Android aplikacija za upravljanje radnim nalozima

Knežević, Magdalena

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

Sveučilišni studij

ANDROID APLIKACIJA ZA UPRAVLJANJE RADNIM NALOZIMA

Završni rad

Magdalena Knežević

Osijek, 2023. godine

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

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

Osijek, 16.09.2023.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada na preddiplomskom sveučilišnom studiju

Ime i prezime Pristupnika:	Magdalena Knežević
Studij, smjer:	Programsko inženjerstvo
Mat. br. Pristupnika, godina upisa:	R4511, 27.07.2020.
OIB Pristupnika:	87245184638
Mentor:	izv. prof. dr. sc. Zdravko Krpić
Sumentor:	,
Sumentor iz tvrtke:	
Naslov završnog rada:	Android aplikacija za upravljanje radnim nalozima
Znanstvena grana rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak završnog rad:	Zadatak završnog rada je istražiti mogućnosti izrade mobilne aplikacije za upravljanje radnim nalozima. Istražiti životni ciklus radnog naloga kao evidencijske liste u izvođenju određenog posla ili zadatka unutar tvrtke ili institucije na uzorku od nekoliko lokalnih tvrtki. Predstaviti barem 8 istih ili sličnih programskih rješenja za tu svrhu koji će poslužiti kao inspiracija za praktični dio rada. Na osnovu istraživanja definirati
Prijedlog ocjene 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: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	16.09.2023.
Datum potvrde ocjene od strane Odbora:	24.09.2023.

Potvrda mentora o predaji konačne verzije rada:	Mentor elektronički potpisao predaju konačne verzije.
	Datum:

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

IZJAVA O ORIGINALNOSTI RADA

Osijek, 25.09.2023.

Ime i prezime studenta:	Magdalena Knežević
Studij:	Programsko inženjerstvo
Mat. br. studenta, godina upisa:	R4511, 27.07.2020.
Turnitin podudaranje [%]:	1

Ovom izjavom izjavljujem da je rad pod nazivom**: Android aplikacija za upravljanje radnim nalozima**

izrađen pod vodstvom mentora izv. prof. dr. sc. Zdravko Krpić

i sumentora ,

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

Potpis studenta:

SADRŽAJ

1. UVOD

Svaka tvrtka, neovisno o djelatnosti kojom se bavi, sadrži prostor gdje se izvršava proces proizvodnje i rada. Bez obzira radi li se o računalnoj tvrtki, malim obrtima ili velikoj tvornici, potrebno je održavati taj radni prostor. Radni prostor mora biti uređen po zakonski određenim propisima kako bi pružao sigurnost svakom radniku koji u njemu radi.

U manje zahtjevnim djelatnostima, najčešće se održavanje radnog prostora izvodi po potrebi, bez konkretnog plana. U većim tvrtkama postoje sektori kojima je osnovna zadaća održavanje radnog prostora, proizvodnih linija i svih postojećih instalacija. Izrađuju se planovi održavanja sa popisom radova koje je potrebno izvršiti kako bi se izbjegli neželjeni zastoji u proizvodnji. Svi izvršeni radovi moraju biti zabilježeni u za to predviđene obrasce kako bi u svakom trenutku bilo vidljivo po čijem nalogu, kada i tko je radio te što je rađeno na konkretnom mjestu, stroju ili opremi.

Obrazac koji se najčešće koristi u tu svrhu je radni nalog. Voditelj radniku izdaje radni nalog s jasnim uputama o mjestu rada i vrsti rada koje je potrebno obaviti. Radnik nakon obavljenog posla popunjava svoj dio naloga sa opisom obavljenog posla te potpisan radni nalog vraća voditelju. Potpisan radni nalog se pohranjuje u arhivu i koristi se kao dokaz o obavljenom poslu. Radni nalog kao dokument je univerzalan i koristi se u svim djelatnostima i nije isključivo vezan za tehničko održavanje i servisiranje.

Razvojem tehnologije, otvorila se mogućnost za digitalizacijom radnih naloga. Trenutno su na tržištu dostupne brojne aplikacije za upravljanje radnim nalozima. Većina tih aplikacija koje su prilagođene hrvatskom govornom području dostupne su kao web aplikacije ili ne nude vrlo intuitivno korisničko sučelje. Osim toga, slične aplikacije najčešće pružaju puno ostalih mogućnosti nevezanih za radne naloge, zbog čega dio koji se odnosi na upravljanje radnim nalozima nije detaljno implementiran. Stoga, Android aplikacija izrađena u okviru ovog rada nastojat će pružiti krajnjim korisnicima jednostavno snalaženje unutar same aplikacije, ali i rad isključivo s radnim nalozima, gdje će korisnici putem jednog klika moći stvoriti radni nalog i nastaviti s njegovom daljnjom uporabom.

1.1. Zadatak završnog rada

Zadatak završnog rada je izraditi Android aplikaciju koja će omogućiti izradu i upravljanje elektroničkim radnim nalozima, čime će se olakšati cjelokupni proces održavanja radnog pogona. Teorijski dio rada treba opisati životni ciklus jednog radnog naloga, ali i bitnu činjenicu kako uopće dolazi do potrebe za radnim nalogom. Predstavit će se nekoliko sličnih, već postojećih programskih rješenja te će se definirati zahtjevi na novo programsko rješenje. U praktičnom dijelu rada napravit će se konkretna aplikacija koja će omogućiti korisniku odabir i popunjavanje svih potrebnih informacija vezanih za sporni dokument.

2. PREGLED ŽIVOTNOG CIKLUSA RADNOG NALOGA I ZAHTJEVI NA APLIKACIJU

Kvalitetne aplikacije za upravljanje radnim nalozima moraju odgovarati na sve zahtjeve njezinih korisnika. Kako bi proces izrade takve aplikacije bio uspješan, potrebno je dobro razumjeti teorijsku podlogu procesa kroz koji jedan radni nalog prolazi. Na primjeru tvrtke *Vinka plus d.o.o.,* bit će prikazani elementi održavanja industrijskog pogona, počevši od vođenja, preko opisa procesa do dokumentacije koja predstavlja službeni izvještaj svega napravljenog, prema [1]. Radni nalog izdaje osoba koja zadaje zadatak na početku posla, a izvršitelji ga popunjavaju na kraju rada. Na temelju opisanih koraka u potpoglavljima 2.2. i 2.3., koji dovode do popunjavanja radnog naloga, bit će definirani zahtjevi na programsko rješenje, tj. potrebne implementacije opisanog procesa u Android aplikaciju.

2.1. Održavanje industrijskog pogona

U ovom poglavlju bit će definiran proces održavanja industrijskog pogona. Prilikom opisa navedenog procesa, definirat će se i proces izrade radnog naloga.

Svake godine, u tvrtki *Vinka plus d.o.o.,* izrađuje se *Plan preventivnog održavanja*. Navedeni *Plan* sadrži cjelogodišnji pregled unaprijed planiranih radova na proizvodnim linijama i prostorima tvrtke. Radove je potrebno odraditi kako proces proizvodnje ne bi imao neplanirane zastoje, koje za posljedicu imaju povećanu cijenu krajnjeg proizvoda. Rukovoditelj Sektora tehnike i energetike odgovoran je za izradu *Plana*, a Voditelj održavanja industrijskog pogona odgovoran je za provedbu *Plana preventivnog održavanja* u djelo. Voditelj održavanja industrijskog pogona brine o nadzoru i koordinaciji nad pojedinim fazama procesa te o mehaničkoj ispravnosti strojeva, električne ispravnosti pogonske opreme, zračnih i vodovodnih instalacija.

Održavanje industrijskog pogona dijeli se na dvije vrste, preventivno i korektivno.

Preventivno održavanje slijedi iz *Plana preventivnog održavanja* i Voditelj održavanja na osnovu *Plana*, izdaje radne naloge po kojima se izvršavaju planirani radovi na opremi. Korektivno održavanje je neplanirano održavanje do kojeg dolazi kada se uoči nepravilnost ili neispravnost na pojedinom stroju ili radnom prostoru.

U potpoglavljima 2.2. i 2.3. detaljnije su opisane obje vrste održavanja.

2.2. Preventivno održavanje

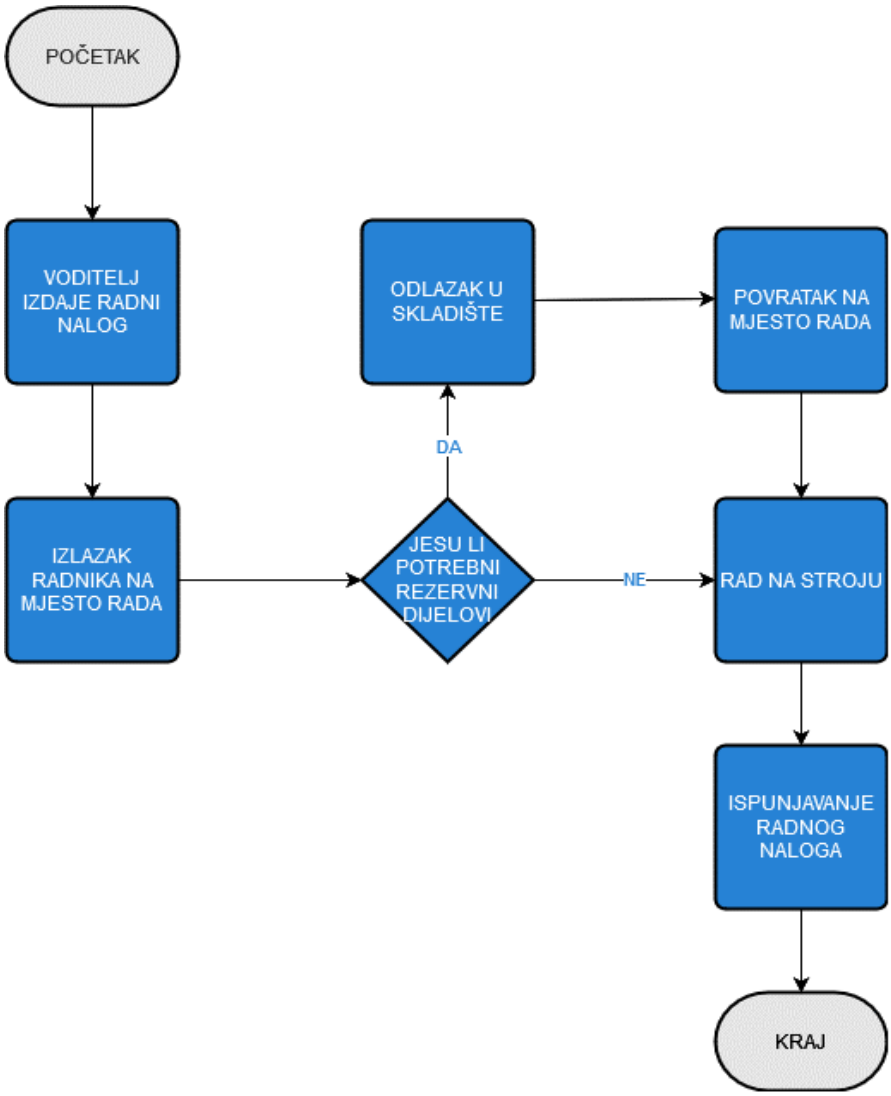
Preventivno održavanje provodi se prema *Planu preventivnog održavanja*, koje se izrađuje krajem svake kalendarske godine za sljedeću godinu. Prva verzija *Plana* šalje se na pregled rukovoditeljima ostalih sektora (Sektor proizvodnje, Sektor skladišta, Sektor financija) i direktoru tvrtke. Nakon pregleda i prijedloga pojedinih sektora, stvara se konačna verzija *Plana* koja postaje važeća.

Plan preventivnog održavanja sadrži sljedeće stavke:

- planirani radovi na opremi (ukoliko je moguće navedeni su i troškovi),
- planirani timovi radnika,
- planirani broj sati rada.

Tijekom preventivnog održavanja, radnici vode evidenciju svih radova na opremi u dokumentu koji se naziva *Evidencija održavanja osnovnog sredstva*. Navedeni dokument služi kao pregled odrađenih poslova u tekućoj godini. Na temelju *Evidencija*, na kraju godine, izrađuje se dokument *Realizacija plana održavanja*, u kojem se jasno vidi u kojoj mjeri je ispunjen *Plan preventivnog održavanja*. Uočeni nedostatci ili neodrađeni poslovi uvrštavaju se u nacrt *Plana preventivnog održavanja* za narednu godinu.

U postupku preventivnog održavanja, Voditelj ili Rukovoditelj izdaje radni nalog u skladu s godišnjim *Planom preventivnog održavanja*. Koraci preventivnog održavanja industrijskog pogona prikazani su na slici 2.1.



Slika 2.1. *Hodogram Preventivnog održavanja*

2.3. Korektivno održavanje

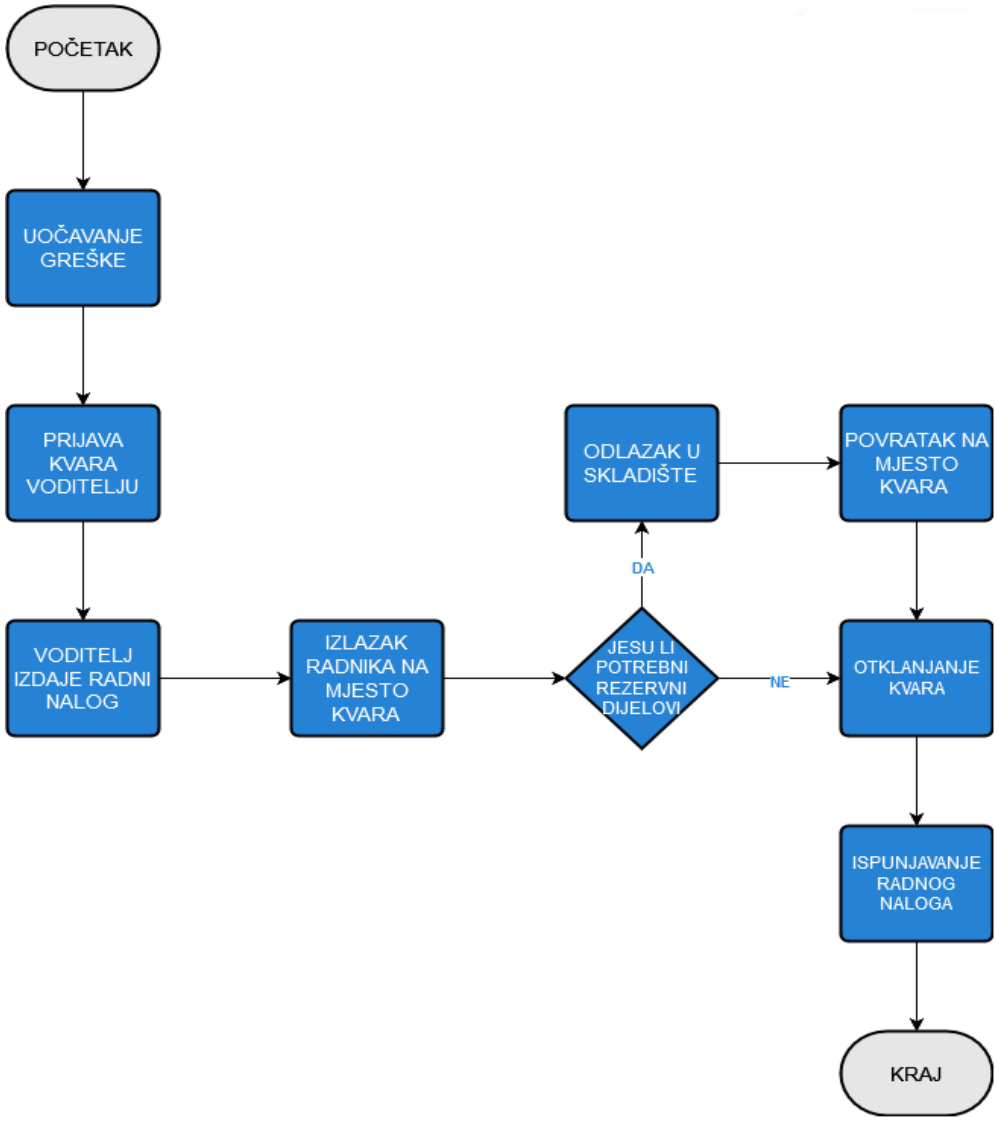
Korektivno održavanje odnosi se na popravak neispravnog dijela opreme, strojeva ili uređaja čije se odstupanje od pravilnog načina rada uočava tijekom proizvodnje neke robe ili tijekom obavljanja bilo kakve vrste posla. Uočenu nepravilnost prijavljuje korisnik opreme, tj. radnik koji je u tom trenutku radio sa određenim strojem.

Radnik kvar usmeno prijavljuje svom Voditelju proizvodnje, koji nakon zaprimljene prijave popunjava obrazac *Prijava kvara.* Popunjeni obrazac dostavlja Voditelju održavanja elektronskim putem, a u hitnim slučajevima telefonskim pozivom. Obrazac sadržava datum prijave kvara, oznaku linije na kojoj se kvar dogodio, evidencijski broj osnovnog sredstva i opis kvara. Nakon

što je zahtjev zaprimljen, Voditelj održavanja izrađuje radni nalog te ga dodjeljuje radnicima osposobljenima za tu vrstu kvara.

