Metode i mobilna aplikacije za potrebe vođenja IT projekata

Biočić, Petar

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

Sveučilišni studij

METODE I MOBILNA APLIKACIJA ZA POTREBE VOĐENJA IT PROJEKATA

Diplomski rad

Petar Biočić

Osijek, 2023.

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

Obrazac D1: Obrazac za imenovanje Povjerenstva za diplomski ispit

Osijek, 03.12.2023.

Odboru za završne i diplomske ispite

Imenovanje Povjerenstva za diplomski ispit

Ime i prezime Pristupnika:	Petar Biočić
Studij, smjer:	Automobilsko računarstvo i komunikacije
Mat. br. Pristupnika, godina upisa:	D-58ARK, 06.10.2021.
OIB studenta:	87147859385
Mentor:	prof. dr. sc. Dominika Crnjac Milić
Sumentor:	izv. prof. dr. sc. Zdravko Krpić
Sumentor iz tvrtke:	Srđan Kovačević, dipl.ing.stroj. MSc
Predsjednik Povjerenstva:	prof. dr. sc. Krešimir Nenadić
Član Povjerenstva 1:	prof. dr. sc. Dominika Crnjac Milić
Član Povjerenstva 2:	izv. prof. dr. sc. Alfonzo Baumgartner
Naslov diplomskog rada:	Metode i mobilna aplikacije za potrebe vođenja IT projekata
Znanstvena grana diplomskog rada:	**Programsko inženjerstvo (zn. polje računarstvo)**
Zadatak diplomskog rada:	Radom je potrebno dati opis i analizu različitih metodologija vođenja projekata u IT industriji s naglaskom na projekte izrade programskih rješenja. U praktičnom dijelu rada treba napraviti mobilnu aplikaciju koja će se omogućiti jednostavnije i transparentnije vođenje malih timova tijekom provedbe projekta. Ona treba omogućavati analizu i uvid u projektni tijek, zauzetost resursa te kontrolu obavljenog posla u predviđenom vremenu s predviđenim troškovima. Aplikacija treba biti primjenjiva za Android i iOS sustave te je pri njezinoj izradi potrebno koristiti SDK koji omogućuje izradu multiplatformskih mobilnih aplikacija. Mentorica (FERIT): Prof.dr.sc. Dominika Crnjac Milić; Sumentor (FERIT): Izv.prof.dr.sc. Zdravko Krpić; Sumentor (Orqa): Srđan Kovačević, dipl.ing.stroj., MSc (Oxon) TEMA JE REZERVIRANA
Prijedlog ocjene pismenog dijela ispita (diplomskog rada):	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 2 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene od strane mentora:	03.12.2023.

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, 12.12.2023.

Ime i prezime studenta:	Petar Biočić
Studij:	Automobilsko računarstvo i komunikacije
Mat. br. studenta, godina upisa:	D-58ARK, 06.10.2021.
Turnitin podudaranje [%]:	8

Ovom izjavom izjavljujem da je rad pod nazivom**: Metode i mobilna aplikacije za potrebe vođenja IT projekata**

izrađen pod vodstvom mentora prof. dr. sc. Dominika Crnjac Milić

i sumentora izv. prof. dr. sc. Zdravko Krpić

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

Projektni menadžment u industriji informacijskih tehnologija (engl. *Information Technology, IT*) igra ključnu ulogu u uspješnoj realizaciji i isporuci proizvoda koji nastaje kao rezultat projekta. Uz brz razvoj tehnologije i stalne promjene u zahtjevima tržišta, projektni timovi se suočavaju s kompleksnim izazovima u upravljanju IT projektima. Kako bi postigle uspješne rezultate, one se oslanjaju na učinkovite metode projektnog menadžmenta koje omogućuju praćenje, upravljanje i usklađivanje aspekata projekta. Projektni menadžment se odnosi na primjenu specifičnih znanja, vještina, alata i tehnika kako bi se ostvarili ciljevi projekta unutar definiranih ograničenja, poput vremenskog okvira, resursa, proračuna i kvalitete. U kontekstu IT projekata, projektni menadžment igra još veću ulogu zbog složenosti tehnoloških zahtjeva, dinamičnosti industrije i interakcije s korisnicima.

Zadatak rada je analizirati sekvencijske i agilne metodologije vođenja IT projekata i izraditi programsko rješenje u obliku mobilne aplikacije koja će omogućiti jednostavnije i transparentnije vođenje malih timova tijekom provedbe projekta. Mobilna aplikacija treba omogućiti osnovne alate za analizu i upravljanje projektom te biti primjenjiva za različite mobilne operacijske sustave.

Prvim poglavljem je dan uvod u temu, u drugom poglavlju detaljnije je opisano značenje projektnog menadžmenta u IT industriji uz usporedbu sekvencijskih i agilnih metodologija vođenja IT projekata. U trećem poglavlju analizirano je trenutno stanje na tržištu mobilnih aplikacija za upravljanje projektima i usporedba vlastitog rješenja s konkurencijom. U četvrtom poglavlju opisano je programsko rješenje podijeljeno u tri cjeline: idejno rješenje, izradu i analizu programskog rješenja. Zaključak rada dan je u zadnjem poglavlju u kojem se navode moguća proširenja programskog rješenja.

2. PROJEKTNI MENADŽMENT U IT INDUSTRIJI

U današnjem dobu, informacijska tehnologija igra ključnu ulogu u gotovo svim aspektima poslovanja. Upravljanje IT projektima zahtijeva poseban pristup koji je prilagođen brzim promjenama, visokoj složenosti i zahtjevima tržišta. Kako bi se uspješno suočile s tim izazovima, organizacije se oslanjaju na projektni menadžment kao ključnu disciplinu koja pruža okvir za upravljanje i isporuku rješenja proizašlih iz IT projekata. [1]

Također, upravljanje informacijama, koje predstavljaju najvrjedniju robu u današnjem poslovnom svijetu, oblikuje važnost uloga u poslovanju. Dinamično tržište i društvo stvaraju različite faktore koji se brzo mijenjaju usred poslovnih operacija te zbog toga poduzeća moraju biti svjesne uvjeta na tržištu i promjena u njemu u svakom trenutku. Upravljanje tehnologijom radi podrške poslovnim operacijama predstavlja važan dio upravljačkih operacija u većini poduzeća te iz tog razloga postoji snažna potreba za stvaranjem organizacije učenja i organizacije u kojoj informacije koje pojedinci nauče dijele s timom. Tehnologija je jedini način na koji takvo stanje može postojati na pravovremen i opsežan način. [2]

Projektni menadžment u IT industriji zahtijeva kombinaciju posebnih vještina, tehnologija i metodologija kako bi se osiguralo uspješno vođenje projekata. Projektni menadžeri moraju posjedovati širok spektar vještina koje uključuju tehničko razumijevanje informacijskih tehnologija, upravljanje resursima, komunikacijske vještine i vještine vođenje timova. Jedna od ključnih karakteristika projektnog menadžmenta u IT industriji je njegova sposobnost prilagodbe promjenama. U kontekstu brzog razvoja tehnologije, projektni menadžeri moraju biti svjesni novih trendova i tehnologija te ih primijeniti u svojim projektima kako bi ostali konkurentni. Uvođenje novih tehnologija i sve brži napredak zahtjeva sve veće razumijevanje njihovog utjecaja na projektne aktivnosti i mogućnosti koje pružaju za poboljšanje učinkovitosti i produktivnosti. [1]

2.1. Povijesni razvoj projektnog menadžmenta u IT industriji

Razvoj projektnog menadžmenta započeo je u drugoj polovici 20. stoljeća napretkom tehnologije i pojavom složenijih tehnoloških procesa. [3] Jedna od ključnih prekretnica u razvoju projektnog menadžmenta je bila uspostava Projektnog menadžment instituta (PMI) 1969. godine. PMI je postao jedna od najutjecajnijih organizacija u području projektnog menadžmenta. [4]

U ranim fazama razvoja IT industrije, projektni menadžment bio je usredotočen na razvoj softvera. Tradicionalni vodopadni model projektnog menadžmenta bio je dominantan, gdje su projekti podijeljeni na slijedne faza, poput analize zahtjeva, dizajna, implementacije i testiranja. Međutim, s rastom kompleksnosti projekata i potrebom za brzom isporukom, agilne metode postale su popularne u IT industriji. Agilne metode projektnog menadžmenta, kao što su *Scrum* i *Kanban*, razvile su se tijekom 1990-ih godina kao odgovor na potrebu za fleksibilnošću i brzom isporukom softvera. Ove metode se fokusiraju na iterativni i inkrementalni razvoj, omogućavajući timovima da se prilagođavaju promjenama zahtjeva tijekom projekta. Agilni pristup podrazumijeva kontinuiranu suradnju, brze petlje povratnih informacija i brzo učenje. [4]

Sredinom 1990-ih godina, internet je počeo mijenjati gotovo svaki poslovni proces. Pružio je brz i interaktivan novi medij. Krajem 1995. zajednica upravljanja projektima usvojila je internetsku tehnologiju kako bi postala učinkovitija u kontroli i upravljanju projektima. [5] Uz napredak tehnologije, projektni menadžeri su se morali prilagođavati i usvajati nove alate i metode kako bi iskoristili prednosti koje tehnologija pruža. Danas, projektni menadžment u IT industriji nastavlja evoluirati kako bi se prilagodio brzim promjenama i zahtjevima tržišta. Hibridni pristupi, koji kombiniraju elemente tradicionalnog i agilnog projektnog menadžmenta, postaju sve popularniji. Navedeni pristupi omogućavaju fleksibilnost u vođenju projekata, uz istovremeno osiguravanje strukture i upravljanja. [4]

Upotreba specijaliziranog softvera za upravljanje projektima olakšala je primjenu alata i tehnika, stoga je razvoj softvera značajno pridonio razvoju alata i tehnika upravljanja projektima. [6]

2.2. Životni ciklus IT projekta

Najveći broj poslova u okviru IT poduzeća obavlja se putem projekata. Uspjeh rada poduzeća ovisi od uspjeha pojedinačnog projekta ili programa. [7] Pojam projekta se rabi za opis aktivnosti koje organizacije obavljaju povremeno i prema potrebi. Projekt predstavlja jedinstvene radnje dok pojam programa označava skupine međusobno povezanih projekata. [8] Projekt predstavlja događaj koji je vremenski ograničen, dakle projekt se rađa, živi i umire. [7]

Životni ciklus razvoja softverskog projekta je opisni model koji obuhvaća sve faze kroz koje softverski projekt prolazi. Struktura životnog ciklusa projekta se može raščlaniti na sljedeće faze: početak projekta, organiziranje i priprema, izvršavanje rada te zatvaranje projekta. [9] Zatvaranje projekta u slučaju IT projekata, koji su većinom softverska rješenja, može se preformulirati u fazu održavanja.

Ovaj ciklus detaljnije obuhvaća aktivnosti kao što su analiza zahtjeva, dizajn, implementacija, testiranje, uvođenje i održavanje softvera. [10] Svaka faza ima svoje specifične ciljeve, zadatke i rezultate koji su detaljnije opisani u poglavlju 2.3.1.

