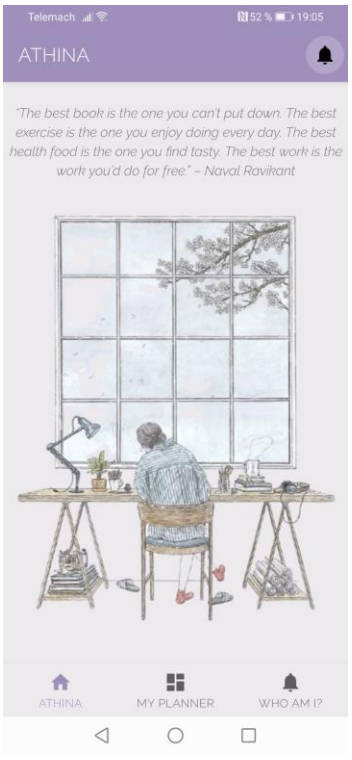
```
1:      private int saveAndScheduleNotification(String nName, String nDescription) {
2:            AppDatabaseNotifications database =
3:                  AppDatabaseNotifications.getDBInstance(this.getApplicationContext());
4:            Notification notification = new Notification();
6:            notification.nTitle = nName;
7:            long alarmTime = getDateTimeFromSelectedValues();
8:            notification.nAlarmId = (int) alarmTime;
9:            notification.nDescription = nDescription;
10:           notification.nYear = year;
11:           notification.nMonth = month;
12:           notification.nDay = day;
13:           notification.nHour = hour;
14:           notification.nMinute = minute;
15:           database.notificationDao().insertNotification(notification);
16:           addAlarm(alarmTime, notification.nTitle, notification.nDescription);
17:           Toast.makeText(getApplicationContext(), "New notification was made!",
18:                 Toast.LENGTH_SHORT).show();
19:           return notification.uid;
20:     }
```

Slika 3.25 Metoda saveAndScheduleNotification()

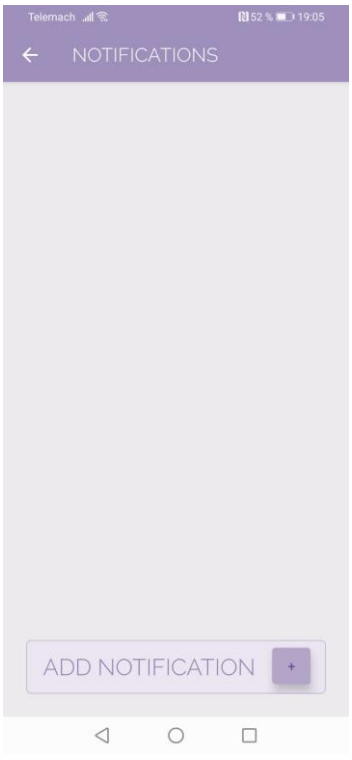
4.3.5. Primjena aplikacije Athina

Aplikaciju Athina može se opisati kao digitalni rokovnik u koji se upisuju planovi i bilješke te pomoću kojega se izrađuju podsjetnici. Okolnosti u kojima ovakav podsjetnik može poslužiti jest prilikom obavljanja mjesečne kupovine, bilježenja zakazanih termina te zapisivanja planova, ideja ili ciljeva. Način zapisivanja bilješki u aplikaciji Athina vrlo je jednostavan. Želi li se izraditi podsjetnik za stomatološki pregled potrebno je obaviti sljedeće korake:

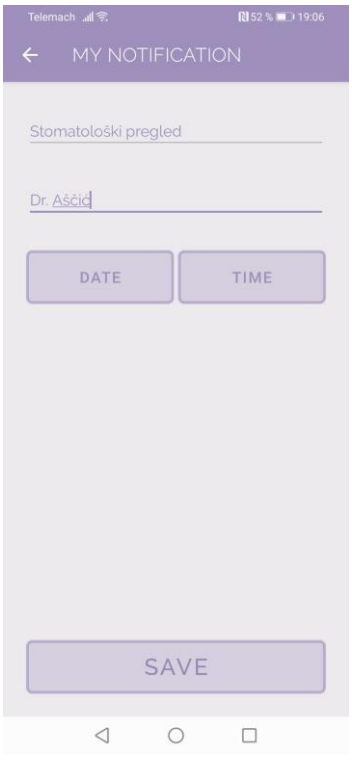
- Preuzeti aplikaciju na mobilni uređaj
- Otvoriti aplikaciju pritiskom na ikonu vidljivu na slici…
- Pritisnuti zvonce u gornjem desnom kutu ekrana ()
- Pritisnuti gumb s oznakom „+“ na dnu zaslona
- Ispuniti tražene podatke među kojima su naslov i opis podsjetnika te datum i vrijeme u koje se želi biti obaviješten
- Pritisnuti gumb s oznakom „SAVE“.



Slika 3.26 Pritisak gumba za pristup obavijestima

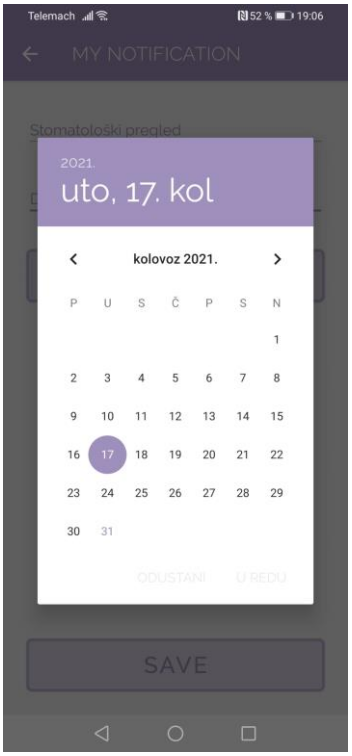


Slika 3.27 Pritisak gumba s oznakom "+" za izradu obavijesti



Slika 3.28 Unošenje naslova i opisa podsjetnika

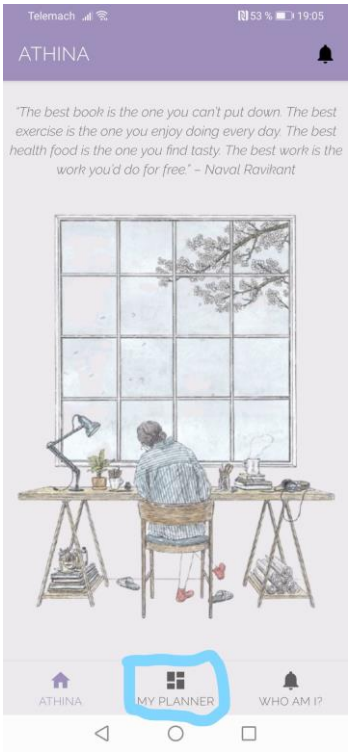
Pritiskom na navedeni gumb kreiran je podsjetnik. Obavijest o stomatološkom pregledu stići će u točno odabrano vrijeme i na točno odabrani datum. Ovako izrađena obavijest vrlo je jednostavna