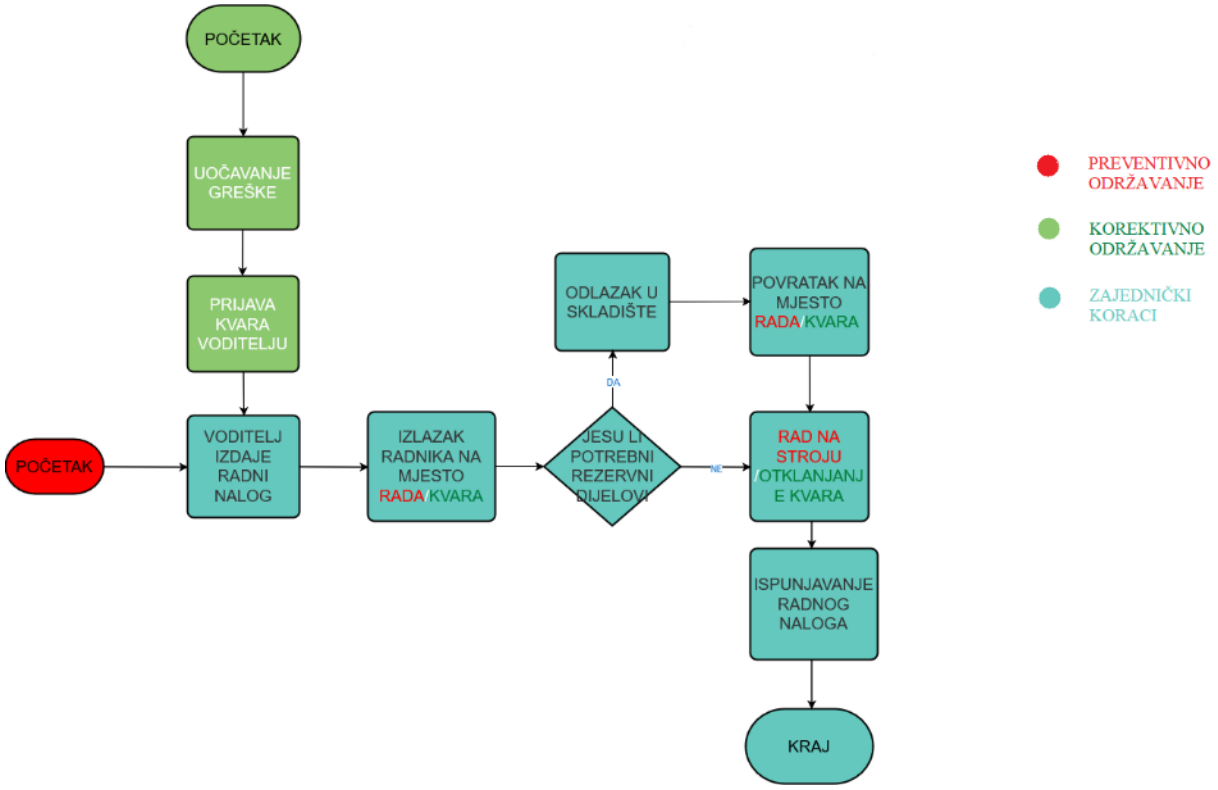
Radnik kojem je dodijeljen radni nalog, odlazi na mjesto kvara. Radnik je dužan poštovati pravila ponašanja u pogonu, kao i pravila struke. Nakon otklonjenog kvara, radnik je dužan ostaviti radno mjesto čisto i u skladu s propisima. Ukoliko postoji potreba za rezervnim dijelovima, radnik iste podiže u skladištu rezervnih dijelova. Ako potrebnih dijelova nema u skladištu, radnik daje specifikaciju dijelova skladištaru, koji naručuje dijelove. Nakon što dijelovi stignu i zaprime se u skladištu, radnik ih podiže i nastavlja s popravkom kvara.

Po završetku rada i otklanjanja kvara, radnik popunjava radni nalog te ga popunjenog vraća Voditelju održavanja. Koraci korektivnog održavanja prikazani su na slici 2.2.



Slika 2.2. *Hodogram Korektivnog održavanja*

Osnovna razlika između preventivnog i korektivnog održavanja jest u tome što korektivno održavanje treba svesti na minimum. Navedeno se postiže kvalitetno odrađenim preventivnim održavanjem. Slika 2.3. prikazuje razlike između ove dvije vrste održavanja industrijskog pogona.



Slika 2.3. *Sličnosti i razlike preventivnog i korektivnog održavanja*

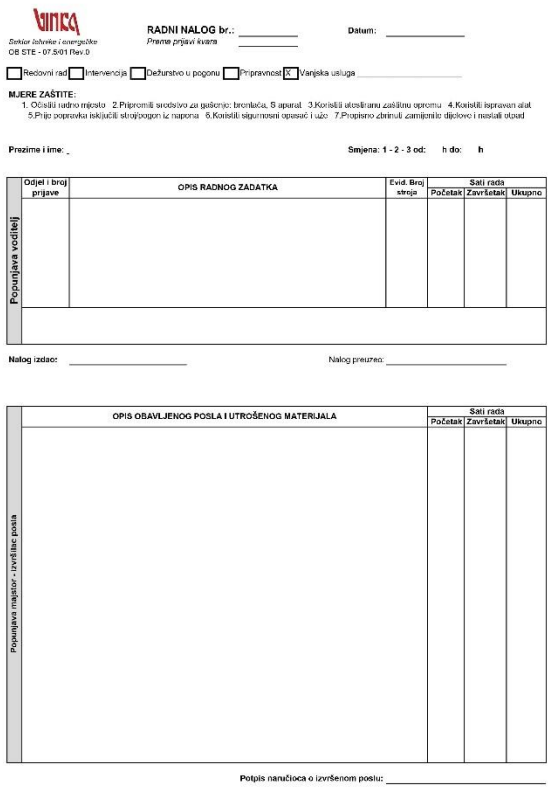
2.4. Radni nalog u drugim tvrtkama

Svaka tvrtka ima vlastiti proces izdavanja radnog naloga. Osnova izrade radnog naloga slična je u većini tvrtki uz određene specifičnosti. Primjerice, tvrtka *Pupilla d.o.o., Zagreb,* prema [2], u radni nalog unosi podatke o proizvodu, količinama i informacije o skladištu gdje će se proizvod smjestiti nakon što bude gotov. Nakon definiranja radnog naloga, pomoću dva dokumenta: „Izdatnica materijala“ i „Izdatnica poluproizvoda“, u proces se unose informacije o materijalima i ostalim proizvodima koji će se koristiti u procesu. Tijekom proizvodnje, uz jedan radni nalog moguće je napraviti više različitih izdatnica. Nakon proizvodnog procesa, na temelju podataka o iskorištenim sirovinama, radni nalog izračunava ukupnu nabavnu vrijednost utrošenog materijala.

Sljedeći primjer je tvrtka *IBM*, koja, prema [3]**,** opisuje sedam koraka u izradi radnog naloga. Prvi korak je identifikacija zadataka koji se moraju obaviti. Drugi korak je stvaranje upita za posao

(eng. *work request*) koji se šalje timu za održavanje ili nadređenoj osobi kako bi se zahtjev odobrio. Treći korak je evaluacija zahtjeva. Ukoliko upit bude odobren, u ovom koraku se upit za posao pretvara u radni nalog. Sljedeći korak je izrada radnog naloga, tj. informacija vezanih za posao koji se mora odraditi. Nakon toga, odabire se kvalificirano osoblje za odrađivanje zadatka. Šesti korak se odnosi na zatvaranje radnog naloga, ukoliko je sav posao obavljen i ukoliko su popunjeni svi podaci koji se odnose na obavljeni posao. Zadnji korak je pregled i analiza završenog Radnog naloga**.**

Slika 2.4. prikazuje konkretne primjere aktivnih radnih naloga pojedinih tvrtki. Programsko rješenje izrađeno u sklopu završnog rada bit će primjenjivo na navedene radne naloge.



RADNI NALOG br.: _____ Datum: _____

Prema prijavi kvara _____

Redovni rad | Intervencija | Dežurstvo u pogonu | Pripravnost [X] | Vanjska usluga _____

MJERE ZAŠTITE:

1. Očistiti radno mjesto 2.Pripremiti sredstvo za gašenje: brentača, S aparat 3.Koristiti utisnirani zaštitnu opremu 4.Koristiti ispravan alat 5.Prije popravka isključiti strojpogon iz napona 6.Koristiti sigurnosni opasač i uže 7.Propisno zbrinuti zamijenjene dijelove i masleni otpad

Prezime i ime: _ Smjena: 1 - 2 - 3 od: h do: h

Popunjava voditelj

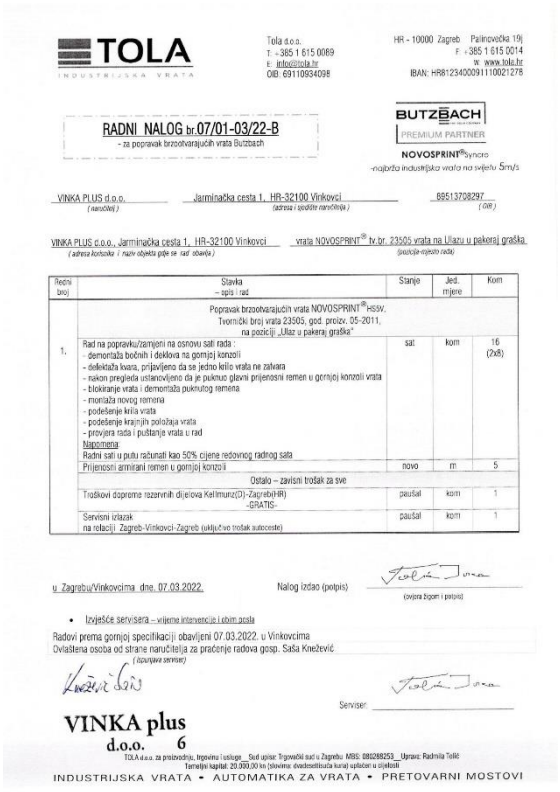
Odjel i broj prijave	OPIS RADNOG ZADATKA	Evid. Broj stroja	Sati rada		
			Početak	Završetak	Ukupno

Nalog izdao: _____ Nalog preuzeo: _____

Popunjava majstor - izvršitac posla

OPIS OBAVLJENOG POSLA I UTROŠENOG MATERIJALA	Sati rada		
	Početak	Završetak	Ukupno

Potpis naručioca o izvršenom poslu: _____



TOLA
INDUSTRIJSKA VRATA

Tola d.o.o.
T: +385 1 615 0089
E: info@tola.hr
OIB: 69110934098

HR - 10000 Zagreb Palinovečka 19j
F: +385 1 615 0014
W: www.tola.hr
IBAN: HR8123400091110021278

RADNI NALOG br.07/01-03/22-B
- za popravak brzootvarajućih vrata Butzbach

BUTZBACH
PREMIUM PARTNER
NOVOSPRINT® synchro
-najbrža industrijska vrata na svijetu 5m/s

VINKA PLUS d.o.o. (naručitelj) Jarmintačka cesta 1, HR-32100 Vinkovci (adresa i sjedište naručitelja) 88513708297 (OIB)

VINKA PLUS d.o.o., Jarmintačka cesta 1, HR-32100 Vinkovci (adresa korisnika i naziv objekta gdje se rad obavlja) vrata NOVOSPRINT® tv.br. 23505 vrata na Ulazu u pekarej graška (predmet-vrijeme rada)

Redni broj	Stavka – opis i rad	Stanje	Jed. mjere	Kom
	Popravak brzootvarajućih vrata NOVOSPRINT® HSSV, Tvornički broj vrata 23505, god. proizv. 05-2011, na poziciji „Ulaz u pekaraj graška"			
1.	Rad na popravku/zamjeni na osnovu sati rada : - demontaža bočnih i deklova na gornjoj konzoli - detektaža kvara, prijavljeno da se jedno krilo vrata ne zatvara - nakon pregleda ustanovljeno da je puknuo glavni prijenosni remen u gornjoj konzoli vrata - blokiranje vrata i demontaža puknutog remena - montaža novog remena - podešenje krila vrata - podešenje krajnjih položaja vrata - provjera rada i puštanje vrata u rad Napomena: Radni sati u putu računati kao 50% cijene redovnog radnog sata	sat	kom	16 (2x8)
	Prijenosni armirani remen u gornjoj konzoli	novo	m	5
	Ostalo – zavisni trošak za sve			
	Troškovi dopreme rezervnih dijelova Kelmuniz(D)-Zagreb(HR) -GRATIS-	paušal	kom	1
	Servisni izlazak na relaciji Zagreb-Vinkovci-Zagreb (uključivo trošak autoceste)	paušal	kom	1

u Zagrebu/Vinkovcima dne. 07.03.2022. Nalog izdao (potpis) _____ (ovjera žigom i potpis)

- Izvješće servisera – vrijeme intervencije i obim posla

Radovi prema gornjoj specifikaciji obavljeni 07.03.2022. u Vinkovcima
Ovlaštena osoba od strane naručitelja za praćenje radova gosp. Saša Knežević
(ispunjava serviser)

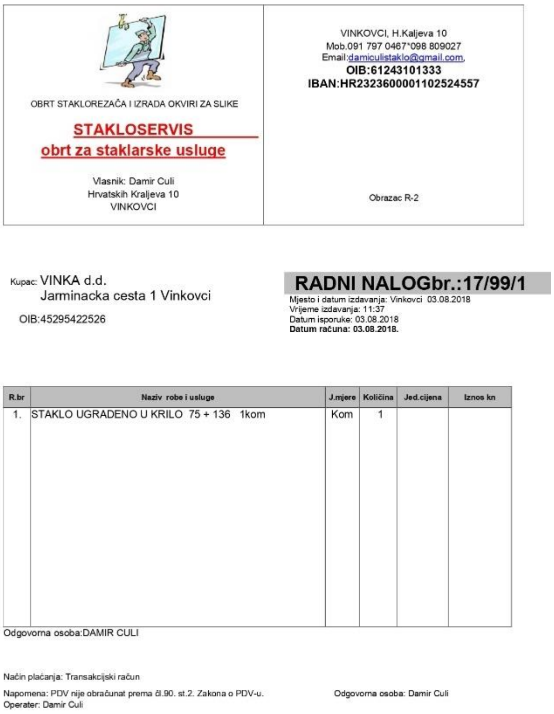
Serviser: _____

VINKA plus d.o.o. 6

TOLA d.o.o. za proizvodnju, trgovinu i usluge Sud upisa: Trgovački sud u Zagrebu MBS: 080288253 Uprava: Radmila Tolić Temeljni kapital: 20.000,00 kn (slovima: dvadesettisuća kuna) uplaćen u cijelosti

INDUSTRIJSKA VRATA • AUTOMATIKA ZA VRATA • PRETOVARNI MOSTOVI

a) Radni nalog *Vinke d.o.o.* **b)** Radni nalog firme *Tola d.o.o.*


OBRT STAKLOREZAČA I IZRADA OKVIRI ZA SLIKE

STAKLOSERVIS
obrt za staklarske usluge

Vlasnik: Damir Culi
Hrvatskih Kraljeva 10
VINKOVCI

VINKOVCI, H.Kaljeva 10
Mob.091 797 0467*098 809027
Email:damiculistaklo@gmail.com,
OIB:61243101333
IBAN:HR2323600001102524557

Obrazac R-2

Kupac: VINKA d.d.
Jarminacka cesta 1 Vinkovci

OIB:45295422526

RADNI NALOGbr.:17/99/1
Mjesto i datum izdavanja: Vinkovci 03.08.2018
Vrijeme izdavanja: 11:37
Datum isporuke: 03.08.2018
Datum računa: 03.08.2018.

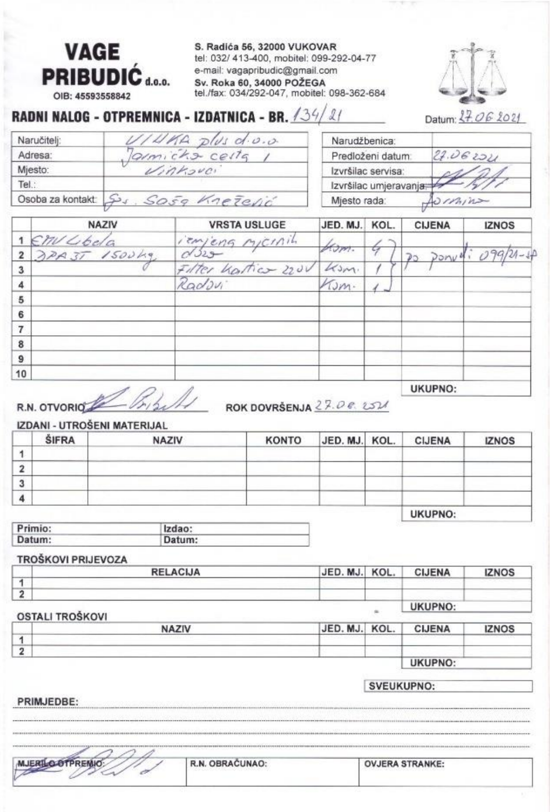
R.br	Naziv robe i usluge	J.mjere	Količina	Jed.cijena	Iznos kn
1.	STAKLO UGRAĐENO U KRILO 75 + 136 1kom	Kom	1		

Odgovorna osoba:DAMIR CULI

Način plaćanja: Transakcijski račun

Napomena: PDV nije obračunat prema čl.90. st.2. Zakona o PDV-u.
Operater: Damir Culi

Odgovorna osoba: Damir Culi


VAGE PRIBUDIĆ d.o.o.
OIB: 45593558842

S. Radića 56, 32000 VUKOVAR
tel: 032/ 413-400, mobitel: 099-292-04-77
e-mail: vagapribudic@gmail.com
Sv. Roka 60, 34000 POŽEGA
tel./fax: 034/292-047, mobitel: 098-362-684

RADNI NALOG - OTPREMNICA - IZDATNICA - BR. 134/21 Datum: 27.06.2021

Naručitelj:	VINKA plus d.o.o.	Narudžbenica:	
Adresa:	Jarmička cesta 1	Predloženi datum:	27.06.2021
Mjesto:	Vinkovci	Izvršilac servisa:	
Tel.:		Izvršilac umjeravanja:	
Osoba za kontakt:	g. Saša Knežević	Mjesto rada:	Jarmina

	NAZIV	VRSTA USLUGE	JED. MJ.	KOL.	CIJENA	IZNOS
1	EMV Libela	izmjena mjernih	Kom.	4	po ponudi: 099/21-VP	
2	DPA 3T 1500kg	doza				
3		Filter kartica 220V	Kom.	1		
4		Radovi	Kom.	1		
5						
6						
7						
8						
9						
10						
					UKUPNO:	

R.N. OTVORIO ROK DOVRŠENJA 27.06.2021

IZDANI - UTROŠENI MATERIJAL

	ŠIFRA	NAZIV	KONTO	JED. MJ.	KOL.	CIJENA	IZNOS
1							
2							
3							
4							
						UKUPNO:	

Primio:	Izdao:
Datum:	Datum:

TROŠKOVI PRIJEVOZA

	RELACIJA	JED. MJ.	KOL.	CIJENA	IZNOS
1					
2					
				UKUPNO:	

OSTALI TROŠKOVI

	NAZIV	JED. MJ.	KOL.	CIJENA	IZNOS
1					
2					
				UKUPNO:	

SVEUKUPNO:

PRIMJEDBE:

MJERILO OTPREMIO: R.N. OBRAČUNAO: OVJERA STRANKE:

c) Radni nalog obrta *Stakloservis* **d)** Radni nalog firme *Vage Pribudić*

Slika 2.4. *Radni nalozi različitih tvrtki*

2.5. Zahtjevi na programsko rješenje

Na temelju prethodno opisanih procedura za izradu radnih naloga kao i njihovih specifičnosti u ovom potpoglavlju bit će definirani zahtjevi na programsko rješenje – mobilne aplikacije za upravljanje radnim nalozima.