Početna faza ovog procesa je planiranje, koja uključuje analizu domene sustava kako bi se razumjelo što se očekuje od budućeg programa te se analizira postojeći sustav slične namjene kako bi se razvila percepcija o funkcionalnosti i namjeni programa u budućnosti. Sljedeći korak je analiza zahtjeva, koja se odnosi na specificiranje obilježja sustava koja će zadovoljiti očekivanja, želje i potrebe korisnika. U fazi dizajniranja detaljno se opisuje sustav koji se izrađuje. Ovdje se razvija generalna arhitektura programa i specifikacija svake komponente. Jasno je da su tri navedene faze međusobno povezane i često se proučavaju kao jedna, kompleksnija grupa. Razvoj sustava predstavlja kompleksniju fazu u kojoj programeri stvaraju budući softverski sustav ili rješenje. Nakon razvoja, slijedi testiranje sustava i njegova integracija. U toj fazi također se obavlja dokumentiranje. Testiranje sustava je također važna faza za buduće održavanje sustava. [10]

Vidljivo je da je ciklus razvoja projekta kontinuiran i cikličan. To znači da se sustav može redizajnirati i dalje razvijati. Životni ciklus sustava može se revitalizirati umjesto da doživi konačan završetak. [10]

2.3. Metodologije razvoja softverskih rješenja

Metode razvoja softvera mogu se podijeliti na dvije skupine: sekvencijske i agilne. Dvije skupine metoda se razlikuju u mnogočemu, pri čemu sekvencijske metode koriste tradicionalne pristupe, dok se agilne metode smatraju suvremenim i specijaliziranim. Detaljnija analiza i razlike metodologija objašnjene su u nastavku ovog poglavlja.

Sekvencijske metode, kao što je već istaknuto, temelje se na tradicionalnim metodama razvoja softvera. Nastale su u 70-ima prošlog stoljeća i karakterizira ih jasna podjela na faze koje opisuju različite aspekte životnog ciklusa softvera. Svaka faza je pažljivo definirana i provodi se prema precizno određenim aktivnostima i postupcima. [10]

Ove metode imaju za cilj i obvezu dokumentiranja svake faze procesa razvoja softvera. Često se nazivaju i klasičnim metodama te obuhvaćaju nekoliko modela upravljanja životnim ciklusom softvera. Detaljnija analiza postojećih sekvencijskih metodologija obrađena je poglavlju 2.3.1.

Agilne metodologije su suvremeni pristupi u upravljanju projektima, posebno u kontekstu softverskog razvoja. One su nastale kao odgovor na izazove tradicionalnih, sekvencijskih metoda koje se temelje na pred definiranim planovima i strogim procesima. Agilne metode se ističu fleksibilnošću, adaptivnošću i suradničkim pristupom te su usmjerene na isporuku vrijednosti korisnicima u kratkim iteracijama. [11]

Glavna obilježja agilnih metodologija uključuju:

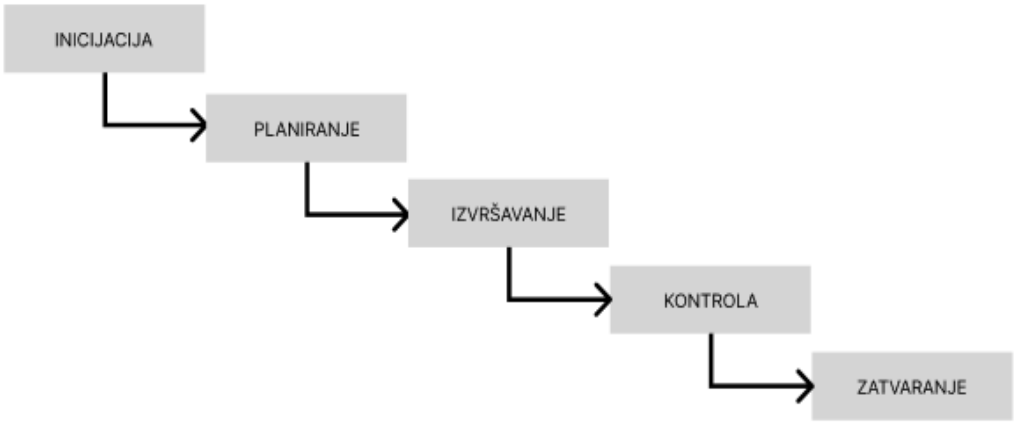
- Iterativni i inkrementalni pristup: Rad se podijeli na manje cikluse (iteracije) tijekom kojih se isporučuje funkcionalan dio proizvoda (inkrement). Time se omogućava kontinuirano usklađivanje s promjenama i brza povratna informacija od korisnika.
- Suradnički timski rad: Agilne metode promoviraju blisku suradnju između članova tima i korisnika. Timovi su multidisciplinarni i samoupravljajući, a komunikacija se smatra ključnim faktorom za uspješnu isporuku.
- Prioritizacija: Zahtjevi se evaluiraju i prioritiziraju na temelju vrijednosti koju donose korisnicima. Promjene i nove informacije se dobro prihvaćaju i integriraju tijekom razvojnog procesa.
- Kontinuirano poboljšavanje: Agilne metode potiču refleksiju i učenje kako bi se kontinuirano poboljšavali procesi i isporučivao bolji proizvod.

Neki od poznatih agilnih okvira i metodologija uključuju *Scrum*, *Kanban* i *Lean*. Svaka od tih metodologija ima svoje specifičnosti, ali zajedničko im je da promiču vrijednosti agilnosti i adaptivnosti. Agilne metode su postale popularne zbog sposobnosti brzog odgovora na promjene, smanjenja rizika, poboljšanja kvalitete proizvoda i povećanja zadovoljstva korisnika. One se često koriste u dinamičnim okruženjima u kojima se zahtjevi mijenjaju tijekom vremena. [11]

2.3.1. Analiza sekvencijske metodologije

Sekvencijske metodologije podrazumijevaju pet faza koje je potrebno u točno određenom slijedu izvršiti kako bi se projekt završio. Faze sekvencijske metodologije prikazane su na slici 2.1.

U inicijacijskoj fazi projekta definira se obujam projekta ili nova faza postojećeg projekta. Planiranje, kao druga faza donosi zadatke i ciljeve te se definira način postizanja cilja. Izvršavanje predstavlja treću fazu u kojemu se izvršava posao definiran u planu. Kontrola je faza u kojoj se prate procesi i donosi uvid u mjerilo ostvarenog u odnosu na planirano. Posljednja faza predstavlja zatvaranja projekta. [12]

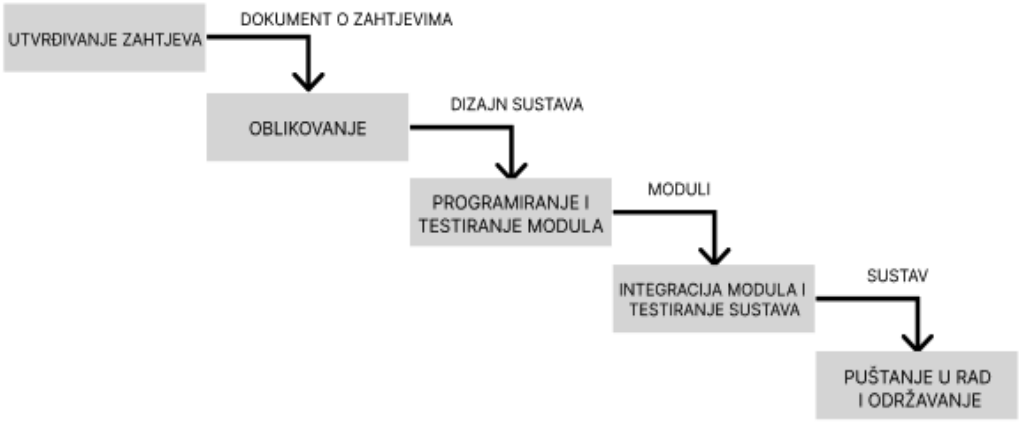


Slika 2.1. Opće faze sekvencijske metodologije vođenja projekta, izrađeno prema [12]

Razvoj softverskih rješenja zahtjeva prilagođeni opći model faza sekvencijskih metodologija. Jedan od najpoznatijih pristupa u projektima softverskog tipa jest vodopadni model razvoja (engl. *Waterfall development model*) kojeg sve učestalije zamjenjuje agilna metodologija upravljanja. [13]

Na slici 2.2. dan je uvid u prilagođene nazive faza u vodopadnom modelu razvoja. Svaka faza u vodopadnom modelu razvoja softvera ima produkt. Nakon utvrđivanja zahtjeva produkt je

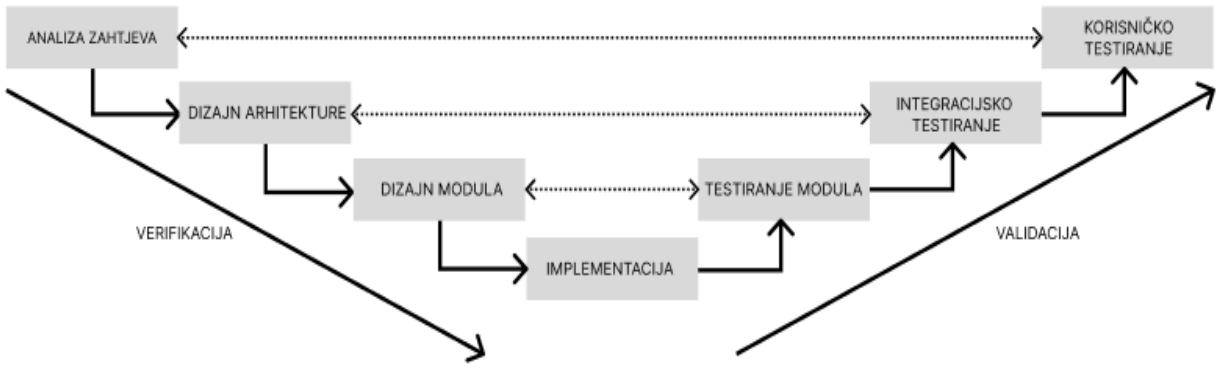
dokument o zahtjevima nakon čega kreće faza oblikovanja dizajna iz čega proizlazi dizajn sustava. Nakon programiranja i testiranja modula, modul postaje produkt treće faze koji ulazi u fazu integracije i testiranja unutar sustava. Nakon uspješne integracije i testiranja sustav predstavlja produkt koji se pušta u rad i na održavanje. [10]



Slika 2.2. Faze vodopadnog modela razvoja softvera, izrađeno prema [10]

Nadogradnja na vodopadni model naziva se V-model koji u prvi plan stavlja analizu prethodne faze. U slučaju V-modela, tijek aktivnosti ne ide samo u jednom smjeru kao u slučaju vodopadnog modela, nego ima povrat prema gore. [10]

Tijek aktivnosti prema dolje predstavlja verifikaciju koja predstavlja provjeru odgovara li softver specifikaciji, dakle dokumentu o zahtjevima. Tijek aktivnosti prema gore predstavlja validaciju koja predstavlja provjeru zadovoljava li softver potrebama korisnika. V-model omogućuje vraćanje iz proces koji dolaze kasnije u procese koji su izvršeni (Slika 2.3). [14]



Slika 2.3. Faze razvoja u V-modelu, izrađeno prema [14]

Vodopadni i V-model nisu jedine sekvencijske metode. Osim njih, sljedeće dvije metode se koriste pri razvoju softvera:

- Prototipiranje: Prototip predstavlja brzo i jednostavno razvijen program koji kao zadatak ima oponašati budući sustav. Prednost prototipa jest lako mijenjanje i nadograđivanje te predstavlja jeftiniji i jednostavniji pristup razvoju softvera. [15]
- Inkrementalni razvoj: Ideja u inkrementalnom razvoju jest isporuka samostalnih radnih cjelina sustava koja se radi u iteracijama. Inkrementi se dostavljaju korisniku i potvrđuju te se ističe važnost povratne informacije od strane korisnika koja utječe na buduće inkremente. U inkrementalnom razvoju struktura mora biti uspostavljena prije nego su zahtjevi sustava potpuni što utječe na ograničenost naknadnih promjena. [10]

Zaključak analize sekvencijskih metodologija jest takav da sekvencijske metodologije predstavljaju tradicionalni način provedbe projekta u kojemu su korisnici aktivno uključeni u razvoj te je konačni sustav ukoliko je prošao kroz sve faze uspješno jednostavniji za korištenje i održavanje. Zbog većeg fokusa na planiranju i razvoju dizajna, za koje je potrebno imati stručnjake, ovakvi sustavi imaju manje nepotrebnih mogućnosti te visoku razinu kvalitete dizajna.