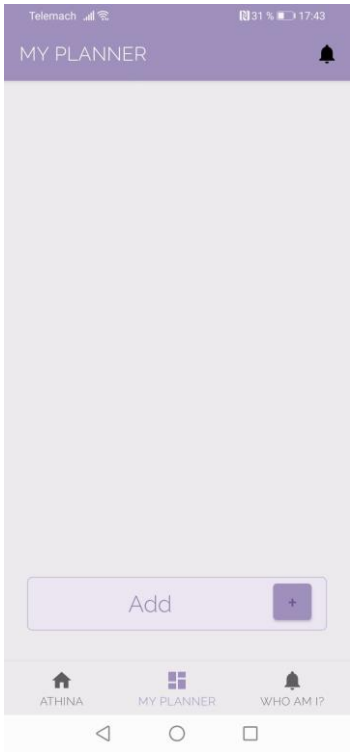
Slika 3.29 Odabir datuma

Slika 3.30 Odabir vremena

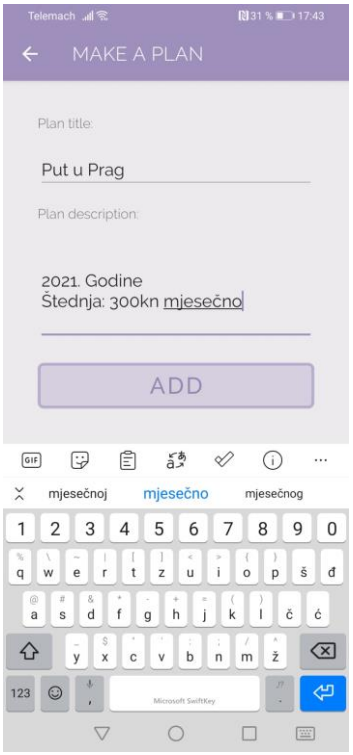
Slika 3.31 Primanje obavijesti na pametni telefon



Slika 3.32 Pritisak gumba "MY PLANNER" u navigaciji



Slika 3.33 Pritisak gumba s oznakom "+" za izradu plana



Slika 3.34 Unošenje naslova i opisa plana

te kao takva može biti laka za izradu i korištenje, no s druge strane može biti oskudna i nedovoljno

razrađena. Postoji mogućnost da bi pojedini korisnici voljeli imati opsežno opisane obavijesti s mnogo detalja. Nekim korisnicima mogao bi nedostajati pogodnost dodavanja medijskih sadržaja ili prikaz dana u tjednu kada će biti obaviješteni.

Izuzev izrade podsjetnika korisnik na vrlo jednostavan način može zabilježiti svoje planove. Izrada plana uključuje naredne korake:

- Preuzimanje aplikacije na mobilni uređaj
- Otvoriti aplikaciju pritiskom na ikonu vidljivu na slici
- Pritisnuti na gumb „MY PLANNER“ koji se nalazi na navigacijskoj traci aplikacije
- Pritisnuti na gumb s oznakom „+“ na dnu zaslona
- Ispuniti naziv i opis plana
- Pritisnuti na gumb pod nazivom „ADD“

Pritiskom na naznačeni gumb uspješno je dodan plan u popis planova. Na gotovo jednak način mogu se dodati i takozvane osobine odnosno kratke bilješke na profilu aplikacije. Zamisao profila jest okarakterizivanje pojedinca. Ova sekcija čini aplikaciju personaliziranom i jedinstvenom. Postoje mnoge aplikacije na tržištu koje pružaju neke od ili čak i više usluga od aplikacije Athina. Danas u velikom broju slučaja takve aplikacije ili posjeduju samo jednu od navedenih usluga ili mnogo više usluga koje su mnogo složenije i tako zahtjevne za korištenje. Aplikacijom Athina lako mogu baratati korisnici svih dobnih skupina kao i u mnoge svrhe. Neki korisnici zbog jednostavnosti i elegantnoga dizajna preferirat će aplikaciju kao što je Athina. Aplikacije kao što su Microsoft OneNote, Evernote, Microsoft To Do, ToDoist i To-Do List samo su neke od mnogih aplikacija koje služe za organizaciju svakodnevice. Svaka od njih nudi mnoštvo usluga koje nakon nekog vremena mogu postati iscrpljujuće za korištenje. Aplikacija Athina je idealan odabir za one korisnike koji žele lijepu aplikaciju u koju će moći zapisivati privremene planove kao i želje i ciljeve.