2.5.1. Opis značajki aplikacije

Prva tražena značajka aplikacije jest mogućnost izrade korisničkih računa i mogućnost prijave, tj. provjere identiteta. Tijekom registracije, korisnik mora navesti i svoje radno mjesto unutar tvrtke, na temelju kojeg će na samom kraju izrade računa odabrati odgovara li to radno mjesto voditelju (rukovoditelju, direktoru ili naručitelju usluge) ili radniku (zaposleniku, djelatniku ili izvršitelju usluge).

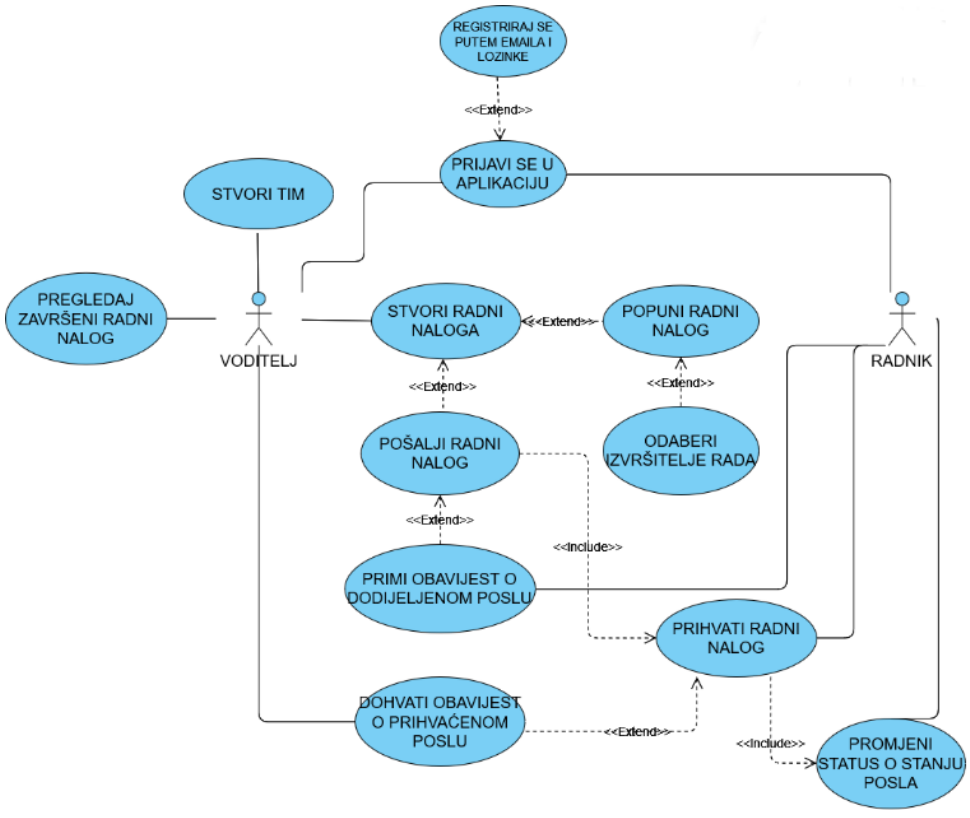
Korisnik koji je prijavljen kao voditelj, treba imati mogućnost izrade novog radnog naloga čiji će predložak već postojati, te će on morati samo popuniti polja koja se od njega traže. Aplikacija treba omogućiti voditelju odabir jednog ili više radnika putem email adrese. Ukoliko

postoji potreba za izdavanjem radnog naloga nekoj drugoj tvrtki ili nekom vanjskom suradniku, voditelj također mora moći odabrati tvrtku ili suradnika preko email adrese. Voditelj mora imati mogućnost stvaranja timova i podjele radnika u odgovarajuće timove.

Kada se radnik prijavi u aplikaciju, treba imati mogućnost pregleda dodijeljenih radnih naloga. Radnik mora označiti radni nalog s jednim od tri dostupna statusa: prihvaćeno, u tijeku i završeno. Status naloga mora biti vidljiv voditelju, kako bi znao u kojoj fazi je zadani posao. Također, kada se radniku dodijeli posao, on mora dobiti obavijest o zadanom poslu. Voditelj mora biti obaviješten svaki puta kada radnik promijeni status radnog naloga.

Svi izdani nalozi trebaju biti spremljeni u bazu podataka, tj. repozitorij koji služi kao spremište radnih naloga. Kada je nalog završen, on više ne treba biti prikazan među aktivnim nalozima, već unutar odjeljka pod nazivom Završeni radni nalozi. Također, svaki prijavljeni korisnik treba imati mogućnost pregleda dobivenih obavijesti i timova. Isto tako, aplikacija treba omogućiti korisniku uređivanje vlastitog profila u svakom trenutku.

Detaljniji prikaz svih potrebnih funkcionalnosti dostupan je na slici 2.5. koja prikazuje standardiziran oblik specifikacije zahtjeva, točnije dijagram slučaja korištenja (eng. *Use Case Diagram*).



Slika 2.5. Dijagram slučajeva korištenja

3. PREGLED POSTOJEĆIH PROGRAMSKIH RJEŠENJA

Kako su radni nalozi vrlo bitna stavka u svakoj tvrtki, tako je postojanje njihovog elektroničkog oblika uvelike olakšalo podjelu istih te tako ubrzalo voditeljima raspodjelu posla. U ovom poglavlju bit će prikazana već postojeća programska rješenja koja omogućuju izradu radnih naloga u elektroničkom obliku. Većina aplikacija koje će biti navedene imaju i druge mogućnosti, no ovdje će pozornost biti usmjerena samo na dio koji se odnosi na radne naloge. Neka od postojećih programskih rješenja poslužila su kao inspiracija za izradu aplikacije u praktičnom dijelu ovog rada. Aplikacije koje se bave ovom tematikom uglavnom nisu dostupne u obliku mobilne aplikacije i na hrvatskom jeziku, što aplikacija izrađena u sklopu ovog rada nudi kao glavnu nadmoćnu karakteristiku u odnosu na spomenute aplikacije.

Neke od dostupnih aplikacija koje su opisane u daljnjem tekstu su: Moja tvrtka, UpKeep, MaintainX, Fracttal, Limble CMMS, Synchroteam, Fiix i Wrike. Također, dostupne aplikacije mogle bi se neformalno kategorizirati u par skupina:

1. Aplikacije koje podržavaju rad bez internetske veze:
    - Wrike,
    - Synchroteam.
2. Aplikacije temeljene za CMMS (održavanje uz pomoć računalnih sustava) načinu rada:
    - Limble CMMS,
    - MaintainX,
    - Fracttal,
    - UpKeep.
3. Aplikacije koje omogućavaju slanje push-obavijesti:
    - UpKeep,
    - Moja Tvrtka,
    - Fiix.
4. Aplikacije koje su dostupne u web i mobilnoj verziji:
    - UpKeep,
    - MaintainX,
    - Fracttal,
    - Wrike.

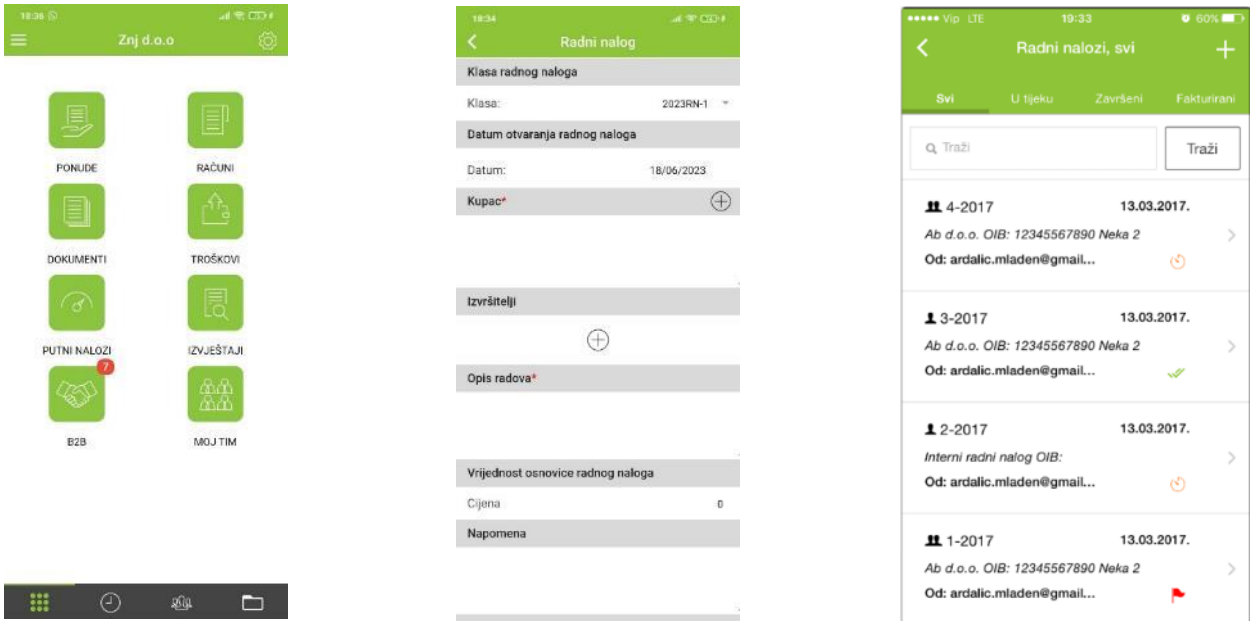
3.1. Moja tvrtka

Aplikacija Moja Tvrtka osim izrade novog radnog naloga nudi i ostale mogućnosti, kao što su stvaranje platnih računa, izvještaja, praćenje troškova, izdavanje putnih naloga, izrada različitih dokumenata i slično. Kada se otvori odgovarajući odjeljak (na slici 3.1.a. *Moj tim*), prikazuje se opcija *Radni nalog* te se klikom na istu prikazuju svi postojeći radni nalozi, koji su zatim grupirani u tri kategorije: U tijeku, Završeni, Fakturirani. Vizualizacija prikaza svih vrsta radnih naloga prikazana je na slici 3.1.c. Pritiskom na jedan radni nalog otvara se detaljan prikaz svega što je sadržano u radnom nalogu. Početna stranica još nudi i mogućnost pretrage radnih naloga i izrada novog. Prilikom izrade novog radnog naloga, unose se sljedeći podaci:

- Klasa radnog naloga,
- Datum otvaranja radnog naloga,
- Kupac,
- Izvršitelj,
- Opis radova,
- Vrijednost osnovice radnog naloga,
- Napomena.

Na slici 3.1.b. prikazan je izgled sučelja pri izradi radnog naloga. Jednom kada je radni nalog stvoren, automatska *push* obavijest se pojavljuje na mobilnim uređajima svih korisnika koji su navedeni kao izvršitelji tog radnog naloga. Navedeni korisnici zatim mogu odabrati status koji odgovara trenutnom stanju posla: U tijeku ili Završen. Nakon odabira statusa radnog naloga, vlasnici dobivaju obavijest o njemu, prema [4].

Međutim, obzirom na postavljene zahtjeve na programsko rješenje, vidljivo je da aplikacija *Moja tvrtka* ne nudi mogućnost praćenja svih potrebnih dijelova radnog naloga, kao što su utrošeni materijali te vrijeme koje je bilo potrebno za obavljanje posla opisanog u radnom nalogu. Ovo je upravo jedna od funkcionalnosti koje nudi aplikacija izrađena u okviru ovog rada. Ova mogućnost osigurava naručitelju usluge bolji uvid u detalje vezane za izdani dokument.



a) *Početna stranica* **b)** *Izrada radnog naloga* **c)** *Svi nalozi*

Slika 3.1. *Aplikacija Moja tvrtka*

3.2. UpKeep

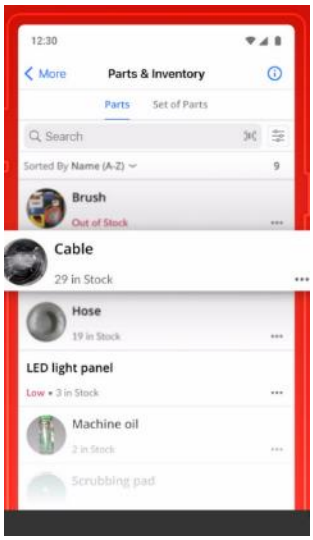
Aplikacija pod nazivom UpKeep nudi mobilnu i web aplikaciju svojim korisnicima s ciljem digitalizacije radnih naloga i ostalih dokumenata unutar tima zaposlenika. Omogućuje izdavanje radnih naloga i njihovo momentalno ažuriranje. Prilikom izrade radnih naloga, implementirana je i mogućnost prilaganja fotografije oštećene robe, stroja ili dijelova kako bi se lakše i preciznije ukazalo na problematiku zadatka koji će biti dodijeljen nekom tehničaru. Slično kao i prethodna aplikacija, radni nalog se sastoji od datuma do kojeg se problem mora riješiti, izvršitelja radova, detalja posla, lokacije i slično. Ova aplikacija još nudi i skeniranje bar-kodova koji se nalaze na alatu ili stroju kako bi ih se lakše dodijelilo nekom radnom nalogu. Sortiranje radnih naloga prema datumu, vremenu ažuriranja, statusu, lokaciji te prioritetu, pretraživanje prijašnjih radnih naloga, praćenje vremena utrošenog na popravak, slanje *push* obavijesti za praćenje timskog napretka neke su od opcija koje ovaj softver nudi.

Na slici 3.2. a. – d., prikazan je izgled mobilne verzije aplikacije i to redom: broj izvršenih i postotak odrađenih naloga u zadanom roku, pregled stanja alata i inventara na skladištu, prikaz obavijesti i popisa radnih naloga s naznačenim statusom, prema [5].

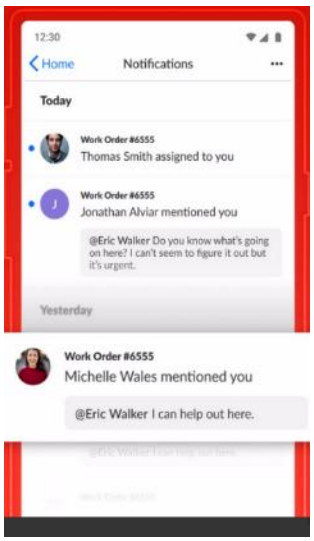
Ova aplikacija, iako vrlo pogodna za izradu radnih naloga, ne omogućuje naručitelju radnog naloga (voditelju) stvaranje timova zaposlenika, što je jedan od zahtjeva na programsko rješenje

koje je postavljeno i implementirano u novostvorenoj aplikaciji, a ono omogućava lakši odabir ispravnog izvršitelja rada za određeni posao.

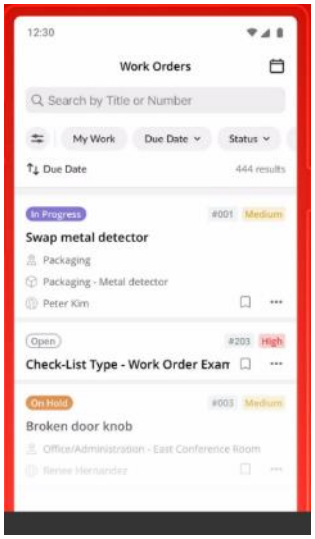
a) *Broj izvršenih naloga*



b) *Stanje na skladištu*



c) *Obavijesti*



d) *Radni nalozi*

Slika 3.2. *Aplikacija UpKeep*

3.3. MaintainX

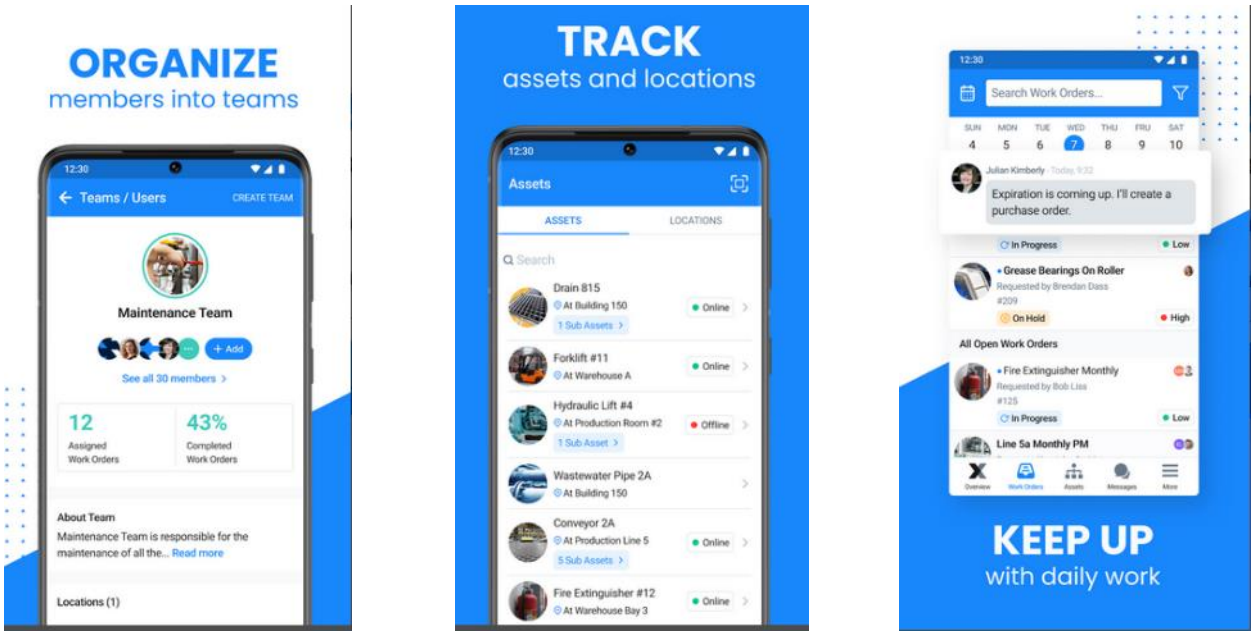
Još jedna aplikacija namijenjena za android i web jest MaintainX. MaintainX je aplikacija za izdavanje radnih naloga koji pomaže timovima u tvornicama zaduženim za obavljanje posla što točno trebaju napraviti i na koji način, koristeći prilikom toga sigurnosne mjere i alate. MaintainX

dostupan je za sve vrste tvrtki, bilo da se radi o restoranima, ugostiteljstvu općenito, upravljanju nekretninama, edukacijskim ustanovama, industrijskim postrojenjima ili proizvodnji. Prednosti korištenja ove aplikacije, prema [6], su:

- Izrada formi radnih naloga koje se kasnije koriste kao predlošci za nove naloge,
- Neograničen broj fotografija koje se mogu priložiti za lakše ilustriranje problema,
- Dostupan za sve platforme: web, iOS, Android, iPad,
- Organiziranje članova u timove,
- Implementiran chat za interakciju članova tima kojim se postiže efikasniji tijek rada i veća organizacija i informiranost svih radnika,
- Automatska ažuriranja statusa kako bi voditelj ili naručitelj zahtjeva bio informiran o stanju rada u svakom trenutku,
- Praćenje napretka u stvarnom vremenu,
- Sortiranje poslova po prioritetu i hitnosti,
- Izrada preventivnih radnih naloga.