2.3.2. Analiza agilnih metodologija

Agilne metodologije i pristup nastali su iz potrebe za povećanjem uspješnosti projekta što uključuje bolje upravljanje resursima. Agilne metode skup su metoda i karakteristika pri vođenju projekata koji se razlikuju od načela sekvencijskog modela. U nastavku ovog poglavlja predstavljen je najkorišteniji za potrebe programskih rješenja unutar agilne metodologije *Scrum*, osvrt na druge okvire i specifičnosti agilne metodologije.

Odjeljak analize *Scrum* radnog okvira u nastavku ovog poglavlja služi kao podloga za izradu programskog rješenja pošto programsko rješenje sadrži elemente *Scrum* radnog okvira kao što su *sprintevi* i agilni tipovi zadataka.

Scrum

Scrum je popularni programski okvir procesa, koji se koristi za upravljanje razvojem kompleksnih rješenja. Svoju upotrebu je pronašao u razvoju softverskih rješenja te se velika većina IT kompanija već koristi ili upoznaje s ovim okvirom procesa. Programski okvir ovog tipa sastoji se od *Scrum* timova, uloga, događaja, artefakata i pravila predstavljeni u tablicama 2.1, 2.2 i 2.3. [16]

Tablica 2.1. Raspodjela uloga unutar Scrum tima

Naziv uloge	**Uloga i zaduženja**
Vlasnik proizvoda	Osoba odgovorna za maksimizaciju vrijednosti proizvoda i rada razvojnog tima koja upravlja proizvodnim *backlogom.*
Razvojni tim	Profesionalci koji rade posao isporučujući inkrement proizvoda na kraju svakog perioda razvoja (engl. *sprint*).
Scrum master	Odgovoran za provođenje *Scrum* okvira.

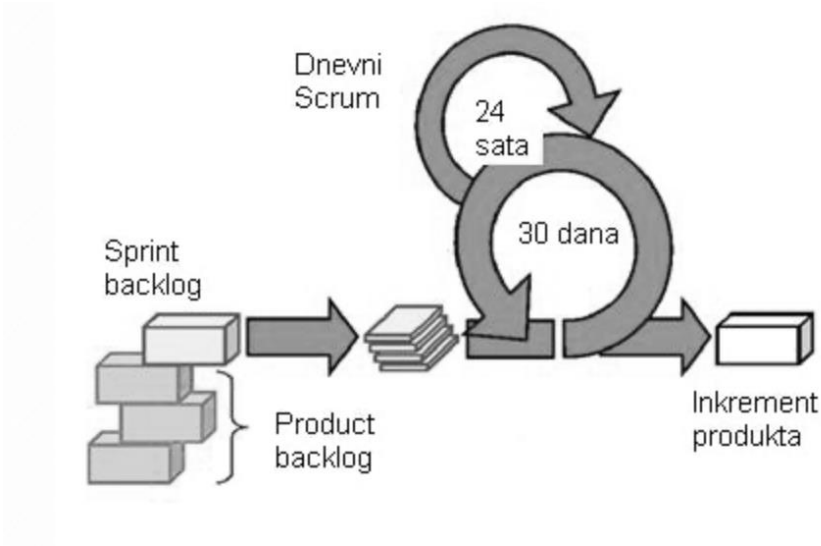
Tablica 2.2. Događaji u Scrum okviru

Događaji	**Opis**
Sprint	Jezgra *scruma,* vremenski ograničen period tijekom kojega se proizvede upotrebljiv inkrement proizvoda.
Dnevni *scrum* sastanak	Svakodnevni, vremenski ograničen događaj koji služi za usklađenje razvojnog tima i usvajanje plana za sljedećih 24 sata.
Sastanak revizije *sprinta*	Sastanak koji je na rasporedu pred kraj *sprinta* u kojemu se radi revizija odrađenog posla u trenutnom *sprintu.*
Sastanak planiranja *sprinta*	Sastanak na početku *sprinta* u kojemu se zadaci prioritiziraju i raspodjeljuju razvojnom timu.

Tablica 2.3. Popis artefakata u Scrumu

Artefakti	**Opis**
Proizvodni *backlog*	Sortiran popis zadataka potrebnih za realizaciju krajnjeg proizvoda.
Sprint backlog	Skup stavki iz proizvodnog *backloga* odabran za *sprint*

Tijek *Scrum* programskog okvira prikazan je na slici 2.4. U prikazanom slučaju *sprint* traje trideset dana koliko iznosi predloženi maksimum.



Slika 2.4. Prikaz tijeka Scrum okvira [17]

Osvrt na ostale agilne radne okvire

Pored *Scruma* postoji mnoštvo agilnih radnih okvira no ne odgovara svaki radni okvir za implementaciju u razvoj programska rješenja. Poznatiji agilni okviri koji se mogu iskoristiti u svrhu razvoja programskog rješenja predstavljeni su u tablici 2.4. U desnom stupcu nabrojane su specifičnosti okvira koje su različite od *Scrum* modela.

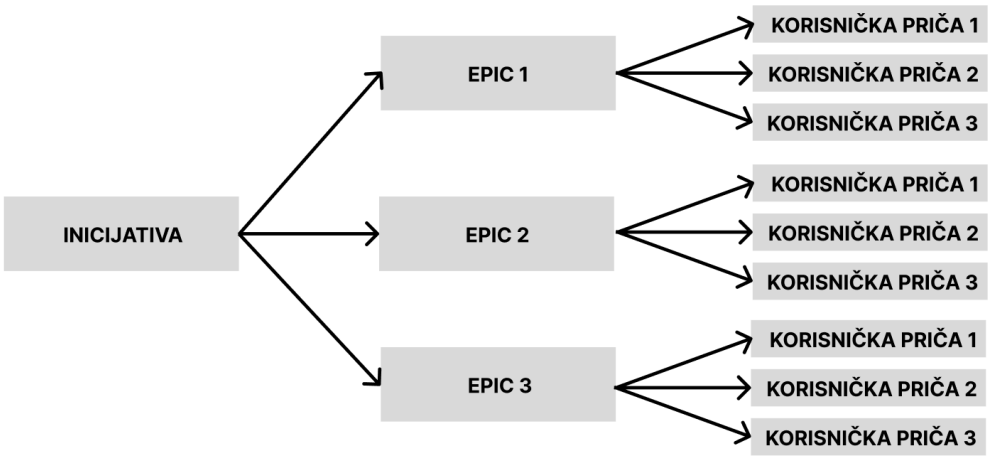
Tablica 2.4 Agilni radni okviri i njihove specifičnosti

Agilni radni okvir	Specifičnosti
Kanban	- Razvoj se bazira na vizualizaciji radnog tijeka - Zadaci koji su u tijeku izvođenja imaju prioritet - Nema ograničenog perioda za razvoj komponente - Moguće mijenjanje plana razvoja u bilo kojem trenutku

Hibridni	- Spoj agilne i sekvencijske metodologije - Koriste se specifičnosti iz vodopadne metode kao što su fiksan period izvođenja, projicirani budžet, analiza rizika
Lean	- Okvir koji promiče brzi softverski razvoj s malim utroškom vremena i resursa - Što kraći razvojni periodi - Isporučen proizvod se konstantno održava - Razvojni tim je samostalan i ima veću odgovornost

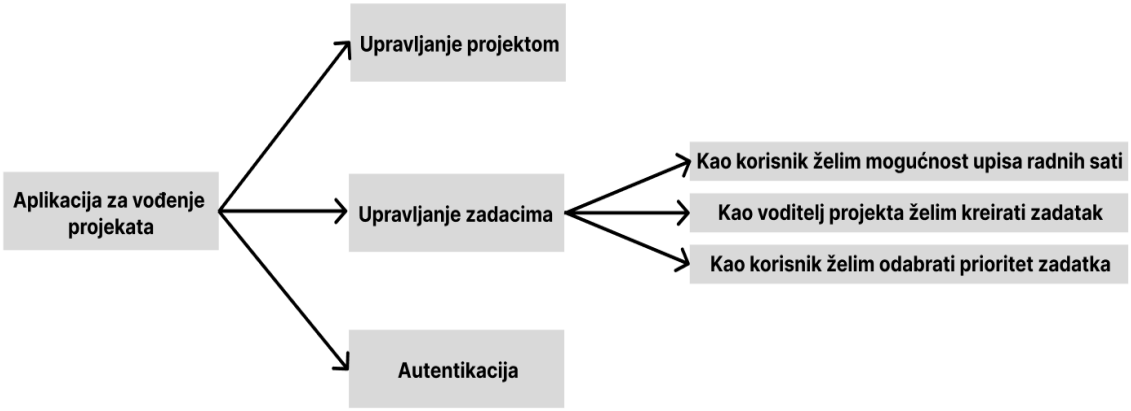
Agilni tipovi zadataka

Posebna struktura raspisa zadataka u hijerarhijskoj ovisnosti specifična je za agilne okvire koja pridonosi boljoj organizaciji posla. Zadaci mogu biti raspisani kao: inicijative, *epici* i korisničke priče. Korisničke priče predstavljaju kratke zahtjeve raspisane u stilu ili od strane krajnjeg korisnika. *Epic* predstavlja veće blokove posla koji se ostvaruju realizacijom više korisničkih priča. Analogno tome, incijative predstavljaju još veće blokove posla koji se ostvaruju realizacijom svih *epica.* [18] Hijerarhijska shema odnosa inicijative, *epica* i korisničkih priča prikazana je na slici 2.5.



Slika 2.5. Hijerarhijski odnos agilnih tipova zadataka

Primjer raspodjele zadataka po agilnim tipovima zadataka pri izradi softverskog rješenja prikazan je na slici 2.6. Aplikacija za vođenje projekata korištena je za primjer pri raspisu agilnih tipova zadataka. Aplikacija predstavlja inicijativu, *epice* predstavljaju blokovi: upravljanje zadacima, upravljanje projektom i provjera identiteta. *Epic* pod nazivom „Upravljanje zadacima“ pod sobom ima tri korisničke priče čijom se realizacijom kompletira *Epic* koji će ponuditi modul za upravljanje zadacima unutar aplikacije.



Slika 2.6. Primjer hijerarhijskog raspisa agilnih tipova zadataka

2.3.3. Usporedba sekvencijskih i agilnih metodologija

U ovom poglavlju prikazana je usporedba sekvencijskih i agilnih metodologija pri razvoju softverskih rješenja. Usporedbom sekvencijskih i agilnih metodologija dan je osvrt na prednosti i nedostatke sekvencijalnih i agilnih metodologija i razlog odabira agilne metodologije u svrhu upravljanja modernim IT projektima i izrade programskog rješenja za upravljanje IT projektima.