5. ZAKLJUČAK

Pri izradi Android aplikacija vrlo je korisno slijediti korake koji se sastoje u većini slučajeva od raščlambe ideje, izrade dizajna te izrade same aplikacije korištenjem alata Android Studio. Kako je izrada kvalitetne aplikacije složen proces od velike pomoći može biti timski rad iliti rad u timovima. Grupa ljudi će ipak mnogo brže doći do željenoga rezultata negoli jedna osoba. Korištenjem alata kao što su google-ova usluga Google Meets, Figma, GitHub i drugi omogućava se sinkronizirani rad tima. Iznošenje ideje i njeno dokumentiranje je vrlo važna stavka u izradi aplikacije. Kako bi se mogao napraviti dizajn aplikacije i sama aplikacija mora imati točno definirane funkcionalnosti te način na koji ih se želi prikazati. U ovome koraku svi članovi tima trebaju sudjelovati ili barem biti dobro upućeni u sve donesene odluke. Ideja aplikacije Athina jest omogućiti korisniku pomoć pri organizaciji svakodnevice. To se postiglo sljedećim funkcionalnostima aplikacije: planerom, mogućnošću izrade obavijesti iliti podsjetnika te dodavanjem takozvanih osobina u svoj profil. Sve navedene funkcionalnosti prvobitno su dizajnirane u alatu Figma gdje se uz nekoliko varijacija izgleda aplikacije došlo do konačnoga dizajna. Završetkom dizajna izgled aplikacije i njeno ponašanje trebaju u potpunosti biti određeni. Posljednjim korakom dolazi se do „opipljivoga“ rezultata odnosno android aplikacije koja se zapravo može koristiti. Osim provjere radi li i sadržava li aplikacija sve što je definirano dizajnom dobra praksa jest pokrivanje rubnih slučajeva gdje može doći do otežanog korištenja aplikacije ili grešaka u aplikaciji. U aplikaciji Athina nisu pokriveni svi rubni slučajevi, ali za korisnike koji neće namjerno tražiti greške u aplikaciji omogućeno je nesmetano korištenje svih značajki. Postoje koraci koji su navedeni u radu, ali nisu primijenjeni na aplikaciji. Kada na aplikaciji radi više od jedne osobe oni su vrlo korisni za primijeniti. Treba zapamtiti kako niti jedna aplikacija ne može biti u potpunosti bez grešaka te da se u današnjim poduzećima zapošljavaju osobe koje isključivo brinu o kvaliteti aplikacije.

LITERATURA

[1] S., I., Gonzalez, *AN EXPLORATION INTO THE HISTORY AND SIGNIFICANCE OF THE 'PLANNER', 2018.*, dostupno na: http://thesis.honors.olemiss.edu/1156/7/Gonzalez%20SMBHC%20Thesis.pdf [08.09.2021.]

[2] S., Pitroda, *Timeline, 2020.*, dostupno na: https://www.sampitroda.com/timeline[08.09.2021.]

[3] P., C., Lai, *Intention to Use a Drug Reminder App: A Case Study of Diabetics and High Blood Pressure Patients, 2020.*, dostupno na: https://www.researchgate.net/publication/342223808_Intention_to_Use_a_Drug_Reminder_App_A_Case_Study_of_Diabetics_and_High_Blood_Pressure_Patients [08.09.2021.]

[4] *How people discover, use, and stay engaged with apps*, 2016., dostupno na: https://www.thinkwithgoogle.com/_qs/documents/331/how-users-discover-use-apps-google-research.pdf [08.09.2021.]

[5] A., Rudzate, *5 Best Daily Planner Apps To Boost Your Productivity*, 2021., dostupno na: https://www.lifehack.org/901850/best-daily-planner-app [08.09.2021.]

[6] R. Kumulos, 20 Questions To Ask Your Client Before You Build Their Mobile App, Kumulos, 2015., dostupno na: https://www.kumulos.com/2015/12/17/20-questions-to-ask-your-client-before-you-build-their-mobile-app/ [09.07.2021]

[7] M. Rivera, 7 Steps to Writing the Perfect Project Proposal, The Blueprint, 2021., dostupno na: https://www.fool.com/the-blueprint/project-proposal/ [09.07.2021]

[8] Podsjetnik, Jezikoslovac, dostupno na: https://jezikoslovac.com/word/y38h [09.07.2021]

[9] *project planning*, TechTarget, dostupno na: https://searchcio.techtarget.com/definition/project-planning [10.07.2021]

[10] *Figma-Our story*, Figma, dostupno na: https://www.figma.com/about/ [10.07.2021]

[11] J.J., Garret, *The Elements of User Experience: User-Centered Design for the Web and Beyond*, Second Edition, New Riders, USA, 2011.

[12] S., Minhas, *8 rules of mobile design - Useful Tips for Designing Mobile Apps*, UX Collective, 2019., dostupno na: https://uxdesign.cc/8-rules-of-mobile-design-1b8d9936c241 [10.07.2021]

[13] *Introduction to XML*, w3schools, dostupno na: https://www.w3schools.com/xml/xml_whatis.asp [12.07.2021.]