Na slici 3.3.a. prikazano je kako izgleda sučelje za stvaranje tima, slika 3.3.b. prikazuje praćenje svih dobara i lokacija, a na slici 3.3.c. prikazani su napravljeni radni nalozi.

Iako ova aplikacija ima puno mogućnosti, njen nedostatak je što nije dostupna na hrvatskom jeziku. Upravo taj problem rješava Android aplikacija opisana u ovom radu.



a) *Formiranje tima* **b)** *Praćenje dobara* **c)** *Lista naloga*

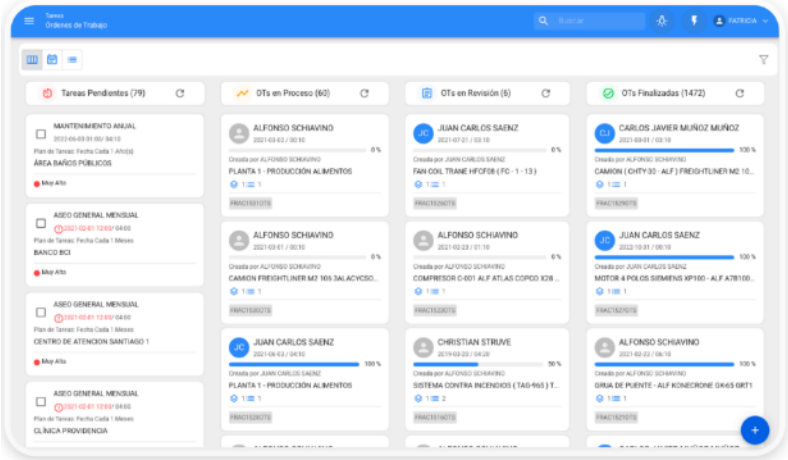
Slika 3.3*. Aplikacija MaintainX*

3.4. Fracttal

Fracttal je aplikacija temeljena na održavanju uz pomoć računalnih sustava (engl. *CMMS – Computerized maintenance management system*) s funkcijom upravljanja radnim nalozima. Aplikacija Fracttal pomaže u planiranju i izdavanju svih radnih naloga radnicima kojima je posao dodijeljen. Nadzorna ploča (engl. *Dashboard*) prikazuje sve neraspoređene radne naloge i omogućava korisniku (voditelju) da ih dodijeli svojim radnicima ili drugoj tvrtki. Voditelj također može putem ove aplikacije pružiti specifične detalje i upute koje se moraju slijediti, kao što su alati, usluge i procesi koji trebaju biti korišteni. Omogućuje slanje pravovremenih ažuriranja o statusu radnih naloga putem emaila, WhatsAppa, Messengera ili bilo koje druge suradničke aplikacije. Fracttal nudi rad s kanban pločom, koja prema [7], vizualno prikazuje rad u različitim fazama procesa pomoću kartica koje predstavljaju radne stavke i stupaca koji predstavljaju svaku fazu procesa. Vremenski pregled (engl. *timeline view*) omogućuje raspodjelu radnih naloga i praćenje statusa aktivnog zadatka. S Fracttal aplikacijom moguće je pratiti metrike radnih naloga u stvarnom vremenu i generirati pametne izvještaje bez gubljenja vremena na pravljenje ručnih proračunskih tablica.

Na slici 3.4. vidljivo je korisničko sučelje web verzije ove aplikacije na kojem su prikazani svi aktivni radni nalozi, prema [8].

Nedostatak ove aplikacije je to što ne nudi mogućnost uređivanja profila korisnika, što je također jedan od zahtjeva na programsko rješenje ovog rada, a koja značajno olakšava odabir radnika za stvaranje timova i za međusobnu komunikaciju.



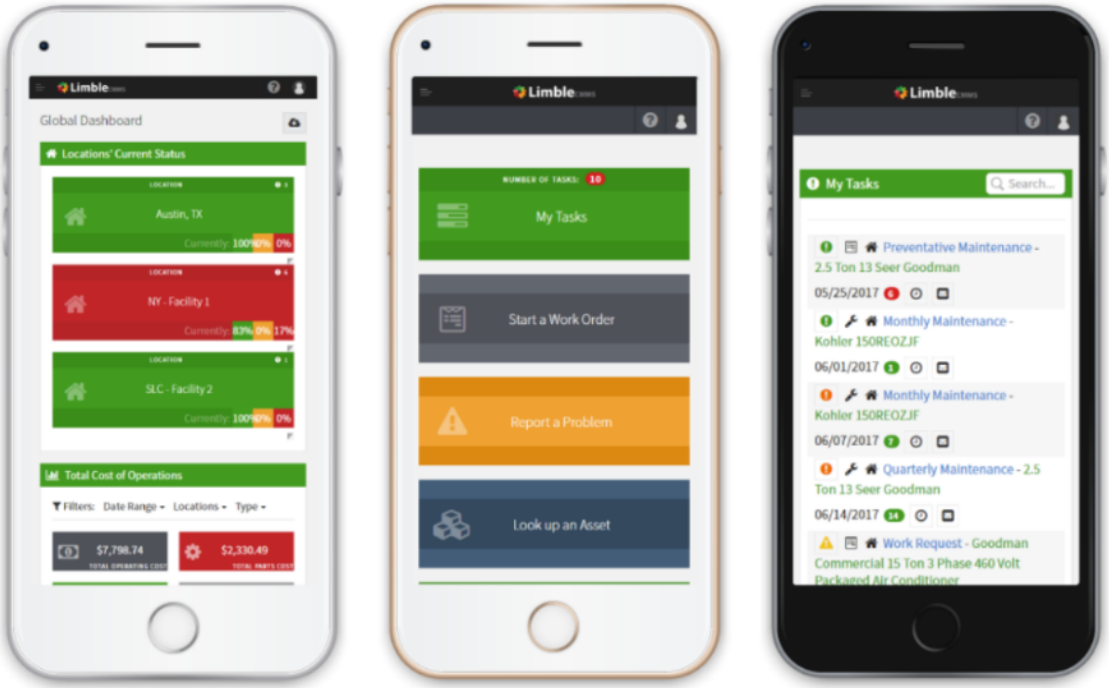
Slika 3.4.*Fracttal - radni nalozi*

3.5. Limble CMMS

Aplikacija za izradu radnih naloga, upravljanje istima te praćenje opreme u skladištu neke tvrtke. Limble CMMS osmišljena je kako bi pojednostavila poslovne operacije i zabilježila povijest rada. Sa svojom centraliziranom bazom podataka omogućuje voditeljima i rukovoditeljima odjela održavanja jednostavnije organiziranje radnih naloga, preventivno održavanje i druge zadatke. Također omogućuje dodjeljivanje zadataka timu ili pojedinačnom korisniku s detaljnim uputama. Korisnici mogu ostavljati komentare, priložiti slike, slati obavijesti i email poruke radi poboljšanja komunikacije vezane uz radni nalog. Limble CMMS pruža korisnicima postavljanje prioriteta zadataka i planiranje trenutnih i nadolazećih zadataka. Ova aplikacija ima implementiranu mogućnost praćenja podataka o održavanju resursa (alata, strojeva, postrojenja), kao i o učestalosti održavanja, vrijeme potrebno za održavanje i ukupni trošak održavanja. Prema [9], praćenje ovih informacija pomaže u planiranju preventivnog održavanja.

Limble CMMS još je jedna aplikacija koja dobro odgovara na postavljeni problem upravljanja radnim nalozima, ali ono što nedostaje ovdje je prikaz dobivenih obavijesti filtriranih na Pročitane i Nepročitane. Upravo zbog toga, aplikacija izrađena u okviru ovog rada ispravlja taj problem na način što implementira spomenutu značajku.

Slika 3.5. prikazuje tri različite mogućnosti ove aplikacije.



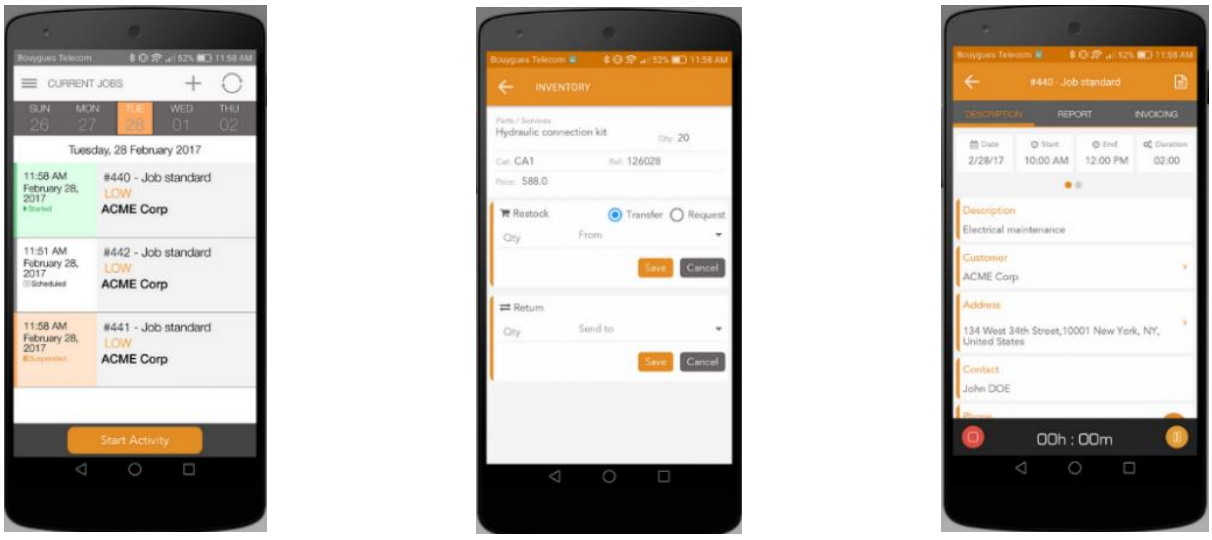
Slika 3.5. *Limble CMMS - izgled korisničkih sučelja aplikacije*

3.6. Synchroteam

Synchroteam aplikacija omogućuje podjelu informacija o poslu i zadataka koje je potrebno odraditi između radnika koji su umreženi upravo putem ove aplikacije. Ova multifunkcionalna aplikacija može se pohvaliti ugrađenom bazom podataka koja radi bez obzira na kvalitetu internetske mreže korisnika. Šifriranje podataka i integritet transakcija održavaju se na visokoj razini čak i kada je veza s mrežom izgubljena. Dio aplikacije zbog kojeg se ona i nalazi na ovom popisu postojećih programskih rješenja je učinkovito upravljanje radnim nalozima. Pomoću aplikacije, informacije o napravljenom Radnom nalogu mogu se pregledati prije početka posla. Aplikacija također pruža interaktivne značajke kao što su trenutne upute za obavljanje nekog zadatka, obavljanje telefonskog poziva jednim pritiskom na tipku i mogućnost pregleda opisa posla i izvještaja. Upravljanje postojećim radnim nalozima vrlo je intuitivno, jer se ažuriranje poslova događa u stvarnom vremenu te se statusi radnog naloga označavaju logičnim oznakama: današnji, nadolazeći, zakašnjeli i završeni. Synchroteam nudi sposobnost pregleda prethodnih radnih naloga. Omogućeno je stvaranje, promjena rasporeda ili odbijanje poslova. Pristupanje privitcima povezanih s poslom ili klijentima, aktivacija ili deaktivacija automatske sinkronizacije još su neke od mogućnosti koje ova aplikacija pruža, prema [10].

Aplikacija Synchroteam ne nudi različito korisničko sučelje ovisno o poziciji unutar tima ili tvrtke. Ova značajka jedna je od onih koje aplikacija izrađena u sklopu ovog rada ima te na taj način olakšava snalaženje korisnika prilikom njenog korištenja.

Izgled Synchroteam aplikacije prikazan je na slici 3.6.



a) *Trenutni poslovi* **b)** *Skladište* **c)** *Detalji posla*

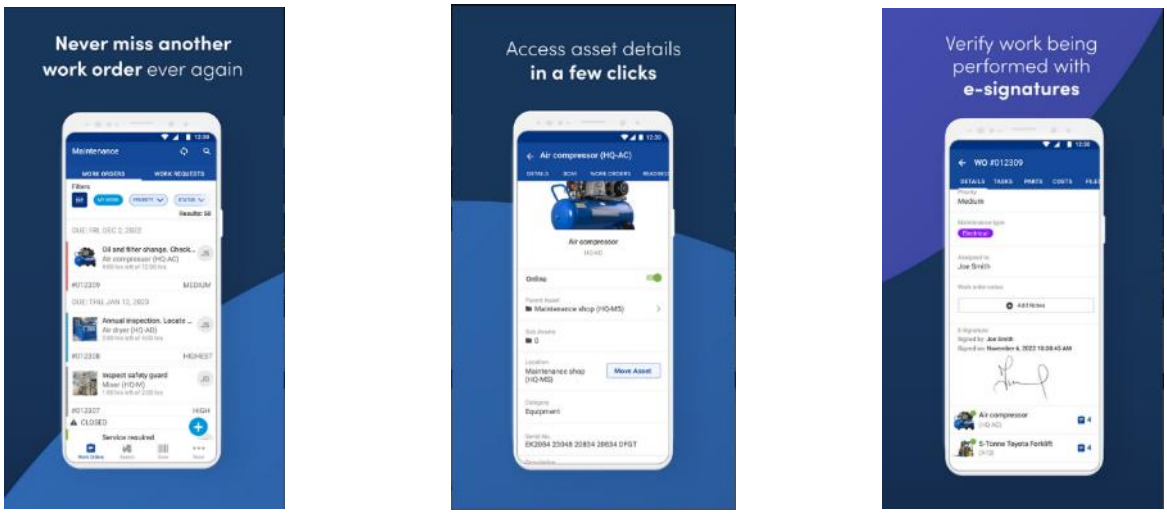
Slika 3.6. *Aplikacija Synchroteam*

3.7. Fiix

Aplikacija Fiix omogućava jednostavnu izradu, pronalaženje i dodijeljivanje radnih naloga putem mobitela. Korisnici koji izrađuju radne naloge mogu dizajnirati vlastitu formu, tj. predložak koji će zadovoljiti njihove potrebe. Prilikom izrade, također mogu iskoristiti mogućnost stvaranja obaveznih polja, koje radnik mora obavezno popuniti kako se ne bi izgubile ključne informacije o poslu. Zaposlenik zaprima *push* obavijest u trenutku kada mu je dodijeljen posao, nakon čega sve svoje poslove (tj. radne naloge) može poredati po prioritetu ili nekom drugom ponuđenom filteru. Fiix aplikacija također nudi svojim korisnicima mogućnost prilaganja fotografija unutar radnog naloga ili zapisniku o resursima, prema [11].

Međutim, ova aplikacija ne omogućuje svojim korisnicima podjelu radnika u timove, što je jedan od zahtjeva na programsko rješenje u poglavlju 2. Aplikacija izrađena u sklopu ovog rada, između ostalog, odgovara i na ovaj zahtjev.