Tablica 2.5. Prednosti i nedostaci sekvencijskih i agilnih metodologija

	Sekvencijske metodologije	**Agilne metodologije**
Prednosti	- Jasno definirani ciljevi - Struktura i redoslijed - Pouzdanost i predvidljivost	- Fleksibilnost - Brza isporuka vrijednosti - Kontinuirana povratna informacija
Nedostaci	- Manjak fleksibilnosti - Rizik promašaja - Produženi vremenski okviri	- Manjak formalnog planiranja - Nestabilnost zahtjeva - Ovisnost o timskoj dinamici

U konačnici, uvidom u tablicu 2.5. koja prikazuje prednosti i nedostatke pojedine metodologije u odnosu na drugu, nijedna metodologija nije savršena i ona mora odgovarati preferencijama kompanije, korisnika i tima. Zbog porasta razvoja aplikacija i programskih rješenja s napretkom tehnologije sve je više aplikacija koje moraju biti u doticaju s vremenom zbog konkurencije, stoga agilne metodologije pružaju bolji odgovor na dinamičan razvoj aplikacije u kojima se promjene nad zahtjevima događaju gotovo svakodnevno.

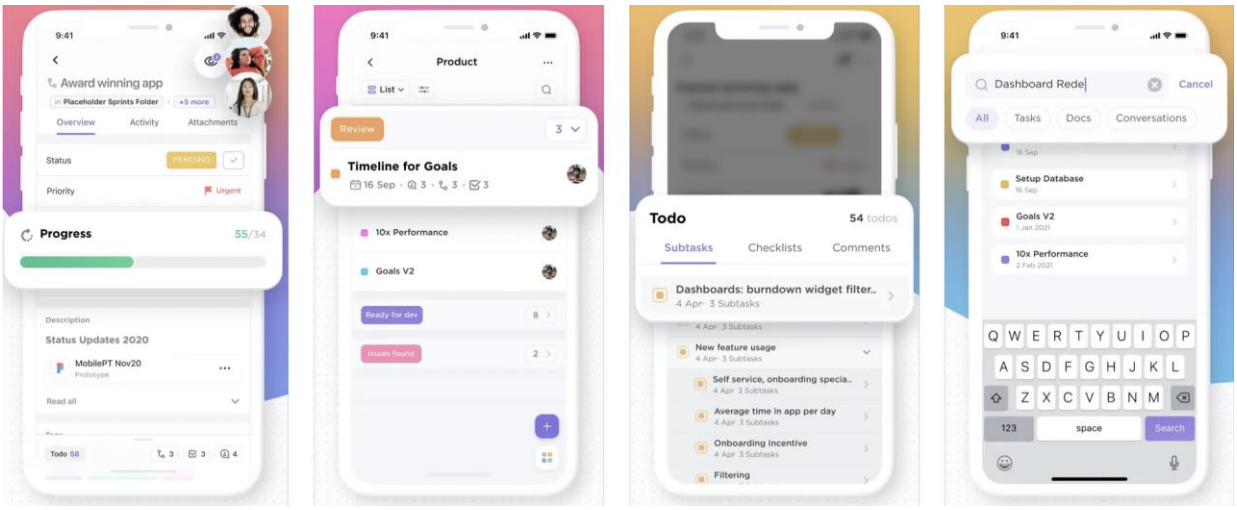
3. MOBILNE APLIKACIJE ZA UPRAVLJANJE PROJEKTIMA

Tržište softverskih rješenja za potrebe vođenja projekata je u stalnom porastu u posljednjih deset godina te odabir rješenja za organizacije ovisi o njihovim zahtjevima, načinu rada i politici same organizacije. Niti jedan softver za vođenje projekata nije isti kao drugi te svaki od njih ima značajku koju drugi nema ili je napravljena intuitivnije i funkcionalnije. Softverska rješenja ovog tipa primarno služe pri upotrebi na računalu jer je moguće više funkcionalnosti objediniti unutar aplikacije ili mrežna (engl. *web*) sučelja na računalu. Verzije za mobilne uređaje većinom predstavljaju pojednostavljene verzije istih te ne sadrže sve funkcionalnosti aplikacija za računalo ili *web* sučelje.

Kako je za potrebe diplomskog rada zadatak izrada mobilne aplikacije za vođenje IT projekata, ovo poglavlje služi kao pregled područja teme u kojem je dan uvid u postojeća rješenja za vođenje IT projekata na mobilnim platformama i nedostaci koje će programsko rješenje diplomskog rada pokušati riješiti.

3.1. Mobilna aplikacija ClickUp

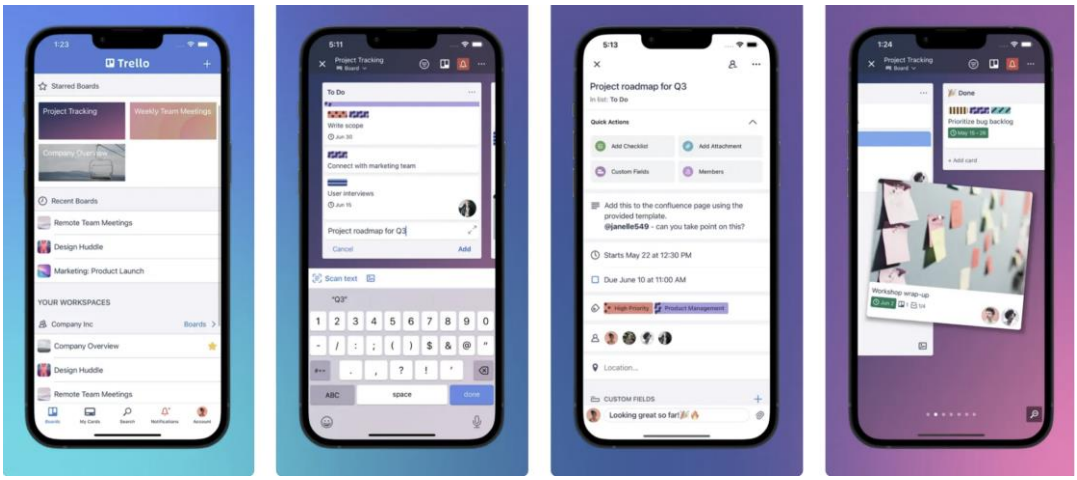
ClickUp je mobilna aplikacija za upravljanje zadacima i projektima koja se ističe intuitivnim dizajnom i popularnošću u alatima za vođenje svih vrsta projekata. Predstavlja pojednostavljenu verziju *web* aplikacije *ClickUp*. Omogućava timsku suradnju unutar više projektnih prostora (engl. *project spaces).* Osim osnovnih funkcionalnosti pri upravljanju projektom kao što su upravljanje zadacima i praćenje vremena provedbe aktivnosti, aplikacija nudi mogućnosti pisanja dokumenata i komunikaciju u stvarnom vremenu između članova projektnog tima. Zbog svojih funkcionalnosti korištenje mobilne verzije ovisno je o korištenju *web* verzije čija cijena po korisniku može predstavljati veliki trošak za male timove. Novčani iznos korištenja *ClickUp* ovisi o broju korisnika unutar organizacije te zbog svoje visoke cijene po jednom članu predstavlja najskuplji alat na tržištu. Glavnu značajku *ClickUp* mobilne aplikacije predstavlja intuitivni i vizualno primamljiv grafički dizajn koji sadrži moderne zaobljene elemente i animacije uz mogućnost personalizacije izgleda. Grafički dizajn je vidljiv na nizu zaslona na slici 3.1. Nedostatak aplikacije *ClickUp* koji se ističe jest visoka cijena korištenja dodatnih funkcionalnosti poput stvaranja gantograma i sprinteva. Navedeni problem je riješen u aplikaciji izrađenoj za potrebe diplomskog rada na način da je korištenje aplikacije besplatno.



Slika 3.1. Prikaz korisničkog sučelja ClickUp mobilne aplikacije [19]

3.2. Mobilna aplikacija Trello

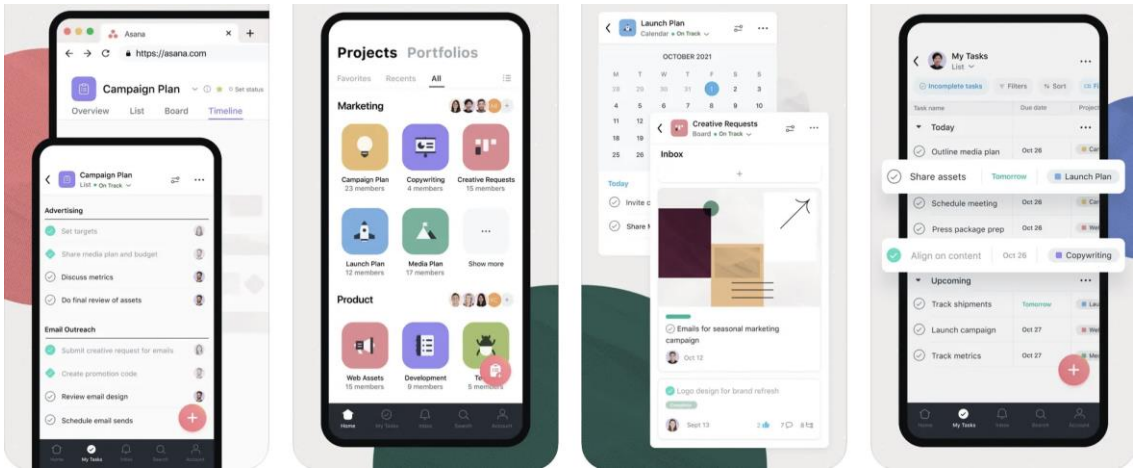
Trello je mobilna aplikacija koja fokus stavlja na organizaciju projekta putem *Kanban* ploča i kartičnog dizajna rasporeda zadataka vidljivog na posljednjoj sličici u nizu na slici 3.2. Prebacivanje zadataka iz jednog statusa u drugi izvršava se povuci i pusti (engl. *drag and drop)* pristupom. Također, kao i u slučaju *ClickUp* aplikacije, *Trello* mobilna verzija je pojednostavljena *web* aplikacija pod istim nazivom. *Trello* pruža besplatno korištenje alata, ali s ograničenjima u obliku broja ploča. Model naplate korištenja aplikacije s pristupom svim funkcionalnostima zasniva se po članu projektnog tima, no cjenovno, *Trello* predstavlja pristupačnije rješenje za male timove nego *ClickUp* aplikacija. Nedostatak aplikacije *Trello* u okviru diplomskog rada jest promoviranje isključivo *Kanban* agilnog radnog okvira što je riješeno u aplikaciji koja je izrađena za potrebe diplomskog rada, gdje se potiče agilni radni okvir *scrum* putem pogleda preko lista umjesto *Kanban* ploče.