[14] *Layouts*, Android for Developers, dostupno na: https://developer.android.com/guide/topics/ui/declaring-layout [12.07.2021.]

[15] A., Rawat, *Using Room Database | Android Jetpack*, MindOrks, 2019., dostupno na: https://medium.com/mindorks/using-room-database-android-jetpack-675a89a0e942 [12.07.2021.]

[16] *Save data in a local database using Room*, Android for Developers, dostupno na: https://developer.android.com/training/data-storage/room [12.07.2021.]

SAŽETAK

Android aplikacija za organizaciju obavijesti na mobitelu

Ovim završnim radom opisan je prije svega postupak izrade proizvoljne Android aplikacija. Prvi korak pri izradi jest razrada ideje i njeno dokumentiranje. Kako bi se ostali koraci implementacije lakše izveli potrebno je što bolje promisliti o željenoj ideji i alatima koji će se koristiti. Drugi veliki korak jest izrada dizajna. Uz priloženi dizajn aplikacije Athina, ovaj korak je lako razraditi te primijeniti na vlastitoj aplikaciji. U dizajnu se opisuje korisničko sučelje i brine o korisničkom iskustvu prateći ranije definirane smjernice. U završnom koraku dolazi do krajnje izrade aplikacije u alatu Android Studio gdje je bitno ranije dobro istražiti mogućnosti i načine implementacije željenoga dizajna. Informira li se i dobro istraži ideja u prvom koraku izrade niti u samoj implementaciji neće doći do problema. Problem može nastati ukoliko postoji gotov dizajn, no sama ideja nije dovoljno razrađena što može dovesti do toga da ne postoji način na koji bi se mogla izraditi aplikacija sa željenim funkcionalnostima. Ovako definirani koraci svakodnevno se koriste pri izradi brojnih aplikacija te omogućuju rad više osoba na jednom projektu i time brže rezultate.

Ključne riječi: Android aplikacija, dizajn, organizacija, korisničko iskustvo, korisničko sučelje

ABSTRACT

Android application for organizing notifications on a mobile phone

This final paper primarily describes the process of creating an arbitrary Android application. The first step in development is to develop an idea and document it. In order to make the other steps of the implementation easier, it is necessary to think as well as possible about the desired idea and the tools that will be used. The second big step is making the design. With the included Athina app design, this step is easy to work out and apply on your own app. The design describes the user interface and takes care of the user experience following the previously defined guidelines. In the final step, the application is finalized in the Android Studio tool, where it is important to research the possibilities and ways of implementing the desired design. If the idea is well informed and researched in the first step, there will be no problems in the implementation itself. The problem can arise if there is a ready-made design, but the idea itself is not sufficiently elaborated, which can lead to the fact that there is no way in which an application with the desired functionalities could be created. The steps defined in this way are used daily in the development of numerous applications and enable the work of multiple people on one project and thus faster results.

Keywords: Android application, design, organization, user experience, user interface

ŽIVOTOPIS

Vlatka Mihić rođena je 25. ožujka 2000. godine u Osijeku. Živi u Osijeku gdje je završila Osnovnu školu Ivana Filipovića. Poslije osnovne škole upisuje srednju školu, III. Gimnaziju u Osijeku.

Nakon završetka srednje škole upisuje sveučilišni studij Računarstva na fakultetu Elektrotehnike, računarstva i informacijskih tehnologija u Osijeku. Za vrijeme studiranja sudjeluje u izvannastavnim aktivnostima kao što su IEEExtreme natjecanje, kolegij Napredni programeri te na trećoj godini zauzima poziciju potpredsjednice IEEE studentskog ogranka u Osijeku.

2021. godine počinje raditi kao tester osiguranja kvalitete u digitalnoj agenciji COBE u Osijeku.

PRILOZI

Link na gitHub repozitorij: https://github.com/vlatkamihic/Athina

Link na Figma dokument: https://www.figma.com/file/oqh4NOqkidAnUUoqMzvS4o/Athina?node-id=5%3A52