Izgled Fiix sučelja prikazan je na slici 3.7.a. - c.



a) *Radni nalozi* **b)** *Detalji o resursima* **c)** *E-potpisi*

Slika 3.7. *Aplikacija Fiix*

3.8. Wrike

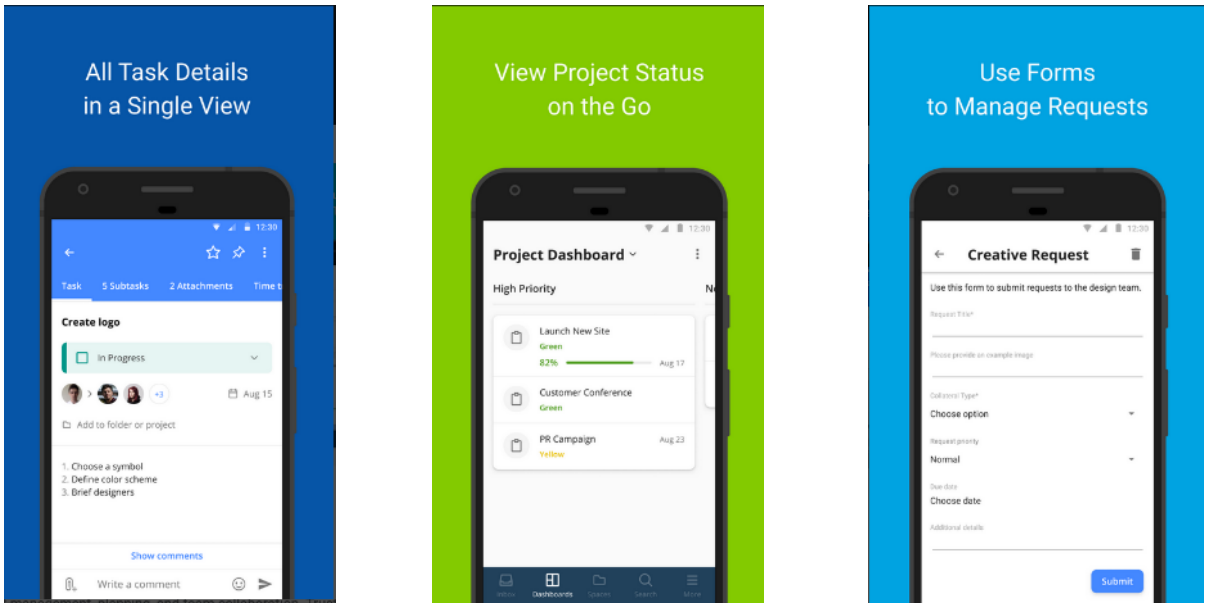
Aplikacija Wrike omogućuje izrada i praćenje radnih naloga, kao i praćenje kompletnih projekata. Bitne značajke ove aplikacije prema [12], su:

- suradnja u stvarnom vremenu sa vlastitim timom i ostalim bitnim korisnicima,

- jednostavno stvaranje zadataka putem postojećih ili novostvorenih obrazaca,
- praćenje projekata i održavanje kontinuiteta posla,
- trenutno pristupanje izvješćima i dokumentima koji su neophodni za nastavak rada,
- napredne sigurnosne značajke kako bi se osigurala sigurnost informacija,
- način rada bez internetske veze,
- dostupno na mnogobrojnim jezicima.

Wrike aplikacija ne odgovara na sve zahtjeve na programsko rješenje definirane u ovom radu. Za razliku od Wrike aplikacije, aplikacija izrađena u okviru ovog rada odgovara na zahtjev slanja obavijesti potrebnim korisnicima. Ova značajka omogućava trenutno saznanje statusa radnog naloga.

Na slici 3.8. prikazano je kako ova aplikacija izgleda u svojoj Android verziji.



a) *Detalji* **b)** *Statusi naloga* **c)** *Forma zahtjeva*

Slika 3.8. *Aplikacija Wrike*

4. TEHNOLOGIJE KORIŠTENE U RADU

Za izradu aplikacije za izdavanje radnih naloga za razvojno okruženje korišten je Android Studio. Odabrani programski jezik za razvoj aplikacije je Kotlin. Razlog korištenja ovog jezika, a ne češće korištenog - Jave jest taj što Kotlin zahtjeva manje programskog koda od Jave za izvođenje istih operacija. Za definiranje korisničkog sučelja aplikacije korišten je XML (engl. *eXtensible Markup Language*) opisni jezik.

Za uspostavu baze podataka korištena je platforma Firebase, iz razloga što ona pruža bitne usluge koje su potrebne za stvaranje jedne Android aplikacije. Jedna od njih je *Cloud Firestore*, *NoSQL* baza koja omogućava pohranu podataka, pretragu baze po upitima kao i mnoge druge pogodnosti zbog kojih je upravo ova usluga i odabrana za izradu ovog rada. Druga Firebase usluga je *Authentication*, značajka koja se koristi kako bi se novi i postojeći korisnici mogli prijaviti ili registrirati u aplikaciju. *Authentication* značajka integrirana je s prvom spomenutom značajkom *Firestore*, što omogućuje vrlo jednostavno korištenje i povezanost baze podataka s aplikacijom.

4.1. Android Studio

Android Studio predstavlja softver, točnije integrirano razvojno okruženje za razvoj aplikacija za operativni sustav Android, prema [13]. Temeljen je na uređivačima koda i razvojnim alatima korištenim u IntelliJ IDEA, prema [14], razvojnom okruženju, dizajniranom za razvoj računalnih softvera napisanih u jezicima Java, Kotlin, Groovy, itd.

Integracija Android Studija s Google Cloud platformom i ostalim Google uslugama pruža mogućnost korištenja brojnih opcija prisutnih na Internetu te njihovu prezentaciju korisnicima preko izrađenih Android aplikacija. Ovaj softver također podržava komunikaciju s popularnim sustavima za verzioniranje, kao što je npr. Git (GitLab, GitHub,…), što uvelike olakšava praćenje promjena unutar koda, kako autoru aplikacije, tako i drugim programerima s kojima se surađuje. Svojim korisnicima Android Studio nudi i napredne alate za dizajniranje korisničkih sučelja, podršku za različite programske jezike, automatsko dovršavanje koda te ugrađeni emulator koji omogućava trenutno testiranje aplikacije ili na virtualnim uređajima ili na vlastitom fizičkom Android uređaju.

4.2. Kotlin

Kotlin je moderan programski jezik iz 2011. godine, a razvila ga je tvrtka JetBrains. Izrađen je za razvoj aplikacija na Androidu, iOS-u te web-u. Zbog svojih prednosti i pogodnosti, postao je jedan od najpopularnijih programskih jezika za razvoj android aplikacija.

Zbog potpune usklađenosti s Java programskim jezikom, moguće je koristiti postojeće Java biblioteke i okvire dok se kodira u Kotlinu, prema [15]. Još jedna prednost, već prethodno navedena, čitljiva je i jednostavna sintaksa, što rezultira manjom količinom koda i manjoj mogućnosti pogreške. Kotlin rješava i problem "*NullPointerException*" grešaka, jer zahtijeva od programera eksplicitno navođenje može li varijabla poprimiti null-vrijednost ili ne. Za razliku od Jave, Kotlin podržava i funkcijsko programiranje, tj. uporabu lambda izraza, obradu kolekcija te korištenje proširenja, prema [16].

4.3. XML

XML (engl. *eXtensible Markup Language*) opisni je jezik koji se koristi za označavanje hijerarhijskih i strukturiranih podataka, u svrhu pohrane i razmjene informacija, prema [17]. Izrađen na način da bude čitljiv i čovjeku i programu, XML nije programski jezik, već jezik za označavanje koji ne sadrži naredbe za izvršavanje. On je neovisan i o računalnoj platformi i o operacijskom sustavu unutar kojeg se koristi, prema [18].

Glavni dijelovi XML-a su elementi i atributi. Svaki element započinje oznakom, npr. '<*TextView*>' i završava zatvarajućom oznakom '</*TextView*>'. Tekst unutar tih elemenata predstavlja vrijednost tog elementa. Elementi mogu imati i podelemente, što dokazuje da se radi o hijerarhijskom jeziku. Atributi se dodaju elementima zbog dodatnog opisivanja. Također, na početku svakog XML dokumenta nalazi se XML deklaracija. Na slici 4.1. prikazan je jedan XML dokument sa deklaracijom, jednim elementom 'shape', dva podelementa 'gradient' i 'corners' te pripadajućim atributima: tools i android. XML ima široko područje uporabe, kao što su web servisi, konfiguracijske datoteke, razmjene podataka, razvoj web-stranica, razmjena podataka između baza podataka.

Slika 4.1. *XML dokument*

4.4. Firebase

Firebase je razvojna platforma Google usluge koja pruža razne alate i mogućnosti koje pomažu i povećavaju kvalitetu izrade mobilnih i web aplikacija. Firebase svoje usluge nudi u oblaku, čime olakšava i ubrzava razvoj projekata. Firebase je dio ovog rada zato što omogućava jednostavnu komunikaciju između projekta, tj. Android Studija i baze podataka, za čije stvaranje je Firebase i zadužen. Podržava puno korisnih značajki, kao što su Baza podataka u stvarnom vremenu (eng. *Realtime database*), Skladište (eng. *Storage*), Funkcije u oblaku (engl. *Cloud Functions*), Strojno učenje (engl. *Machine Learning*), Udaljena konfiguracija (engl. *Remote Config*) i Ekstenzije (engl. *Extensions*). Značajke koje su upotrijebljene u ovom radu su Autorizacija (engl. *Authentication*) i Baza podataka u oblaku (engl. *Cloud Firestore*).

4.4.1. Authentication

Authentication značajka pruža jednostavno rješenje za upravljanje korisničkim računima. Omogućava različite načine prijave i registracije korisnika, a korisnici koji budu koristili aplikaciju izrađenu u sklopu ovog rada prijavljivat će se pomoću odabrane email adrese i lozinke. Firebase *Authentication* usluga omogućava sigurnu prijavu u aplikaciju i povezivanje sa vlastitim izrađenim računom. Ostali podržani načini prijave su Google *Sign-In,* Facebook *Login,* prijava putem broja telefona ili anonimna prijava.

Firebase *Authentication* podrazumijeva napredne sigurnosne postavke, čime se sprječava svaka neželjena prijava ili pokušaj korištenja tuđih korisničkih podataka u svrhu neželjenih radnji. Prilikom korištenja ove usluge, autor aplikacije ima potpuni nadzor nad korisnicima i kontrolira njihov pristup podacima na temelju autentikacije. Oni imaju omogućen i uvid u praćenje i učestalost korištenja aplikacije od strane svakog korisnika. Jednako kao i većina drugih Firebase usluga, *Authentication* značajka integrirana je s ostalim sličnim značajkama.

4.4.2. Cloud Firestore

Firestore predstavlja NoSQL bazu podataka koja u stvarnom vremenu omogućava komunikaciju između korisnika i podataka pohranjenih u bazi. Općenito, *NoSQL* baze podataka, prema [19], su baze koje se također nazivaju i nerelacijske iz razloga što ne koriste tablice, već nestrukturirane podatke kao što su dokumenti ili JSON (engl. *javaScript Object Notation*, oblik podatka za razmjenu podataka). *Firestore* omogućava vrlo brzu i skalabilnu pohranu podataka u oblaku i sinkronizaciju informacija za klijentsku i poslužiteljsku stranu razvoja projekta. Omogućava način rada van mreže, što znači da će izrađene responzivne aplikacije moći raditi neovisno o povezanosti na internetsku mrežu. Korisnik može čitati iz baze, pisati u nju ili postavljati upite (engl. *query*) i kada uređaj nije spojen na Internet, a u trenutku kada uređaj ponovno poprimi status online, Cloud *Firestore* sinkronizirat će sve lokalne promjene s podacima u oblaku.

Integrirana je s drugim *Firebase* i Google *Cloud* proizvodima, zbog čega se upravo ona vrlo često odabire za skladištenje podataka. Fleksibilnost ove aplikacije ogleda se u tome što omogućava spremanje podataka u hijerarhijskom obliku, krenuvši od kolekcije koja je sastavljena od dokumenata, a dokumenti od jednostavnih podataka ili kompleksnih ugniježđenih objekata. Ova struktura prikazana je na slici 4.2. Postavljanjem upita dohvaćaju se individualni, posebni dokumenti, dijelovi dokumenata ili svi dokumenti koji odgovaraju upitu.

Slika 4.2. *Hijerarhijski prikaz Firestore baze podataka*

5. IMPLEMENTACIJA PROGRAMSKOG RJEŠENJA

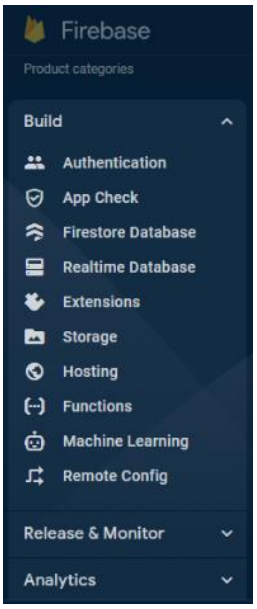
Na osnovu specifikacije zahtjeva na programsko rješenje i odabira tehnologija kojima je moguće kvalitetno realizirati navedene zahtjeve potrebno je implementirati navedeno u konkretnu aplikaciju. Ono što je u prvom koraku implementacije potrebno napraviti jest povezati se s bazom, izmijeniti datoteke *AndroidManifest.xml* i *Gradle*, čime je razvojno okruženje spremno za izradu aplikacije. U sljedećim poglavljima bit će detaljno objašnjeni spomenuti preduvjeti, prikazano korisničko sučelje aplikacije (eng. *frontend*) te pozadinski dio aplikacije (engl. *backend*).

5.1. Preduvjeti za izradu aplikacije

Nakon odabira Firebase usluge potrebno je povezati aplikaciju, tj. projekt s Firebaseom na Internetu. Prvo je potrebno stvoriti projekt na Firebaseu. Na njihovim službenim stranicama [20], klikom na gumb *Go to Console* otvara se sučelje za stvaranje projekta. Nakon popunjavanja dijaloga informacijama vezano za projekt, otvara se početna Firebase stranica, gdje pritiskom na određenu značajku (slika 5.1.), otvaraju se različiti prozori. Odabirom *Authentication* usluge, nude se mogući načini prijave i registracije korisnika, od kojih je u radu korišten Email/Password način autentikacije. Klikom miša na *Firestore Database*, prikazuje se sučelje za izradu baze podataka na kojem se izabire način (eng. *mod*) pokretanja baze, prilikom čega se mora obratiti pozornost na pravila vezana za ponuđene načine. Odabrani je način za ovaj projekt *test mod*, kojemu su promijenjena pravila kako *Firestore* baza podataka ne bi istekla nakon 30 dana.

Nakon postavljanja projekta na Internetu, neophodno je povezati Android Studio sa stvorenim projektom. Postupak povezivanja podrazumijeva dodavanje paketa za razvoj programa (eng. *SDK – Software Development Kit*) unutar aplikacijske datoteke *build.gradle*. Navedena datoteka služi za definiranje postavki za Android projekte. Nakon izmjena u navedenoj datoteci, provodi se sinkronizacija s ostatkom projekta. Ovim korakom završeno je povezivanje Android Studija s *Firebase* projektom.

AndroidManifest.xml je, prema [21], datoteka koja sadrži obavezne deklaracije informacija o aplikaciji i njenim komponentama pomoću kojih pruža sustavu Android informacije o aplikaciji. Navedene informacije Android Studio postavlja automatski prilikom izrade aplikacije. Osim navedenog, tu su popisana i dopuštenja koja aplikacija zahtjeva kako bi mogla pristupiti zaštićenim dijelovima sustava ili drugim aplikacijama. Izmjena ovog dokumenta se odnosi na ručno unošenje dopuštenja za pristup Internetu, kako bi aplikacija mogla komunicirati s Firebase bazom podataka, ali i ostalim podacima prisutnim na Internetu.



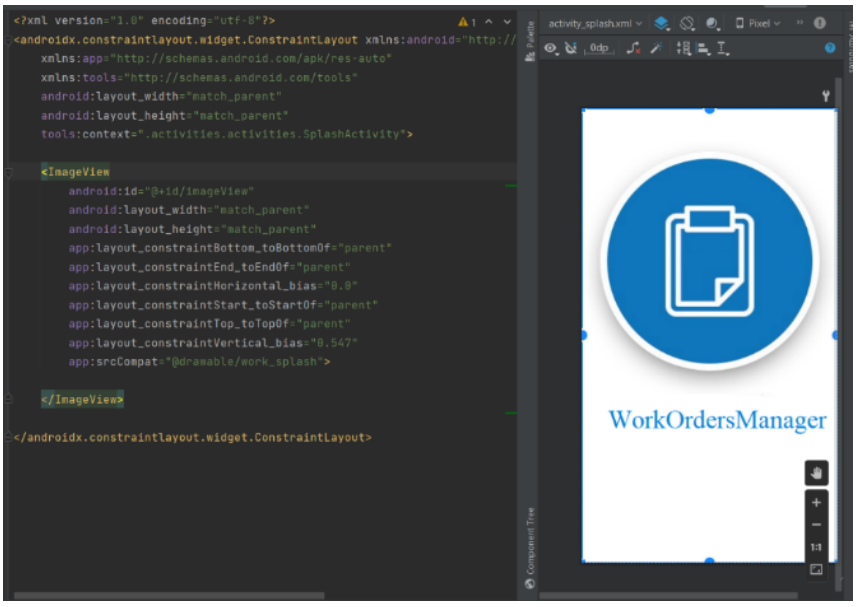
Slika 5.1. *Značajke Firebase usluge*

5.2. Korisničko sučelje

Svaka aplikacija, neovisno o tome pokreće li se na Androidu, iOS-u, webu ili nečem drugom, dijeli se na dva velika dijela. Prvi dio je izgled aplikacije, točnije korisničko sučelje, dio koji je vidljiv korisniku aplikacije i koji predstavlja sponu između korisnika i ostatka aplikacije. Taj ostatak aplikacije je sloj skriven od korisnika, logika cijelog projekta koja se odvija u pozadini aplikacije. Aplikacija izrađena u sklopu ovog rada, pod nazivom *WorkOrdersManager,* koristi četiri aktivnosti i više fragmenata, koji predstavljaju zaslone aplikacije. Više o ovoj temi pisano je u potpoglavlju 5.3. koje se bavi programskim dijelom aplikacije, tj. *backendom*.

Za vizualni dio aplikacije korišten je opisni jezik XML. Korišteni su njegovi razni elementi, kao što su *textView* za prikaz teksta, *editText* za unos teksta, *imageView* za prikaz fotografija, *button* za okidanje događaja, *recyclerView* za učinkovit prikaz velikih skupina podataka i mnogi drugi.