Slika 3.2. Prikaz korisničkog sučelja Trello mobilne aplikacije [20]

3.3. Mobilna aplikacija Asana

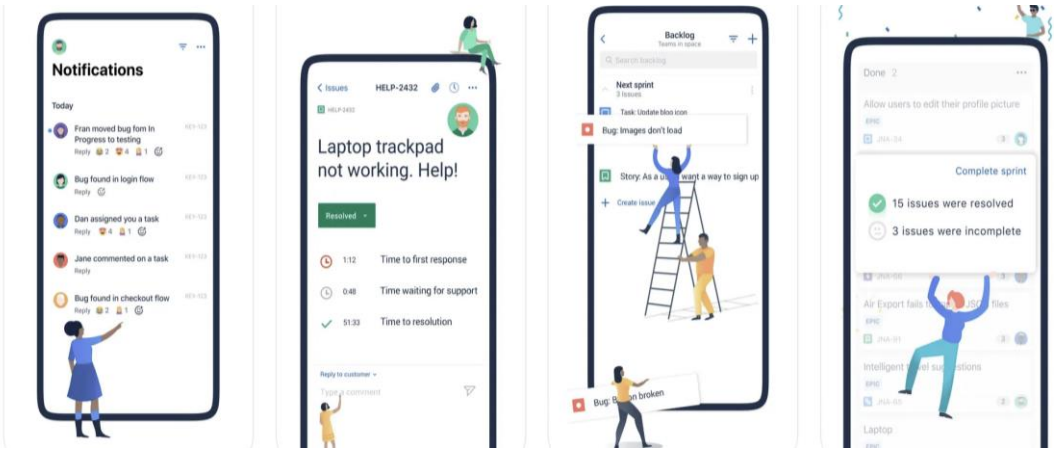
Asana predstavlja softversko rješenje za upravljanje osobnim zadacima kao i projektima. Pregled zadataka i organizaciju temelji se na prikazu liste zadataka što aplikaciju čini preglednom i intuitivnom za korištenje. Model naplate korištenja svih funkcionalnosti je poprilično sličan *ClickUp* aplikaciji, no korištenje *Asane* isključivo je ograničeno u mrežnom načinu. Izgled aplikacije na više zaslona vidljiv je na slici 3.3. Nedostatak aplikacije *Asana* jest nemogućnost povezivanja zadataka u besplatnoj inačici te je taj nedostatak riješen u aplikaciji izrađenoj za potrebe diplomskog rada, omogućavajući korisnicima povezivanje zadatka niže razine sa zadatkom jedne više razine.



Slika 3.3. Prikaz korisničkog sučelja Asana mobilne aplikacije [21]

3.4. Skup mobilnih aplikacija Jira

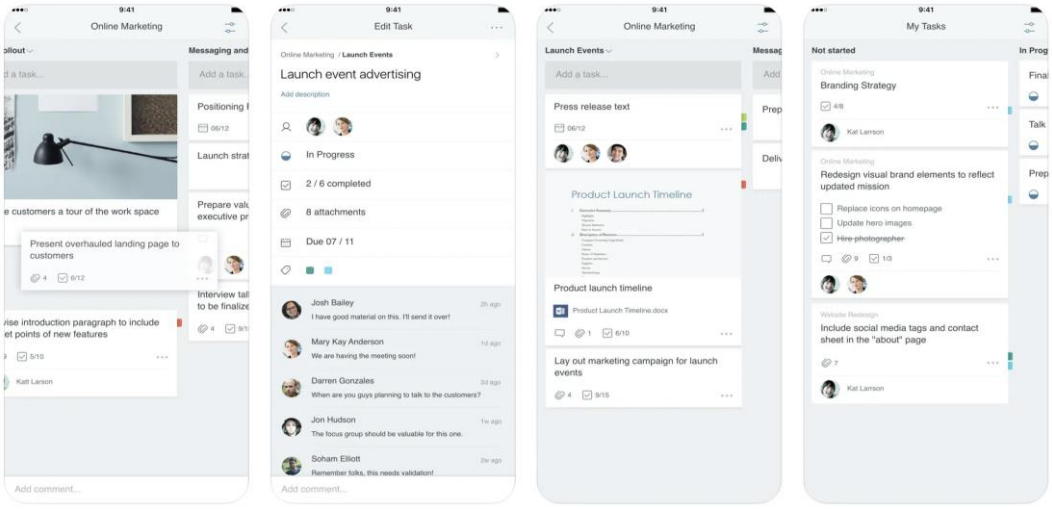
Jira predstavlja skup aplikacija za upravljanje isključivo softverskim projektima čiji je dizajn zaslona prikazan je na slici 3.4. Naslonjena je na moderne metodologije vođenja projekata, posebice *Scrum* okvir. *Jira* se isključivo koristi u korporacijske svrhe za razliku od prethodnih rješenja koji imaju mogućnosti vođenja osobnih zadataka te se iz tog razloga naplata izvršava ovisno o veličini tima ili poduzeća. Glavna značajka uz agilno vođenje projekta jest integracija s programima koje koriste stručnjaci za razvoj programskih rješenja. Nedostatak skupine mobilnih aplikacija *Jira* ističe se u vidu visoke razine znanja projektnog menadžmenta i metodologija potrebne za uspješno korištenje aplikacije. Navedeni nedostatak riješen je u aplikaciji izrađenoj za potrebe diplomskog rada na način izrade projekta, zadataka i *sprinteva* po točno određenim slijednim koracima.



Slika 3.4 Prikaz korisničkog sučelja Jira mobilne aplikacije [22]

3.5. Microsoft Planner

Microsoft Planner aplikacija je iz obitelji aplikacija *Office365* koja korporacijama nudi mogućnost upravljanja projektom u vizualno jednostavnom dizajnu u kojemu su moguće personalizacije izgleda upravljačke ploče i integracija s ostalim *Microsoft* aplikacijama. Mogućnosti aplikacije za računalo i mobilne nisu velike kao u slučaju aplikacija prethodno analiziranih. Jednostavni dizajn i specifični elementi za *Microsoft* aplikacije vidljivi su na slici 3.5. Nedostatak *Microsoft Planner* aplikacije jest nemogućnost odvajanja pogleda za voditelja i projektnog člana. Navedeni nedostatak je riješen u aplikaciji tako da projektni voditelj posjeduje uvid u sve zadatke, dok projektni član posjeduje uvid u samo njemu pridružene zadatke.



Slika 3.5. Prikaz korisničkog sučelja Microsoft Planner mobilne aplikacije [23]

4. PROGRAMSKO RJEŠENJE ZA UPRAVLJANJE IT PROJEKTIMA

Aplikacija za potrebe vođenja IT projekata proizlazi iz potrebe za olakšavanje i poboljšavanje upravljanja projektima posebice unutar malih IT timova koji prelaze iz evidentiranja projekta usmenim putem ili putem tablica prema programskom rješenju koje će timovima, posebice projektnim menadžerima ili osobama s autoritetom unutar organizacije, pružiti alat za učinkovito planiranje, praćenje i provedbu IT projekata, bez obzira na veličinu ili složenost. Također, inspiracija za stvaranje aplikacije dolazi iz vlastitog višegodišnjeg iskustva rada na projektima i vođenju istih te generalno kaotičnom načinu vođenja IT projekata unutar malih timova i organizacija kojima ostali napredni alati predstavljaju prekomplicirano rješenje. Organizirani sustav omogućiti će sveobuhvatno upravljanje IT projektima. Aplikacija je orijentirana prvenstveno na agilnu metodu vođenja te kao takva predstavlja besplatno rješenje sa zadanim, agilnim načinom vođenja.

Aplikacija izrađena u svrhu diplomskog rada ima posebne ciljeve rješavanja nekih od izazova s kojima se susreću mobilne aplikacije za upravljanje projektima koje su navedene u trećem poglavlju, a oni su:

- Jednostavnost korisničkog sučelja: funkcionalnost koja posebice pomaže projektnim voditeljima u malim organizacijama koji nisu primarno projektni voditelji te im je vrijeme prilagodbe na aplikaciju smanjeno,
- Intuitivnost: cilj koji se veže na jednostavnost korisničkog sučelja tako da radnje unutar aplikacije imaju logičan tijek i očekivan rezultat,
- Postavljanje hijerarhije zadataka: unutar agilnog načina vođenja vrlo bitna značajka jest postavljanje hijerarhijske zavisnosti zadataka koja u pojedinim alatima dostupnima na tržištu zna biti skrivena, teško dostupna ili nedostupna u besplatnom planu,
- Prikaz gantograma: gantogram predstavlja jedan od najboljih i najkorištenijih uvida u projektni tijek te se u pojedinim alatima ne nalazi u zadanim pogledima ili ga je potrebno ručno dodati ili se dodatno naplaćuje, a osnovna je vrsta informacije za voditelja projekta,
- Ograničenost značajki ovisno o plaćenom planu: alati dostupni na tržištu imaju veliku razliku dostupnih značajki ovisno o plaćenom planu gdje besplatne verzije većinom nemaju veliki dio funkcionalnosti u odnosu na plaćeni.

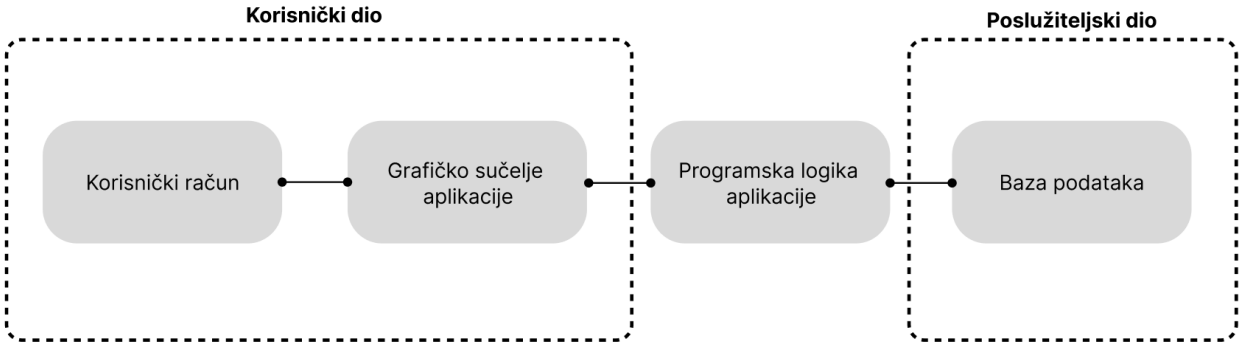
Aplikacija u odnosu na rješenja dostupna na tržištu koji su analizirani u trećem poglavlju diplomskog rada ne donosi cjelokupno inovativno programsko rješenje, ali svojim grafičkim

elementima i načinom rada promiče korištenje radnog okvira *Scrum* za vođenje IT projekata. Uz navedeno, aplikacija sadrži dodatne funkcije uvida u gantogram i postavljanje hijerarhijske ovisnosti zadataka agilnom nomenklaturom. Jednostavno korisničko sučelje i slijedno postavljanje projekta omogućuje korisnicima svih razina znanja vođenja projekata na efektivan način. Prednost aplikacije u odnosu na programska rješenja analizirana kroz treće poglavlje jest omogućavanje navedenih osnovnih i dodatnih funkcija besplatno uz mogućnost prilagođavanja rješenja za pojedino poduzeće korištenjem izvornog koda dostupnog u prilogu diplomskog rada.

Na dalje kroz poglavlje dano je idejno rješenje, opisana je izrada mobilne aplikacije te je dana analiza njenog rada.

4.1. Idejno programsko rješenje

Unutar ovog poglavlja predstavljena je arhitektura visoke razine (engl. *High level architecture, HLA*) i specifikacija programskog rješenja podijeljena na tehničke zahtjeve, strukturu aplikacije koja prikazuje korisničke uloge, funkcionalnosti i arhitekturu baze podataka. Poglavlje služi kao priprema za uspješnu realizaciju programskog rješenja.



Slika 4.1. Arhitektura visoke razine programskog rješenja

Arhitektura prikazana na slici 4.1. predstavlja najvažnije blokove sustava koji su preduvjet izrade programskog rješenja. Korisnički račun koji korisnik stvara, putem sučelja aplikacije stvara zahtjeve koji se obrađuju kroz programsku logiku prema bazi podataka. Odgovori od strane baze podataka obrađuju se kroz programsku logiku te se promjene ili informacije prikazuju na korisničkom sučelju. Programska logika aplikacije predstavlja blok koji komunicira s grafičkim sučeljem i bazom podataka. *HLA* arhitektura je podijeljena na korisnički i poslužiteljski dio.