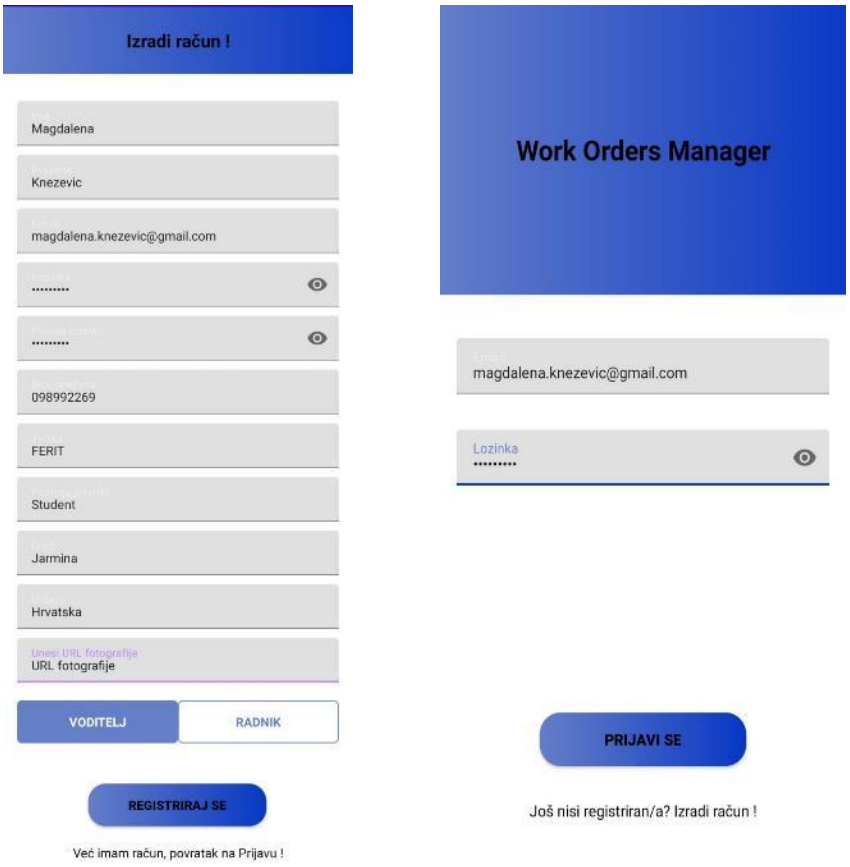
Slika 5.2. prikazuje jedan dio koda napisan u XML-u, kao i njegov rezultat s desne strane.



Slika 5.2. *XML datoteka*

Pokretanjem aplikacije, korisniku se prvo prikaže *Splash Screen* (slika 5.2.), početni zaslon koji sadrži logo aplikacije *WorkOrdersManager*. Zatim se otvara prozor za prijavu korisnika, ili, ukoliko nisu registrirani, sučelje koje omogućava izradu novog računa. Prilikom registracije, korisnik unosi svoje ime, prezime, službeni mail, lozinku, broj mobitela, tvrtku u kojoj radi, radno mjesto u tvrtki, grad i državu iz koje dolazi. Također je omogućen unos URL poveznice fotografije koju korisnici žele postaviti za sliku profila. Poveznica se u kodu pomoću biblioteke *Glide*, prema [22], pretvara u fotografiju. Korisnik prilikom prve prijave u aplikaciju odabire ima li status Voditelja ili Radnika. Ovo je bitno jer se obzirom na to otvara odgovarajuće sučelje, čime se odgovara na jedan od zahtjeva na programsko rješenje. Prijava se vrši email adresom i izabranom lozinkom.

Na slici 5.3. prikazani su sučelje za prijavu i sučelje za registraciju.

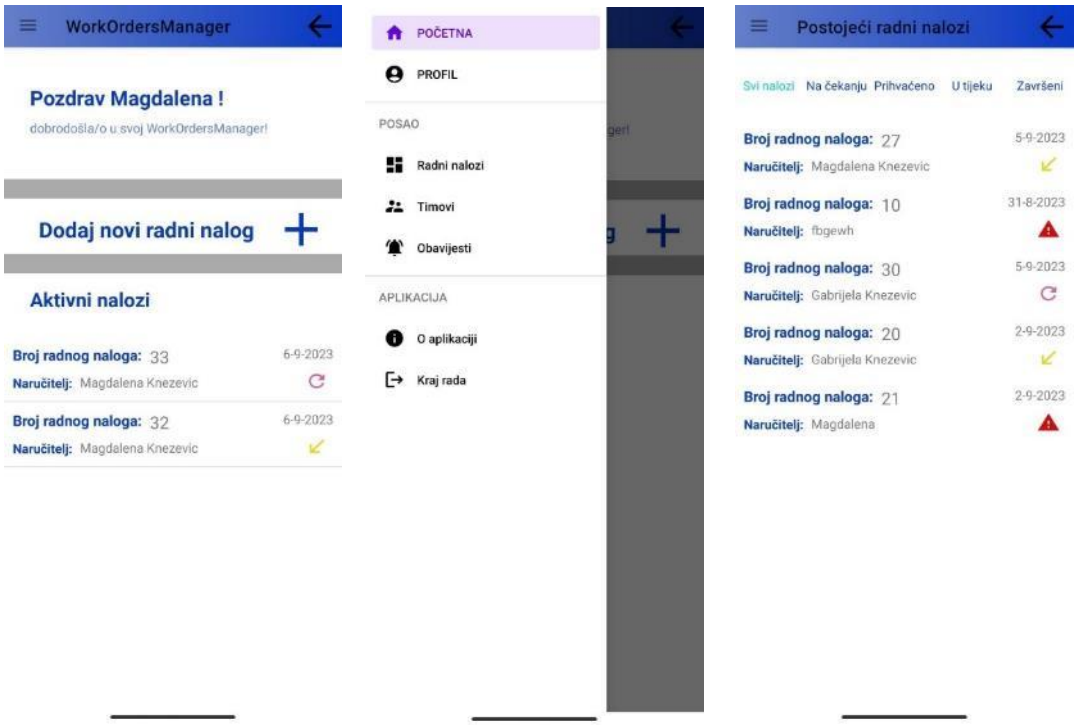


Slika 5.3. *Registracija i prijava korisnika*

Ukoliko je prijavljeni korisnik voditelj, njegova početna stranica izgleda kao što je prikazano na slici 5.4.a. Tu se nalazi opcija dodavanja novog radnog naloga te su unutar komponente *recyclerView* prikazani trenutno aktivni nalozi. Mogući statusi radnog naloga mogu se vidjeti u odjeljku *Radni nalozi* u navigacijskom okviru aplikacije. Statusi su vidljivi i na temelju zastavice koja je prikazana unutar *recyclerViewa*, a drukčija je za svaki pojedini status. Navigacijsko okno ručno je implementirano kako bi se olakšalo snalaženje unutar aplikacije, a njegov izgled prikazan je na slici 5.4.b. U odjeljku *Radni nalozi*, klikom na svaki od ponuđenih statusa, prikažu se odgovarajući radni nalozi sa željenim statusom (slika 5.4.c.). Status označava u kojem dijelu radnog procesa se nalog nalazi, a statusi su: Prihvaćeno, U tijeku i Završeno.

Prilikom izrade radnog naloga, voditelj prvotno navodi broj radnog naloga i datum izrade radnog naloga, prilikom čega mu se otvara sučelje kalendara za odabir trenutnog dana u mjesecu. Zatim unosi ime naručitelja, odabire vrstu radnog naloga (u aplikaciji, sve moguće vrste prikazane su unutar padajućeg izbornika), opis posla i napomenu za radnike, tj. izvršitelje posla. Izgled ovakvog radnog naloga prikazan je na slici 5.5.a. Nakon što je to popunjeno, pritiskom na tipku

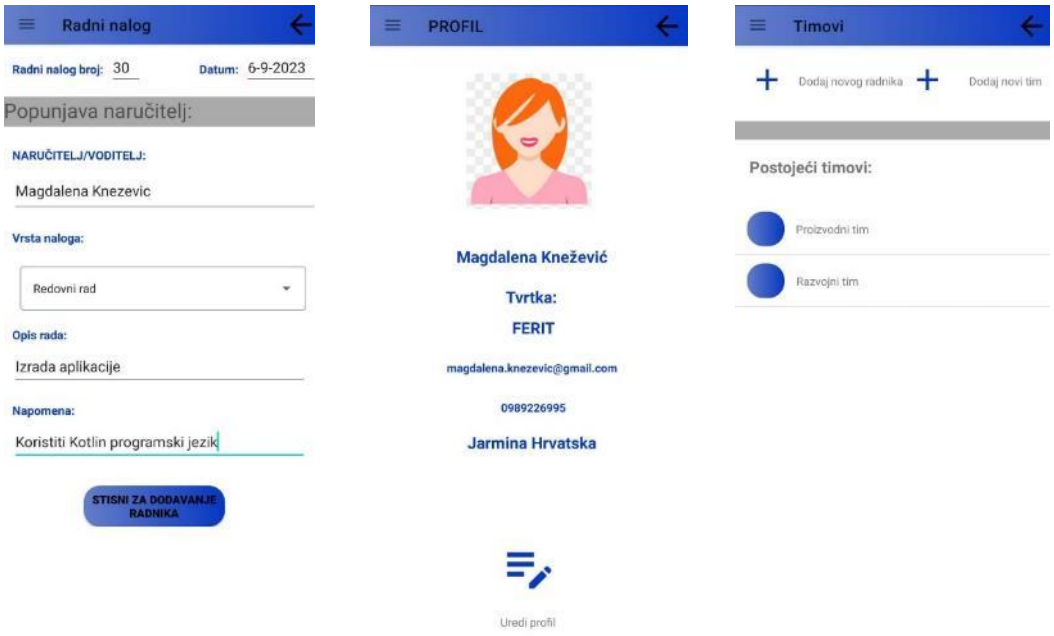
Stisni za dodavanje radnika otvara se novi fragment, koji nudi korisniku pretragu svih dostupnih radnika u bazi po email adresi. Nakon odabira svih radnika, klikom na *Pošalji,* radni nalog se pohranjuje u bazu podataka te se prikazuje u ranije spomenutom odjeljku, zajedno sa ostalim radnim nalozima koji se nalaze u vlasništvu trenutnog korisnika.



a)*Početni zaslon voditelja* **b)** *Navigacijski okvir* **c)***Filtriranje radnih naloga*

Slika 5.4. *WorkOrdersManagero*

Unutar navigacijskog okvira, nalazi se i opcija kojom se pregledava profil trenutno prijavljenog korisnika, što se može vidjeti na slici 5.5.b. Ukoliko korisnik želi promijeniti neke informacije vezane uz svoj račun, može to učiniti pritiskom na element *imageView,* ispod kojeg piše *Uredi profil* (vidljivo na slici 5.5.b.). Odjeljak *Timovi*, prikazan na slici 5.5.c., omogućuje voditelju dodavanje novog radnika u bazu podataka i stvaranje novog tima.

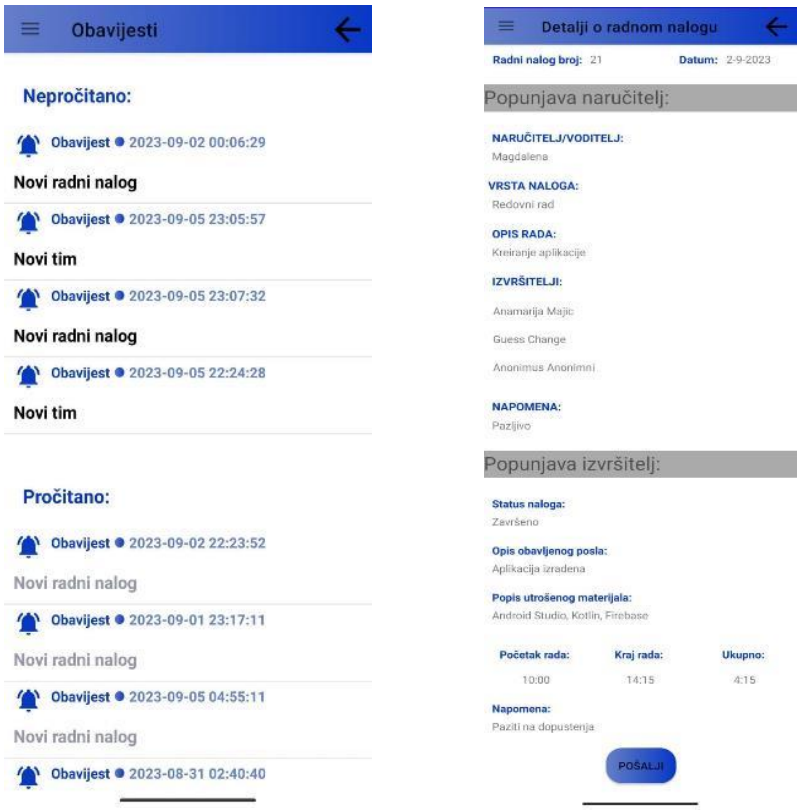


a) *Izrada radnog naloga* **b)** *Profil* **c)** *Timovi – voditelj*

Slika 5.5. *WorkOrdersmanager*

Odabirom *Obavijesti*, prikazuje se sučelje na slici 5.6.a**,** gdje su dobivene obavijesti podijeljene u Pročitane i Nepročitane. Fragment *O aplikaciji* nudi informativne podatke o aplikaciji, kao i riječ programera i zahvalu svim korisnicima aplikacije. Na kraju rada, korisniku se otvara skočni prozor koji traži potvrdu za odjavom iz aplikacije.

Ukoliko je korisnik prijavljen kao radnik, tada njegova početna stranica izgleda vrlo slično voditeljevoj, samo što radnik nema mogućnost izrade radnog naloga. Još jedna razlika između ove dvije uloge u aplikaciji jest ta što radnik ima mogućnost uređivanja zaprimljenog radnog naloga. Jedan konačan izgled radnog naloga, popunjenog od strane voditelja i radnika, prikazan je na slici 5.6.b. Pod naslovom *Timovi*, radnik ima samo uvid u svoje timove, tj. timove kojima je on dodijeljen, ali ne i stvaranje istih.



a) *Obavijesti* **b)** *Ispunjen radni nalog*

Slika 5.6. *WorkOrdersManager*

Izgled ostalih fragmenata dostupan je u Prilozima (P.5.1).

5.3. Programski dio aplikacije

U ovom dijelu bit će opisano sve ono što se odvija u pozadini aplikacije. Sva sučelja iz potpoglavlja 5.2. i svaka akcija podržana na bilo kojoj aktivnosti ili fragmentu omogućena je zahvaljujući upravo programskom dijelu aplikacije.

Stvaranjem novog projekta, stvara se i *Main Activity.kt*, datoteka unutar koje se piše programski kod u programskom jeziku Kotlin. Zajedno s ovom datotekom, otvara se i njezina pripadna datoteka *activity_main.xml*, sa XML naredbama, u kojoj je definirana struktura *Main Activity.kt* zaslona. Aktivnost (eng. *activity*) je klasa koja predstavlja jedan zaslon aplikacije te ima svoj vlastiti životni ciklus. Pokretanjem aplikacije, Android Studio automatski postavlja ovu aktivnost kao početnu. Ukoliko je potrebno, to se može promijeniti izmjenom već spomenute datoteke *AndroidManifest.xml*, što je upravo i učinjeno u ovom radu. U sklopu ovog projekta, izrađene su samo četiri aktivnosti, ali mnogo više fragmenata. Fragmenti su komponente Androida koje također predstavljaju korisničko sučelje, ali su oni dio jedne aktivnosti. Razlozi češćeg korištenja fragmenata su njihova brža izmjena prilikom prebacivanja s jednog na drugi fragment,

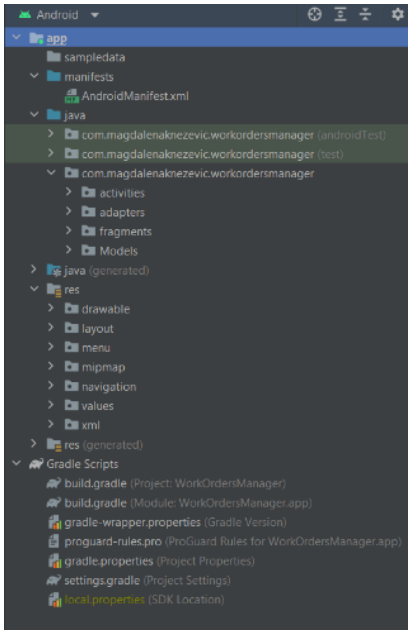
jednostavnije korištenje i mogućnost prikaza više fragmenata unutar iste aktivnosti, prema [23]. Način na koji se prelazi s jednog fragmenta na drugi, prikazan je u kodu 5.1.

```
val fragmentTransaction: FragmentTransaction? =
    supportFragmentManager?.beginTransaction()
fragmentTransaction?.replace(
    R.id.fragment_container,
    fragment
)
fragmentTransaction?.commit()
```

Kod 5.1. *Izmjena fragmenata*

Osim navedenog, na razini projekta postoje još dvije vrste datoteka, a to su Adapteri i Modeli. Adapter klasa neophodna je za postojanje *recyclerView* komponenti, jer ona povezuje klasu *ViewHolder* koja predstavlja jednu stavku u sklopu *recyclerViewa*, te podatke koji će biti prikazani unutar *recyclerView* značajke. Više o *recyclerView* i adapter temi bit će opisano kasnije u ovom poglavlju. Model klasa predstavlja tzv. *data class,* prema [24], jednostavnu klasu koja sadrži podatke i standardne funkcionalnosti. Nazivi modela u ovom projektu ujedno su i nazivi kolekcija u *Firestore* bazi podataka. Zahvaljujući modelima, moguće je držati informacije o svim vrstama korisnika, radnih naloga, timova i obavijesti u bazi podataka.

Na slici 5.7. prikazana je struktura aplikacije izrađene u sklopu završnog rada.



Slika 5.7*. Struktura projekta*

Korištenje aplikacije započinje registracijom korisnika. Kao što je već spomenuto, u sklopu Firebase *Authentication* značajke, odabrani način za izradu računa je korištenje emaila i lozinke. *SignUp Activity* brine o stvaranju korisnika i njegovoj pohrani u bazi podataka, uz pomoć funkcije *createUserWithEmailAndPassword(),* kojoj se kao parametri šalju email i lozinka. Prilikom komunikacije s bazom, uvijek se postavljaju dva *listenera,* od kojih jedan označava uspješnu operaciju, a drugi obavještava o pogrešci koja se može i ne mora dogoditi tijekom procesa. Prema tome, ukoliko je ova komunikacija uspješna, stvara se model *Director (*hrv. *Voditelj)* ili *Worker (*hrv. *Radnik)*, odgovarajuća kolekcija u *Firestore* bazi podataka te se stvoreni model sprema kao dokument kolekcije. Za naziv svakog dokumenta postavljen je identifikacijski broj (u daljnjem tekstu: id) svakog korisnika. Ukoliko dođe do pogreške, u *Logcatu* Android Studija ispisuje se poruka o pogrešci. *SignIn Activity* se zatim spaja na bazu i pristupa potrebnoj kolekciji. Postavljanjem upita (eng. *query*) iteriraju se svi dokumenti te se iteracija završava na onom koji odgovara upitu i on se vraća kao odgovor metode *get().*

Na temelju dohvaćenog dokumenta, točnije korisnika, otvara se početni zaslon aplikacije te se može krenuti s radom. Na fragmentu *Profil* eksplicitno su prikazani detalji o korisniku upravo zahvaljujući radnji opisanoj u prethodnim rečenicama. Unutar *Profila*, nalazi se mogućnost ažuriranja vlastitog računa. Izmjena podataka o korisniku realizirano je korištenjem *Firestoreove* funkcije *runTransaction*, kojoj se prilikom ažuriranja predaje polje koje se ažurira te nova vrijednost kojom će stara biti zamijenjena. Razlog korištenja ovog načina ažuriranja je taj što osigurava operacije u stvarnom vremenu i radi sa dosljednim, provjerenim informacijama, prema [25].

Sljedeći korak je izrada radnog naloga. Kako bi se osiguralo da radni nalog ima sve osobine koje treba, napravljen je model pod nazivom *WorkOrder (*hrv. *radni nalog)*, koji se sastoji od *id*-a naloga, broja radnog naloga, datuma, izvršitelja, naručitelja, opisa zadanog i odrađenog posla, vremena potrebnog za obavljanje posla i bilješki. Prvo se izrađuje radni nalog kojeg popunjava voditelj. Na tom fragmentu prikazani su samo elementi koje popunjava navedeni tip korisnika. Nakon što voditelj završi sa postavljanjem radnog naloga, nalog se sprema u bazu i šalje radnicima koji su navedeni kao izvršitelji. Za prikaz radnika unutar radnog naloga korišten je *recyclerView*, iz razloga što se za jedan radni nalog može odabrati više izvršitelja. Kako bi se omogućio prikaz radnika na fragmentu radnog naloga, stvoren je još jedan model, pod nazivom *orderContainsWorker*, koji sadrži dvije vrijednosti, a to su *id* radnog naloga i *id* radnika. Pomoć ovog modela dohvaćanje svih radnika jednog naloga uvelike je olakšano. Opisani prikaz radnika unutar *recyclerViewa* prikazan je u kodu 5.2.

```
db.collection("orderContainsWorker")
    .whereEqualTo("orderId", orderID)
    .get()
    .addOnSuccessListener { result ->
        val workers = ArrayList<Worker>()
        for (data in result.documents) {
            val contain = data.toObject(OrderContainsWorker::class.java)
            db.collection("workers")
                .whereEqualTo("id", contain!!.workerId)
                .get()
                .addOnSuccessListener { res ->
                    for (data1 in res.documents) {
                        val worker = data1.toObject(Worker::class.java)
                        workers.add(worker!!)
                    }

                    recyclerAdapter =
                        WorkersInTeamAdapter(workers)
                    workersInOrder.apply {
                        layoutManager = LinearLayoutManager(context)
                        adapter = recyclerAdapter
                    }
                }
        }
    }
    .addOnFailureListener {
        Log.w("WorkOrderFragment", "Greška prilikom dohvaćanja dokumenta.",
it)
    }
```

Kod 5.2. *Dohvaćanje svih radnika jednog radnog naloga*

Na isti način se prilikom stvaranja novog tima odabiru radnici i prikazuju na odgovarajućem fragmentu.

Nakon primitka radnog naloga, radnik može uređivati dijelove naloga koji su namijenjeni za njega. Unosi opis obavljenog posla, popis utrošenog materijala, vlastitu bilješku te označava početak i kraj rada, nakon čega aplikacija pomoću korisnički izrađene funkcije *calculateTimeDifference()* izračunava ukupno vrijeme uloženo u rad. Konačan rezultat ispisuje se na ispravnu komponentu *textView*. Nakon popunjavanja svih polja, pritiskom na odgovarajuću tipku, upisani podaci se spremaju u bazu, u istu kolekciju i isti dokument kao što je radni nalog kojeg je radnik zaprimio, samo sa zapisanim dodatnim informacijama. Ukoliko je status radnog naloga jednak stringu *'Završeno'*, tada je radniku omogućen prijelaz na Konačan radni nalog.

Prilikom prijelaza na drugi fragment, koristi se klasa *Bundle*() koja za zadaću ima prijenos podataka iz jednog fragmenta ili aktivnosti u drugi. Tako se ovdje *Bundle* koristi za slanje svih informacija o radnom nalogu na fragment *Konačni radni nalog*. On prikazuje sve podatke o

nalogu, one koje je unio voditelj i one koje je unio radnik. Ovaj fragment ne može se uređivati, nego služi samo kao pregled obavljenog posla. Pritiskom na *Pošalji*, radni nalog je završen, te se svrstava pod kategoriju Završenih radnih naloga. U kodu aplikacije napisana su ograničenja, tj. ranije navedeni uvjeti koji se moraju ispuniti prije prelaska na *Konačan nalog*. Prvi uvjet je status postavljen na *Završeno*, a drugi uvjet da se tako popunjen nalog mora prvo spremiti, pa tek onda kliknuti na *Konačan nalog*. Opisani algoritam prelaska na drugi fragment, prikazan je u pseudokodu 5.3.

```
Kada korisnik pritisne gumb saveOrderBttn {
   dohvati trenutno izabrani status iz radioGrupe i sačuvaj ga u varijablu "status"
   Pretvori "status" u string i pohrani ga u varijablu "statusInString"

   Ako je "statusInString" jednako "Prihvaćeno" ili "U tijeku" {
      Prikazi poruku korisniku: "Za Konačan nalog, stanje mora biti 'Završeno'."
   } inače, ako promjene nisu sačuvane {
      Prikazi poruku korisniku: "Za Konačan nalog, morate prvo Spremiti promjene."
   } inače {
      - ID radnog naloga: workOrderID
      - opis obavljenog posla: descWorker
      - Napomena radnika: noteWorker
      - Materijal: materialWorker
      - Početak rada: start
      - Kraj rada: end
      - Ukupno sati: allHours
      - Status: statusInString

      Izradi paket "bundleToo" i postavi vrijednosti ovih podataka u njega

      Postavi argumente za fragment "workOrderFinalFragment" na vrijednosti iz "bundleToo"

      Zamjeni trenutni fragment sa fragmentom "workOrderFinal"
   }
}
```

Pseudokod 5.3. *Uvjeti za prelazak na drugi fragment*

Voditelj ima i mogućnost dodavanja novog radnika u bazu. Pozadinska logika ove značajke je korištenje metode *set()* kojoj se predaje stvoreni radnik, a u *Firestore* bazi se dodaje novi dokument koji predstavlja novo dodanog korisnika. Njegovim elementima može se pristupati jednako kao i korisnicima stvorenim prilikom registracije u aplikaciju. Ukoliko je voditelj koristio ovu mogućnost, on zadaje radnikovu email adresu i lozinku, a unutar koda automatski je postavljen

status tako dodanog korisnika kao radnik. Novi radnik se zatim prijavljuje postojećim korisničkim podacima, no na fragmentu *Profil* neće pronaći sve svoje informacije, zbog čega može koristiti opciju ažuriranja profila.

Implementacija obavijesti odnosi se na stvaranje još jednog modela, zvanog *Notification*, koji drži podatke koji su potrebni za stvaranje obavijesti. Ti podaci su *id* obavijesti, *id* korisnika kojem se šalje obavijest, vrijeme izrade, naslov, sadržaj te boolean varijabla isRead, pomoću koje će se obavijesti sortirati u pročitani ili nepročitani odjeljak. Obavijest se šalje radniku kada ga se doda u novi radni nalog ili novi tim, te voditelju svaki put kada radnik promijeni status radnog naloga ili pošalje završeni radni nalog.

Svaki put kada je bilo potrebno izlistati podatke iz određenog skupa, kao što je bio slučaj u početnom fragmentu te fragmentima *Radni nalozi*, *Timovi* i *Obavijesti*, korištena je komponenta pod nazivom *recyclerView*. Ona služi za prikazivanje dinamičkih listi podataka, a vrlo je efikasna jer recikliranjem elemenata sudjeluje u smanjenju opterećenja memorije. Ovisno o željenom rasporedu, koriste se različite vrste *LayoutManagera*, elementa koji se brine o vizualnom načinu prikaza stavki unutar *recyclerviewa*. Kako bi ova komponenta postala upotrebljiva, osim definiranja vrste *LayoutManagera*, potrebno je stvoriti klasu Adapter, spomenutu ranije u tekstu te klasu *ViewHolder*. Adapter služi za spajanje prikaza podataka na korisničko sučelje i podataka iz baze koje je potrebno prikazati. Prema [26], unutar sebe mora imati tri metode, a to su:

- *getItemCount* – izračunava ukupan broj stavki prikazanih u *recyclerViewu,*
- *onCreateViewHolder* – metoda zadužena za recikliranje klase *ViewHolder,*
- *onBindViewHolder* – brine o postavkama različitih podataka i metoda povezanih sa pritiskom na određeni element *recyclerViewa, prema* [27].

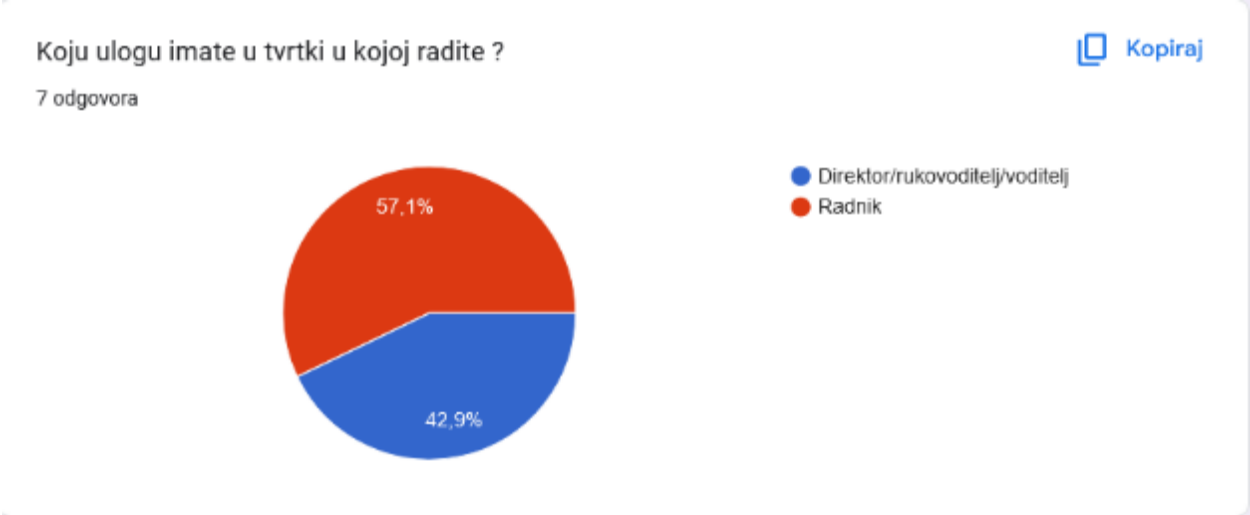
Klasa *ViewHolder* koristi se za dohvaćanje elemenata iz jedne stavke *recyclerViewa* te se na taj način koristi za optimizaciju koda, tj. sprječava se stvaranje nepotrebnih držača za stavke *recyclerViewa* koje nisu trenutno prikazane na zaslonu. Adapter klasa implementira i sučelje nazvano *ContentListener*, koje sadrži funkciju *onItemButtonClick*, a koja će u daljnjem kodu služiti za provođenje logike nakon pritiska na pojedinu stavku u *recyclerViewu*.

Ako je rad sa aplikacijom završen, korisnik odlazi na fragment *Odjava* gdje mu se pojavljuje dijalog koji ga pita da potvrdi svoju želju za izlaskom iz aplikacije. Dijalog predstavlja mali skočni prozor koji ispituje korisnika je li siguran u svoju odluku. Dijalog korišten u ovom radu je *AlertDialog*, prema [28]. Ukoliko korisnik odgovori potvrdno, program ga odjavljuje iz aplikacije te je njegov rad gotov.

6. KORISNIČKO ISKUSTVO

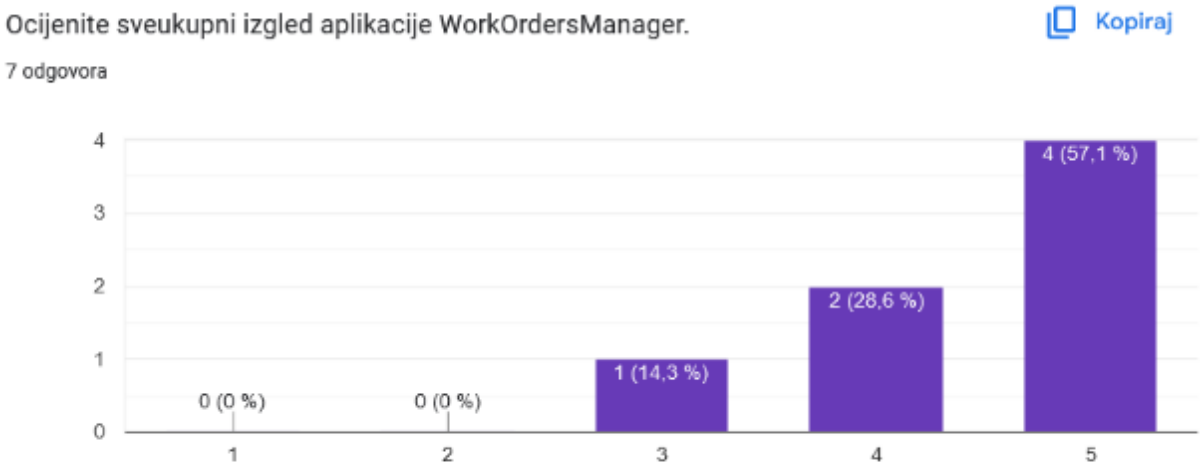
Nakon izrade aplikacije, korisno je dobiti povratne informacije krajnjih korisnika o iskustvu korištenja aplikacije i razini zadovoljstva nakon upoznavanja s istom. Povratni komentari sastavni su dio svakog projekta te pružaju autoru aplikacije uvid u objektivnu kvalitetu izrađene aplikacije. Konstruktivne kritike poželjne su kako bi se ukazalo na nedostatke aplikacije, te u budućnosti poradilo na tome, s ciljem podizanja kvalitete aplikacije na veću razinu.

Ispitivanje korisnika u ovom radu provedeno je na skupu djelatnika već spomenute firme *Vinke plus d.o.o.*. Aplikacija je podijeljena voditeljima i radnicima te im je nakon toga ponuđena upitnik u obliku Google Forms ankete, dostupan u Prilozima (P.6.1.) na kraju rada. Među ispitanicima nalazili su se radnici u proizvodnji, u skladištu, u pakirnici, na strojevima, ali i voditelji održavanja i proizvodnje te rukovoditelj kontrole kvalitete. Ispitanici su bili zamoljeni da ispune kratku anketu, koja im je poslana putem poveznice na njihove email adrese. Na slici 6.1. prikazano je koliko ispitanika spada u koju grupu zaposlenika. Troje ispitanika su voditelji/rukovoditelji, dok su ostatak radnici.



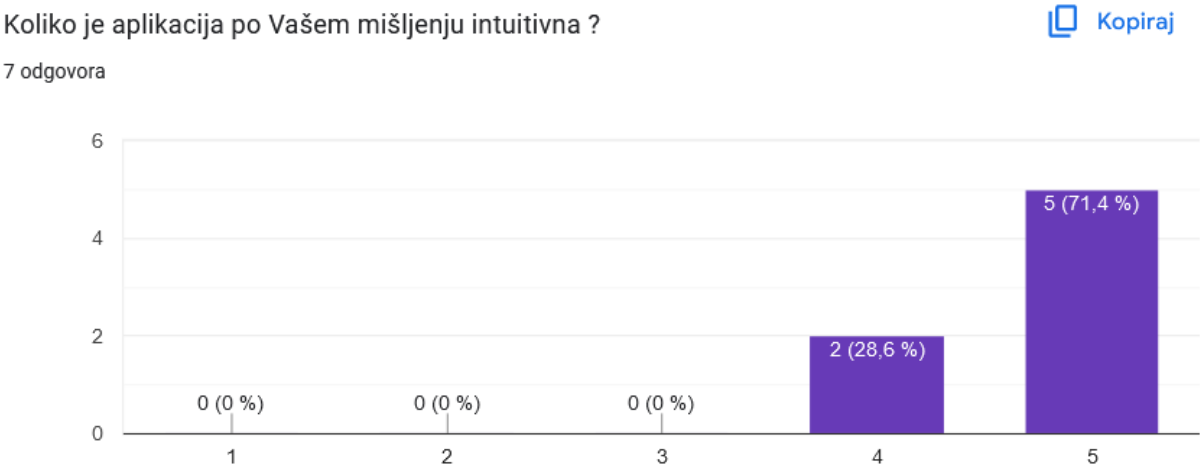
Slika 6.1. *Uloga u tvrtki*

Slike 6.2. - 6.5. ilustriraju raspone odgovora na pitanja na koja su morali odgovoriti ocjenom. Na slici 6.2. ispitanici su odgovarali na pitanje o izgledu aplikacije. Odgovori su raspodijeljeni na ocjene tri (3), četiri (4) i pet (5).