4.1.1. Tehnički zahtjevi

Tehnički zahtjevi donose sistemske i funkcionalne potrebe kako bi se cilj mogao realizirati. U slučaju mobilne aplikacije za vođenje projekata tehnički zahtjevi su navedeni u tablici 4.1 u kojoj su navedeni razlozi odabira pojedinih tehnologija. Lijevi stupac tablice predstavlja opise tehničkog zahtjeva dok desni predstavlja razlog odabira navedenog tehničkog zahtjeva.

Tablica 4.1. Popis tehničkih zahtjeva programskog rješenja i odabir tehnologija

Opis tehničkog zahtjeva	**Razlog odabira**
Aplikacija će biti izrađena za platforme a) *iOS* od verzije 15 nadalje b) *Android* od verzije 8 nadalje	Odabirom dva najkorištenija operativna mobilna sustava i minimalno podržanim verzijama osigurava se veći korisnički doseg aplikacije kao i konkurentnost te se smanjuje mogućnost korištenja aplikacije na starijim verzijama operacijskih sustava koje smanjuju mogućnosti grešaka pošto se aplikacija testira u najnovije dostupnoj verziji.
Tehnologija kojima se realizira aplikacija za navedene platforme iznad: a) *Flutter* 2.x	*Flutter* komplet za razvoj softvera posjeduje sposobnost stvaranja prilagodljivih više platformskih mobilnih aplikacija korištenjem jednog koda. *Flutter* je oslonjen na programski jezik *Dart* i omogućava brz razvoj, intuitivno korisničko sučelje, opsežnu biblioteku alata i *widgeta* te ažurnu zajednicu programera. [24]
Aplikacija je namijenjena za korištenje na mobilnim uređajima, isključivo u portret orijentaciji zaslona.	Ograničavanje portret orijentacije smanjuje vrijeme razvijanja aplikacije i optimizaciju grafičkih elemenata za slučaj korištenja u pejzažnom načinu.
Za *back-end* i bazu koristiti će se usluga *Firebase*	*Firebase* pruža velik broj usluga prilagođene za razvoj u *Flutter* okruženju, a neke od njih su: provjera identiteta korisnika, baza u stvarnom vremenu, upravljanje korisnicima,

	obavijesti. Također, *Firebase* nudi besplatan prostor koji se može koristiti u testne i produkcijske svrhe. [25]
Upotreba *JSON* (engl. *JavaScript Object Notation, JSON*) struktura zahtjeva za komunikaciji između aplikacije i baze	*Firebase* je naslonjen na takav format razmjene podataka. *JSON* predstavlja jednostavan i čitljiv format kompatibilan s *Flutterom* i *Firebaseom* i pogodan za uporabu u *NO-SQL* bazama kao što je Firebase.
Aplikacija je zavisna o internetskoj vezi i radi isključivo u povezanom modu	Zbog mogućnosti aplikacije da omogući upravljanje i nadzor projekta u stvarnom vremenu bitno je ograničiti aplikaciju na rad isključivo u slučaju postojanja stabilne internetske povezanosti.
Izvorni kod aplikacije će biti objavljen na *github* profilu	Aplikacija će nakon razvoja biti objavljena na *github* profilu kao preduvjet za potencijalni prijenos aplikacije na aplikacijske servise u nadolazećem vremenu.

4.1.2. Funkcionalni zahtjevi

Poglavlje funkcionalnih zahtjeva predstavlja glavne značajke aplikacije čijom se izradom realizira sustav prikazan na slici 4.1. U ovom poglavlju navedene su funkcionalnosti aplikacije podijeljene po cjelinama (u slučaju ovog programskog rješenja, funkcionalnosti predstavljaju glavne značajke aplikacije). U nastavku poglavlja nalazi se tablica 4.2. koja sadrži popis funkcionalnosti aplikacije za vođenje IT projekata. Svakoj funkcionalnosti pridružen je identifikator i opis funkcionalnosti. Detaljniji opis funkcionalnosti opisan je u tablici 4.3. U nastavku poglavalja dan je uvid u način rada baze podataka koja je odabrana za potrebe izrade aplikacije.

Korisničke uloge

Programsko rješenje razlikuje dvije vrste uloga: ulogu voditelja projekta i ulogu člana. Uloga voditelja donosi potpunu kontrolu nad projektom i pristup svim funkcionalnostima aplikacije dok uloga člana je ograničena definicijom unutar pojedine funkcionalnosti.

Tablica 4.2. Funkcionalnosti programskog rješenja

Identifikator	Opis
F1	Aplikacija prikazuje početni zaslon
F2	Korisnik se prijavljuje putem e-pošte i zaporke
F3	Aplikacija prikazuje zaslon nadzorne ploče
F4	Korisnik izrađuje projekt i pridružuje članove
F5	Korisnik stvara zadatak
F6	Korisnik stvara sprint
F7	Aplikacija pohranjuje podatke u mrežnu bazu podataka
F8	Aplikacija dohvaća podatke iz baze podataka
F9	Aplikacija prikazuje uvid u zadatke
F10	Korisnik uređuje detalje zadatka
F11	Korisnik može dohvatiti vizualni prikaz gantograma

Tablica 4.3. Detaljan opis funkcionalnosti programskog rješenja

Identifikator	Detaljan opis funkcionalnosti
F1	Prilikom pokretanja aplikacije (u slučaju prvog pokretanja ili nakon odjave) korisniku se prikazuje zaslon s mogućnošću prijave ili registracije.
F2	Korisnik se može registrirati ili logirati putem kombinacije e-mail adrese i zaporke. Registracija zahtjeva unos imena, adrese e-pošte i zaporke.
F3	Nadzorna ploča ovisno o ulozi unutar aplikacije ima različit izgled. Za voditelja projekta prikazan je trenutni sprint i popis svih zadataka. Za člana projekta prikazan je trenutni sprint i popis zadataka koji su mu pridruženi. Nadzorna ploča služi i za evidenciju rada na zadacima, stoga član može mijenjati status te unositi vrijeme provedeno na zadatku. Svaki zadatak u popisu sadrži: status, naziv zadatka, opis zadatka, provedeno vrijeme na zadatku i prioritet. Nadzorna ploča omogućava evidentiranje radnog vremena i promjenu statusa. Statusi zadatka mogu biti: otvoreno, aktivno i zatvoreno
F4	Izrada, prikaz detalja i odabir projekta jest zaslon koji omogućava: izradu projekta, tj. njegovog naziva, dodavanje članova u projekt i promjenu projekta na kojem se korisnik nalazi. Izradom projekta korisnik stječe ulogu vlasnika projekta na istom projektu te projekt ne može imati više od jednog

	vlasnika projekta. Svi dodani članovi u projekt automatski poprimaju ulogu člana na projektu. Svaki korisnik aplikacije može izraditi više projekata i biti član na više projekata, no obje uloge ne može posjedovati na istom projektu.
F5	Zaslon koji prikazuje sljedeće opcije: unos naziva zadatka, odabir početnog i krajnjeg datuma zadatka, odabir prioriteta, odabir tipa zadatka, mogućnost hijerarhijskog povezivanja zadatka i postavljanje odgovorne osobe. Prilikom povezivanja zadataka moguće je isključivo povezati zadatak niže u hijerarhiji sa zadatkom koji je jednog više stupnja u hijerarhiji.
F6	Pri upravljanju sprintom, isključivo je moguće odabrati korisničke priče. Zaslon nudi mogućnosti: odabira trajanja sprinta, postavljanja zadataka u *sprint* i stvaranje *sprinta.*
F7	Način pohrane podataka i struktura baze podataka prikazana je u nastavku poglavlja.
F8	Način dohvaćanja podataka i struktura baze podataka prikazana je u nastavku poglavlja.
F9	Popis zadataka u kojem je dostupan prikaz svih stvorenih tipova zadataka (inicijativa, *epic*, korisnička priča). Zaslon nudi mogućnosti uređivanja detalja i brisanje zadatka.
F10	Korisniku je omogućeno uređivanje detalja zadatka ovisno o razini uloge. Članovi isključivo mogu uređivati detalje o provedenom vremenu na zadatku i status dok voditelji mogu uređivati sve informacije.
F11	Projektnom voditelju omogućeno je generiranje tijeka zadatka u kalendarskom pregledu u kojem su vidljive inicijative i *epici* koji su definirani u projektu.

Osnovna struktura i način rada baze podataka

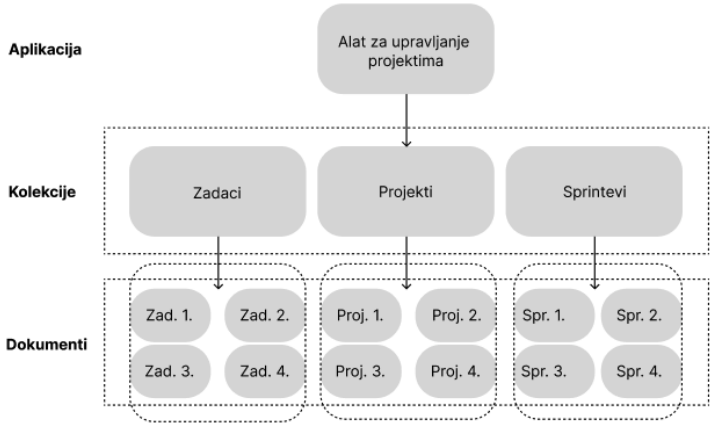
Za potrebe razvoja baze podataka koristi se *Firebase* baza podataka u stvarnom vremenu u kojoj je struktura podataka organizirana u obliku *JSON* stabla u kojemu svaki čvor predstavlja objekt, a čvorovi se mogu ugnijezditi jedan unutar drugoga kako bi stvorili složeniji odnos podataka i omogućili predstavljanje podataka u hijerarhijskom formatu. Glavni dijelovi *Firebase* baze podataka su kolekcije i dokumenti. Komunikacija s bazom podataka je obostrana što omogućuje slanje i primanje podataka u bazu i iz baze podataka.