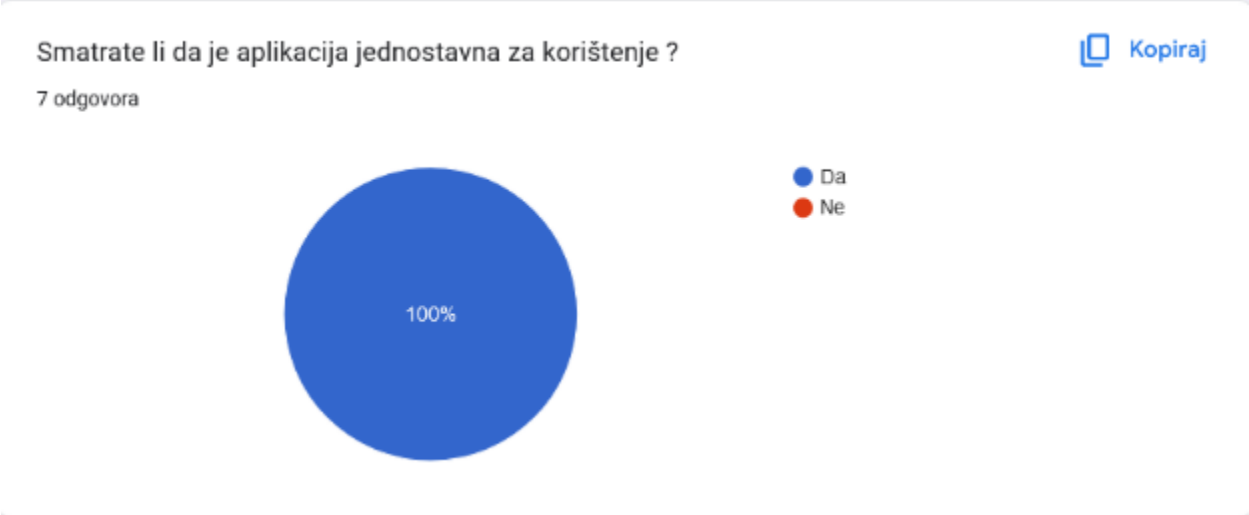
Slika 6.2. *Izgled aplikacije*

Sljedeće pitanje odnosilo se na intuitivnost aplikacije. Točan postotak odabranih ocjena prikazan je na slici 6.3.



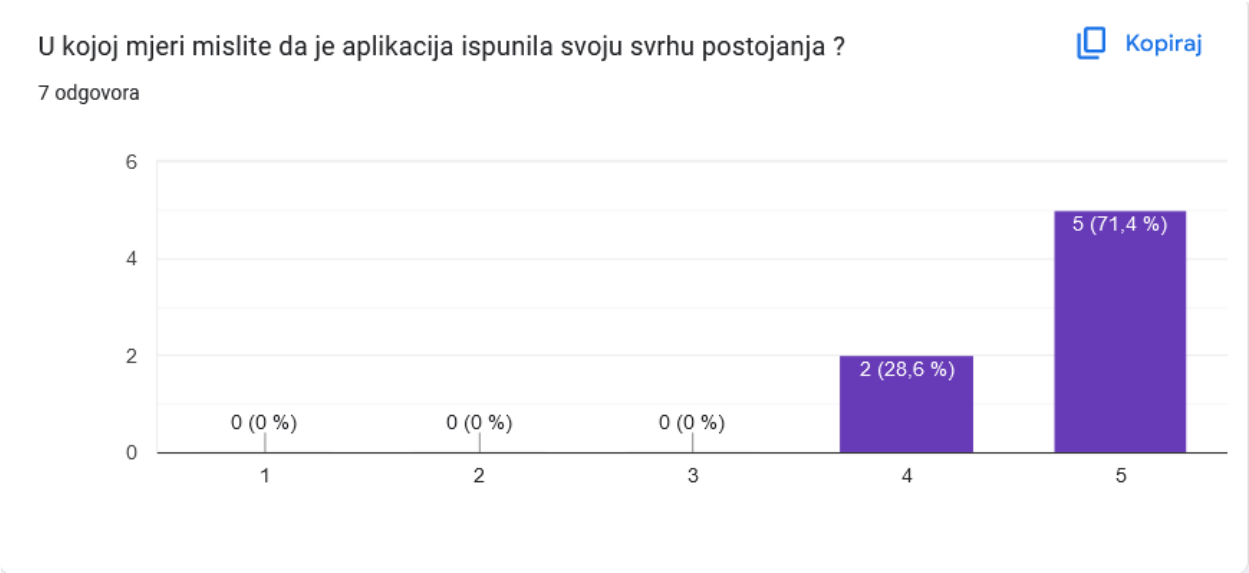
Slika 6.3. *Intuitivnost aplikacij*

Na pitanje smatraju li da je aplikacija jednostavna za korištenje, svi ispitanici su odgovorili potvrdno. Na slici 6.4. prikazan je grafikon sa rezultatima odgovora na ovo pitanje.



Slika 6.4. *Jednostavnost korištenja aplikacije*

Zadnje pitanje na koje su ispitanici odgovarali ocjenom odnosilo se na ispunjenje svrhe postojanja aplikacije. Većina ispitanika odgovorila je najvećom mogućom ocjenom (pet, 5), dok su se ostali odlučili za ocjenu četiri (4).



Slika 6.5. *Svrha aplikacije*

Na pitanje što bi promijenili u ovoj aplikaciji, dva korisnika nisu napisala konkretne odgovore, a ostatak ih je odgovorio sljedeće:

- Korisnik 1 : Dodati opciju ispisa radnog naloga.
- Korisnik 2 : Centriranje teksta.

- Korisnik 3 : Boja slova.
- Korisnik 4 : Povećanje fonta.
- Korisnik 5 : Pretraživanje radnika po imenu i prezimenu, a ne po emailu.

Prolazeći kroz ove odgovore, zaključuje se da većina korisnika ima prigovore o izgledu slova i poziciji teksta.

Posljednje pitanje ove ankete glasilo je: Što biste pohvalili u ovoj aplikaciji ? Odgovori korisnika izgledaju ovako:

- Korisnik 1 : Preglednost aplikacije.
- Korisnik 2 : Obuhvaća sve što je potrebno.
- Korisnik 3 : Mogu brzo vidjeti sve svoje zadatke.
- Korisnik 4 : Mogućnost kontrole statusa radnog naloga u svakom trenutku.
- Korisnik 5 : Praktična je i uvijek imam u džepu sve svoje radne naloge.
- Korisnik 6 : Ušteda papira.
- Korisnik 7 : Ikone koje označavaju status naloga.

Na temelju svih odgovora iz provedene ankete, dolazi se do zaključka da je većina ispitanika zadovoljna iskustvom korištenja aplikacije WorkOrdersManager, kao i početnom idejom digitalizacije radnih naloga. Ispitanici su saželi svoj ukupan dojam korištenja aplikacije, na način da su iznijeli individualne prigovore na aplikaciju, ali i ponudili prijedloge za unaprjeđenje iste.

7. ZAKLJUČAK

Zadatak završnog rada bila je izrada Android aplikacije koja će omogućiti upravljanje radnim nalozima u digitalnom obliku. Navedeno uključuje izradu Radnih naloga, dodjela radnih naloga kvalificiranim radnicima te praćenje napretka svakog pojedinog Radnog naloga. Na temelju opisa životnog ciklusa dokumenta Radni nalog u više različitih tvrtki, napravljeni su zahtjevi na programsko rješenje koji su služili kao uputa za izradu aplikacije. Osim zahtjeva, na tijek izrade programskog rješenja utjecala su i iskustva postojećih programskih rješenja za istu ili sličnu svrhu, prilikom čega se nastojalo nadomjestiti njihove nedostatke. Aplikacija izrađena u sklopu ovog rada odgovorila je na sve postavljene zahtjeve, među kojima su se nalazili prijava i registracija korisnika, provjera radnog mjesta korisnika u tvrtki, tj. je li korisnik voditelj ili radnik, različita korisnička sučelja za pojedine funkcionalnosti, izrada i izdavanje radnih naloga, popunjavanje istih te slanje obavijesti prilikom izrade radnog naloga, dodjele posla radnicima i svih promjena napravljenih na izrađenom radnom nalogu. Detalji implementacije opisani su kroz dvije faze: korisničko sučelje i programski dio aplikacije. Korištene tehnologije su Android Studio kao razvojno okruženje, jezici Kotlin i XML te Firebase platforma za stvaranje baze podataka. Prilikom pisanja programskog koda aplikacije, velika pozornost bila je usmjerena na način komunikacije s bazom, njezinim kolekcijama i dokumentima.

Jednom kada je aplikacija postala gotov proizvod, dana je na testiranje skupu djelatnika iz tvrtke *Vinka plus d.o.o.* Na temelju korisničkog iskustva, stvorene su predodžbe o mogućim popravcima ove aplikacije. Implementacija opcije ispisa radnog naloga mogla bi biti vrlo korisna i dodatno olakšati podjelu radnih naloga. Izmjene izgleda sustava mogu se lagano napraviti, uz odgovarajuću i kreativnu ideju koja će ponuditi novi dizajn aplikacije. Kod koji brine o pretrazi korisnika ne mora se nužno prepravljati, nego mu se može samo nadodati još i pronalazak radnika putem imena, prezimena ili neke druge vrijednosti.

Na temelju postavljenih zahtjeva na programsko rješenje i dobivenih rezultata, dolazi se do zaključka da je aplikacija uspješno odgovorila na sve zadane upute. Kako bi to bilo izvedivo, neophodno je bilo imati ideju o samom izgledu aplikacije, provesti ideju u kod te osigurati visoku razinu preciznosti i usredotočenosti na aplikaciju tijekom njezine cjelokupne izrade.

LITERATURA

[1] Vinka plus. d.o.o., "Održavanje industrijskog pogona", Vinka plus d.o.o., Jarmina, 2019.

[2] Synesis, "Pupilla.hr", Synesis, [Mrežno]. Dostupno na: https://www.pupilla.hr/wp-content/themes/Synesis/details.php?id=PROrder3. [datum pristupa: 05. kolovoz 2023].

[3] C. R. China, "IBM", [Online]. Dostupno na: https://www.ibm.com/blog/work-order-process/. [datum pristupa: 06. kolovoz 2023.].

[4] Moja tvrtka, "Moja tvrtka", [Mrežno]. Dostupno na: https://www.mojatvrtka.hr/moj-tim/radni-nalozi. [datum pristupa: 29. lipanj 2023].

[5] Inc. UpKeep Technologies, "UpKeep", [Mrežno]. Dostupno na: https://www.upkeep.com/product/work-order-software/. [datum pristupa: 29. lipanj 2023].

[6] MaintainX Inc., "MaintainX", [Mrežno]. Dostupno na: https://www.getmaintainx.com/use-cases/work-order-management/. [datum pristupa: 29. lipanj 2023].

[7] K. Consulting, "KK Consulting", [Mrežno]. Dostupno na: https://kanban.com.hr/. [datum pristupa: 29. lipanj 2023].

[8] Fracttal.SpA.,"FracttalOne",[Mrežno].Dostupno na: https://play.google.com/store/apps/details?id=com.fracttalmobile.one&hl=en_US. [datum pristupa: 29. lipanj 2023].

[9] I.Limble,"Limble",[Mrežno].Dostupno na: https://www.softwareadvice.com/cmms/limblecmms-profile/. [datum pristupa: 29. lipanj 2023].

[10] Synchroteam,"Synchroteam",[Mrežno].Dostupno na: https://www.synchroteam.com/mobile.php. [datum pristupa: 29. lipanj 2023].

[11] I. Fiix, "Fiix", [Mrežno]. Dostupno na: https://www.fiixsoftware.com/cmms/mobile-cmms/. [datum pristupa: 29. lipanj 2023].

[12] I. Wrike, "Wrike", [Mrežno]. Dostupno na: https://www.wrike.com/apps/mobile-and-desktop/desktop-app/. [datum pristupa:a 29. lipanj 2023].

[13] Developers, "AndroidDevelopers", [Mrežno]. Dostupno na: https://developer.android.com/studio/intro. [datum pristupa: 06. rujan 2023].

[14] JetBrains, "IntelliJ IDEA", [Mrežno]. Dostupno na: https://www.jetbrains.com/idea/features/. [datum pristupa: 06. rujan 2023].

[15] Kotlin, "Kotlin Overview", [Mrežno]. Dostupno na: https://kotlinlang.org/docs/android-overview.html. [datum pristupa: 06. rujan 2023].

[16] E.I.Chidera, "freeCodeCamp", [Mrežno]. Dostupno na: https://www.freecodecamp.org/news/kotlin-vs-java-whats-the-difference/. [datum pristupa: 06. rujan 2023].

[17] W. Schools, "W3 Schools", [Mrežno]. Dostupno na: https://www.w3schools.com/xml/. [datum pristupa: 06. rujan 2023].

[18] D. Kirasić, "Osnovni koncepti: Sadržaj i oznake", u *XML tehnologija i primjena u sustavima procesne informatike*, Zagreb, Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva.

[19] Integrate.io, "Integrate.io", [Mrežno]. Dostupno na: https://www.integrate.io/blog/the-sql-vs-nosql-difference/. [datum pristupa: 06. rujan 2023].

[20] Firebase, "Firebase", [Mrežno]. Dostupno na: https://firebase.google.com/docs/firestore. [datum pristupa: 06. rujan 2023].

[21] Developers, "Android Developers", [Mrežno]. Dostupno na: https://developer.android.com/guide/topics/manifest/manifest-intro. [datum pristupa: 06. rujan 2023].

[22] Github, "Github", [Mrežno]. Dostupno na: https://github.com/bumptech/glide. [datum pristupa: 06. rujan 2023].

[23] V. Ahire, "tutorialsPoint", [Mrežno]. Dostupno na: https://www.tutorialspoint.com/when-to-use-fragments-vs-activities-in-android-app. [datum pristupa: 06. rujan 2023].

[24] J. T. Point, "Java T Point", [Mrežno]. Dostupno na: https://www.javatpoint.com/kotlin-data-class. [datum pristupa: 06. rujan 2023].

[25] Firebase, "Firebase", [Mrežno]. Dostupno na: https://firebase.google.com/docs/firestore/manage-data/transactions#transactions. [datum pristupa: 06. rujan 2023].

[26] FERIT, "LV7: Recyclerview i RestAPI", u *Osnove razvoja web i mobilnih aplikacija,*

Osijek, 2022.

[27] B.Ojha, "GeeksForGeeks", [Mrežno]. Dostupno na: https://www.geeksforgeeks.org/android-recyclerview/. [datum pristupa: 07. rujan 2023].

[28] Developers, "Android Developers", [Mrežno]. Dostupno na: https://developer.android.com/reference/android/app/AlertDialog. [datum pristupa: 06. rujan 2023].

SAŽETAK

Svaka tvrtka, neovisno kojim se područjem bavi, ima nadređene osobe koje svojim radnicima zadaju posao. Sav posao koji se obavi potrebno je negdje zabilježiti. Tada se izrađuje radni nalog, sa svim detaljima vezanim za odrađeni posao. Android aplikacija *WorkOrdersManager* za zadaću ima pretvoriti papirnati oblik radnog naloga u elektronički. Pomoću smjernica iz službene dokumentacije, stvaraju se zahtjevi na programsko rješenje. Za stvaranje aplikacije odabrano je razvojno okruženje Android Studio, programski jezik je Kotlin, a izgled aplikacije definiran opisnim jezikom XML. Za izradu baze podataka korištena je platforma Firebase te njene usluge *Authentication* i *Cloud Firestore*. Aplikacija nudi svojim korisnicima registraciju te prijavu odabranim identifikacijskim podacima. Ukoliko je prijavljeni korisnik voditelj, tada on može stvoriti novi radni nalog. Također može stvoriti novi tim s radnicima po želji. Ukoliko je trenutni korisnik radnik, on tada među svojim radnim nalozima može urediti one koji nisu završeni ili pregledati one koji jesu, a među timovima pogledati sve one kojima je on dodijeljen. Obje vrste korisnika imaju mogućnost uređivanja profila, pregleda svojih obavijesti, svih radnih naloga i timova. Na kraju je aplikacija testirana na skupu djelatnika tvrtke *Vinka plus d.o.o.,* koji su ocjenjivali svoje zadovoljstvo korištenja aplikacije te su iznosili prijedloge o mogućim izmjenama u aplikaciji.

Ključne riječi : Android, Firebase, mobilna aplikacija, radni nalog, XML

ABSTRACT

Android application for managing work orders

Every company, regardless of its field of business, has superiors who assign tasks to their employees. Naturally, all the work that is done needs to be documented somewhere. This is when a work order is created, with all the details related to the completed task. The Android application WorkOrdersManager, as its task, aims to convert the paper-based format of work orders into electronic ones. Following the guidelines from the official documentation, requirements for the software solution are being created. Android Studio was chosen as the development environment for building the application, where the code will be written in the Kotlin programming language, and the layout will be defined using XML-like markup language. Firebase platform has been used for creating the database, along with its services Authentication and Cloud Firestore. The application offers registration and login functionality to its users using selected identification data. If the logged-in user has the status of a Director, they can create new work orders and create new teams with workers as desired. If the current user is a Worker, they can edit the work orders that are not completed among their assignments or review those that are completed. Among the teams, they can view all the ones they are assigned to. Both types of users have the ability to edit their profiles, view their notifications, all work orders, and teams. In the end, the application was tested with the employees of *Vinka plus d.o.o*. company, who evaluated their satisfaction with using the application and provided suggestions for possible changes.

Key words: Android, Firebase, mobile application, work order, XML

ŽIVOTOPIS

Magdalena Knežević rođena je 07.10.2001. u Vinkovcima, sa prebivalištem u Jarmini. Godine 2008. upisuje Osnovnu školu Matija Gubec Jarmina, te ju završava 2016. Iste godine upisuje Gimnaziju Matija Antun Reljković Vinkovci, smjer opća gimnazija. Završava srednju školu 2020. godine, kada upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek u Osijeku. Odabire prijediplomski studij Računarstva, smjer Programsko inženjerstvo, gdje i trenutno studira.

PRILOZI

P.5.1. Poveznica na GitLab repozitorij sa cijelim kodom aplikacije

- https://gitlab.com/magdalena.knezevic/zavrsni-rad

P.6.1. Poveznica na Google Forms anketu

- https://docs.google.com/forms/d/e/1FAIpQLSd4aeU0m0DWDktGDW7p2CwobByLrov89uSVdDBFN3ZSqIFi9Q/viewform?usp=sf_link