JSON format za razmjenu podataka

JSON predstavlja otvoreni standard dizajniran za čitljivu razmjenu podataka ekstenzije .json. U okviru *JSON* strukture mogu se koristiti sljedeći tipovi podataka: broj, *string, boolean,* niz, objekt, *null. JSON* struktura može biti organiziran u vidu parova (ključ i vrijednost) ili uređenog niza vrijednosti. Struktura na jednostavnom primjeru osobe Hrvoje Horvat s OIB brojem i jezicima kojima se služi prikazana je na slici 4.2. [26]

```
{
        "ime" : "Hrvoje",
        "prezime" : "Horvat",
        "OIB" : 12345678901,
        "jezici" : [ "Hrvatski", "Engleski" ],
}
```

Slika 4.2. Prikaz JSON strukture na primjeru objekta osobe



Slika 4.3. Prikaz klasične arhitekture Firebase baze podataka u stvarnom vremenu

Svaki od dokumenta prikazan na slici 4.3. sadrži *JSON* strukturu podataka. Primjerice dokument naziva Zad. 1. prikazan u hijerarhiji na slici 4.3. u sebi sadrži podatke o detaljima zadatka kao što su prioritet, opis, status itd. Detaljniji prikaz kolekcija i dokumenata sa svojim sastavnicama nalazi se u poglavlju izrade programskog rješenja. Ovakva vrsta baze podataka u opcijama sadrži gotov alat za provjeru identiteta korisnika koji omogućuje jednostavnu implementaciju rješenja za prijavu i registraciju korisnika. Komunikacija između aplikacije i servera odvija se u sljedećih šest koraka:

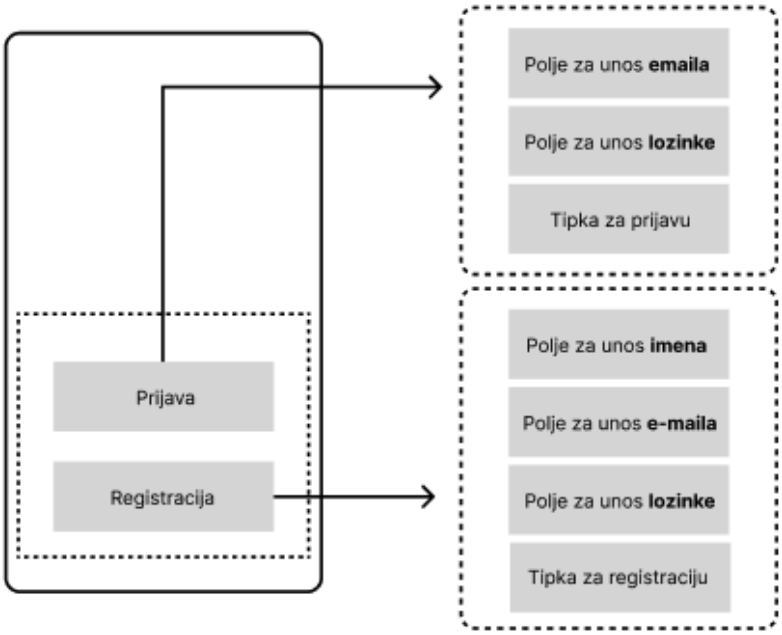
1. Korisnik se želi prijaviti ili registrirati
2. Aplikacija poziva *Firebase* autentikacijski komplet za razvoj softvera
3. *Firebase* server provjerava identitet
4. *Firebase* generira *JWT* token koji sadrži informacije o korisniku i digitalni potpis za provjeru identiteta,

5. *JWT* token se šalje aplikaciji
6. Korisnik ima pristup resursima i funkcionalnostima uz korištenje tokena za provjeru autentičnosti prilikom svakog zahtjeva za pristup resursima. [27]

4.1.3. Korisničko sučelje aplikacije

Poglavlje korisničkog sučelja aplikacije sadrži prototipne dizajne zaslona i radni tijek koji služe za izradu programskog rješenja i postizanje funkcionalnosti raspisanih u poglavlju 4.1.2. Prototipni dizajn predstavlja popis važnih funkcionalnih blokove po zaslonu te ne predstavlja krajnji izgled aplikacije i pozicijski raspored blokova na zaslonu. Poglavlje je raspoređeno po zaslonima koji postižu funkcionalnosti raspisane u poglavlju 4.1.2.

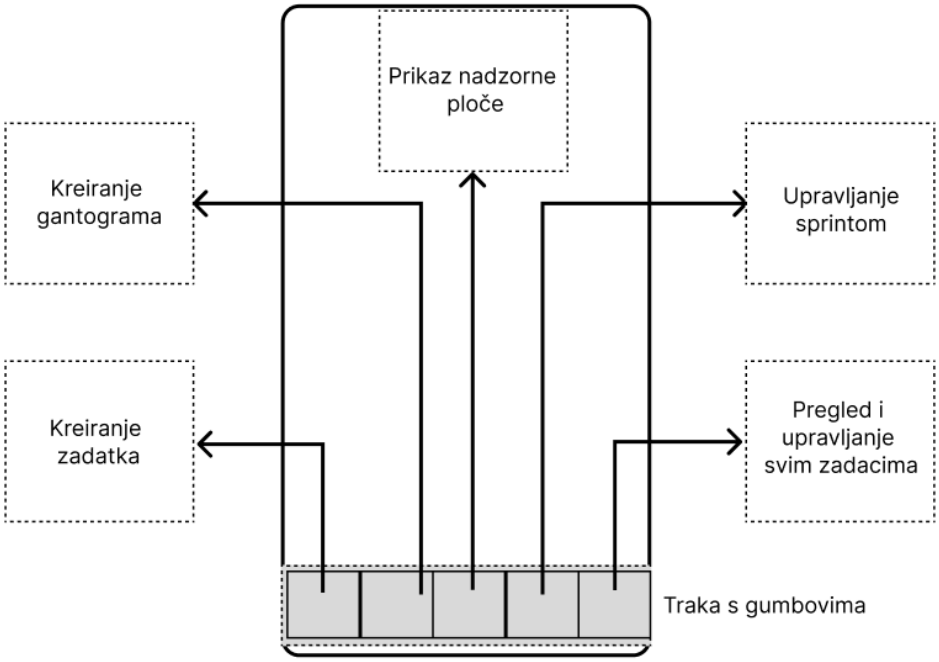
Provjera korisničkog identiteta



Slika 4.4. Prototipni dizajn korisničkog sučelja provjere korisničkog identiteta

Korisničko sučelje provjere korisničkog identiteta predstavlja prvi zaslon u programskom rješenju na kojemu se nalaze dvije tipke pomoću kojih se korisnik prijavljuje ili registrira. Ukoliko se korisnik prijavljuje pojavljuju mu se dva polja za unos e-pošte i zaporke te gumb za prijavu. Ukoliko se radi o registraciji pojavljuju se unosi za ime, e-poštu i zaporku te gumb za registraciju. Primjer prototipnog dizajna korisničkog sučelja provjere korisničkog identiteta prikazan je na slici 4.4.

Pristup projektnim funkcijama

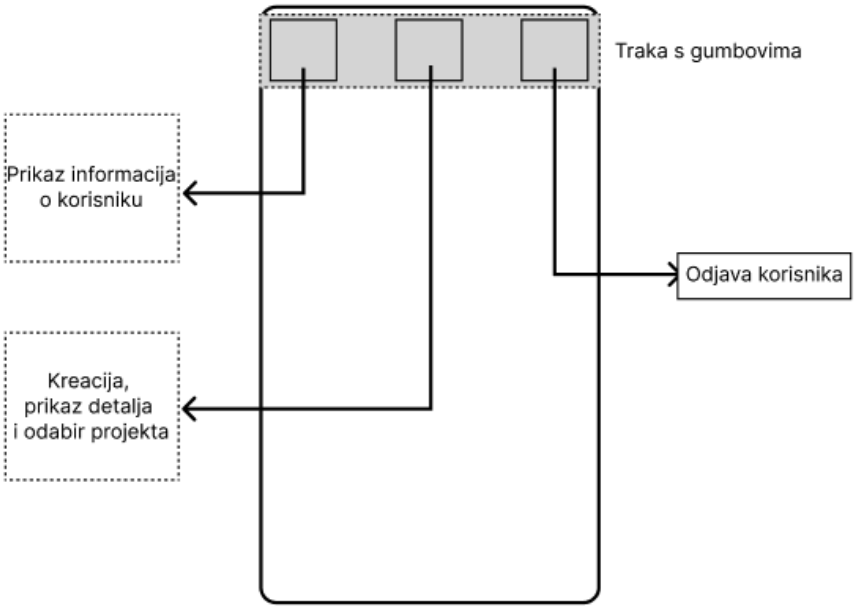


Slika 4.5. Prototipni dizajn pristupa projektnim funkcijama

Traka s gumbovima u podnožju zaslona mobilnog uređaja omogućava brzi pristup funkcionalnostima: stvaranja gantograma, izrade zadatka, prikaza nadzorne ploče, upravljanja sprintom i pregledu svih raspisanih zadataka. Prikaz zaslona ostvaruje se pritiskom na gumb koji je povezan sa željenim zaslonom. Prototipni dizajn korisničkog sučelja pristupa projektnim funkcijama prikazan je na slici 4.5.

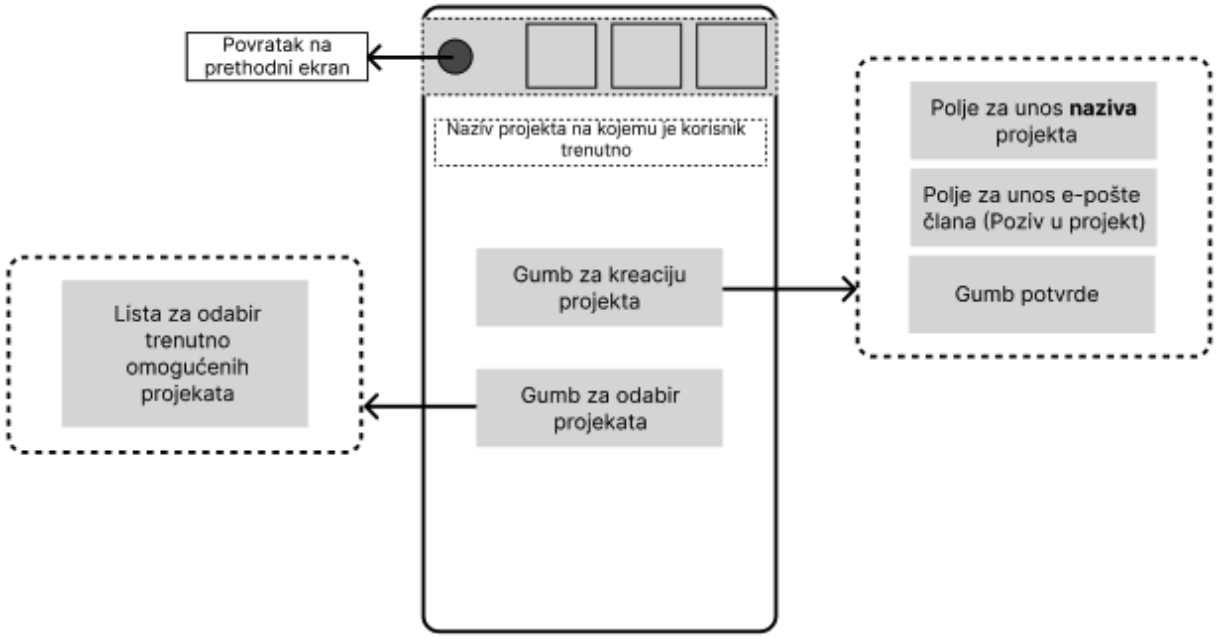
Pristup projektnim informacijama i korisničkim funkcijama

Traka s gumbovima pri vrhu zaslona mobilnog uređaj omogućava brzi pristup funkcionalnostima prikaza informacija o korisniku, izradi, prikazu detalja i odabiru projekta te odjavi korisnika iz sustava. Prikaz zaslona ostvaruje se pritiskom na gumb koji je povezan sa željenim zaslonom ili funkcijom. Prototipni dizajn korisničkog sučelja koji omogućava pristup projektnim informacijama i korisničkim funkcijama prikazan je na slici 4.6.



Slika 4.6. Prototipni dizajn pristupa projektnim informacijama i korisničkim funkcijama

Izrada, prikaz detalja i odabir projekta

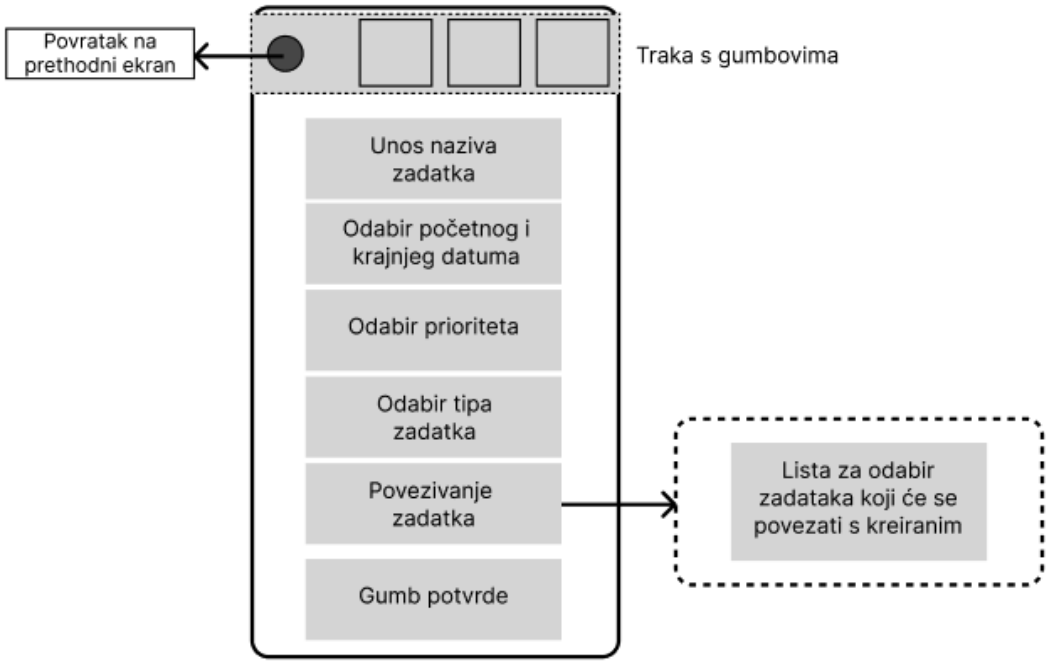


Slika 4.7. Prototipni dizajn korisničkog sučelja za izradu, prikaz detalja i odabir projekta

Na zaslonu koji sadrži funkcionalnosti izrade, prikaza detalja i odabira projekta nalaze se gumbovi za izradu projekta i odabir projekta. Pritiskom na izradu projekta prikazuje se mogućnost unosa naziva projekta, poziv članova i potvrdu kada je voditelj projekta završio s procesom. Gumb za odabir projekata otvara popis trenutno omogućenih projekata na kojima je trenutni korisnik vlasnik ili član. Pritiskom na jedan od projekata iz navedenog popisa korisnik mijenja projekt na kojem se

nalazi. Naziv projekta na kojem se nalazi vidljiv je u gornjem dijelu zaslona ispod gornje trake. U gornjoj traci na zaslonu sa slike 4.7. pojavljuje se gumb za povratak na prošli prethodni zaslon.

Izrada zadatka

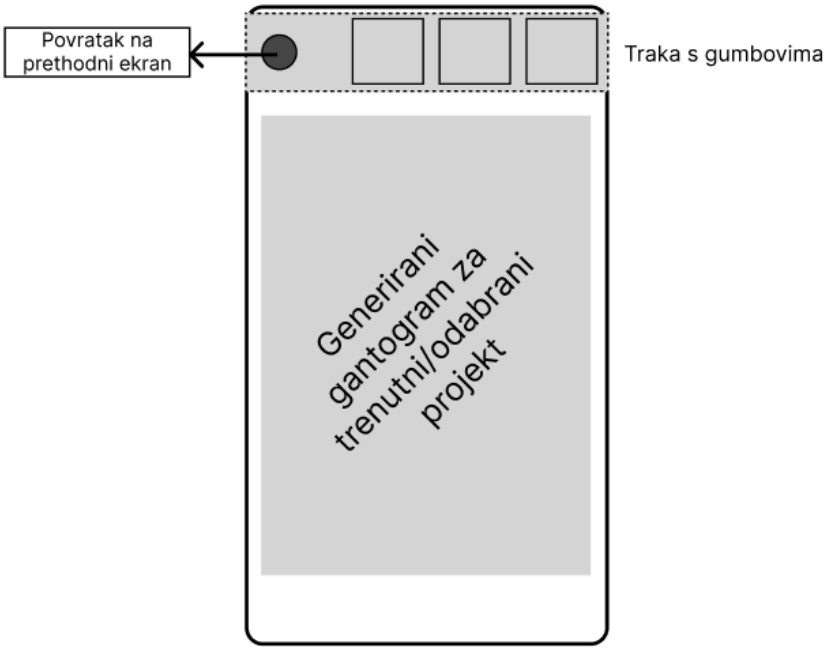


Slika 4.8. Prototip dizajna korisničkog sučelja za stvaranje zadatka

Pristup zaslonu za stvaranje zadatka omogućen je iz trake s gumbovima u podnožju zaslona za voditelje projekta. Zaslon za stvaranje zadatka sastoji se od polja za unos naziva zadatka, odabira početnog i krajnjeg datuma, odabir prioriteta, odabir tipa zadatka i gumba za povezivanje zadatka koji otvara popis hijerarhijski većih tipova zadataka. Gumb potvrde omogućuje stvaranje zadatka s upisanim i odabranim informacijama iznad. Prototip dizajna korisničkog sučelja za stvaranje zadatka prikazan je na slici 4.8.

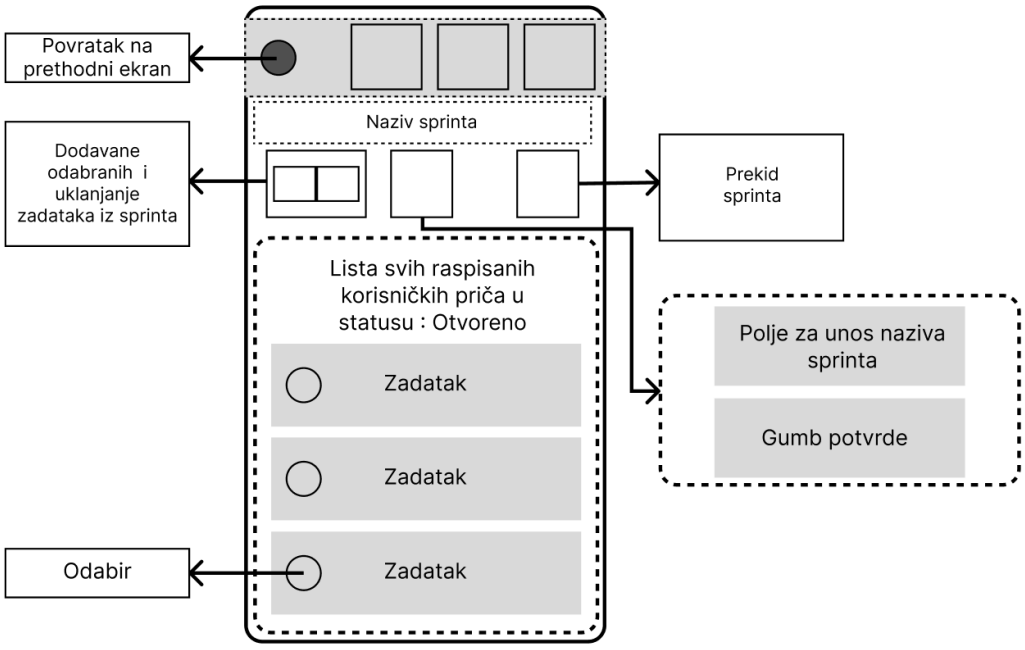
Stvaranje gantograma

Generirani gantogram omogućen je iz trake s gumbovima u podnožju zaslona za voditelje projekta. Zaslon na kojem se nalazi generirani gantogram sastoji se od kalendarske strukture u kojoj su posloženi zadaci po vremenskoj lenti. Prototip dizajna korisničkog sučelja za generirani gantogram prikazan je na slici 4.9.



Slika 4.9. Prototip dizajna korisničkog sučelja za generirani gantogram

Upravljanje sprintom

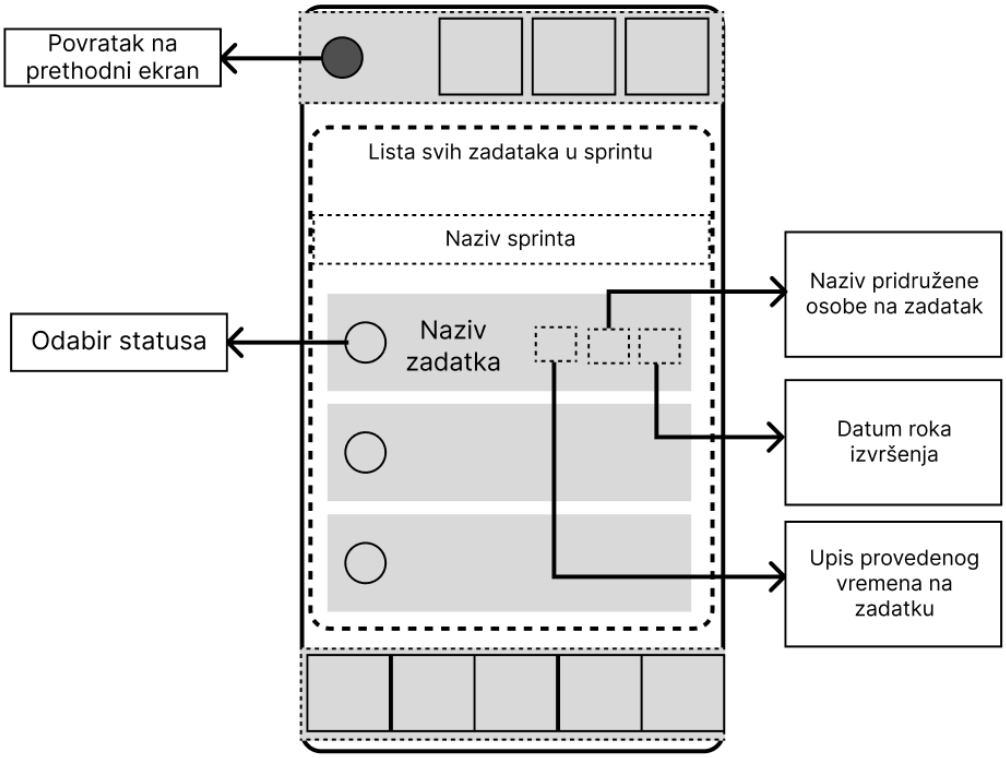


Slika 4.10. Prototip dizajna korisničkog sučelja za upravljanje sprintom

Pristup zaslonu na kojemu se nalaze kontrole za upravljanje sprintom omogućen je iz trake u podnožju zaslona na nadzornoj ploči voditelja projekta. Na ovom zaslonu voditelj projekta može odabrati zadatke koje svrstava u *sprint.* U funkcijama za upravljanje *sprintom* voditelj projekta može dodavati i uklanjati odabrane zadatke iz *sprinta,* upravljati nazivom putem gumba namijenjenog za to i prekinuti *sprint.* Popis svih korisničkih priča u aktivnom sprintu vidljiv je na

nadzornoj ploči voditelja projekta. Prekid *sprinta* omogućava stvaranje novog sprinta i odabir zadataka iz popisa svih raspisanih korisničkih priča koje su u otvorenom stanju. Prototip dizajna korisničkog sučelja za upravljanje *sprintom* prikazan je na slici 4.10.

Nadzorna ploča



Slika 4.11. Prototip dizajna korisničkog sučelja nadzorne ploče

Nadzorna ploča predstavlja glavni zaslon u aplikaciji koji ima različit izgled ovisno o ulozi. Voditelj projekta vide traku s gumbovima u podnožju zaslona te vidi sve zadatke pridružene u aktivni *sprint.* Članovi na projektu vide isključivo zadatke na koje su pridruženi te imaju mogućnost mijenjanja statusa zadatka i upis radnih sati na zadatku. Prototip dizajna korisničkog sučelja nadzorne ploče prikazan je na slici 4.11.

Pregled i uređivanje svih zadataka svih tipova

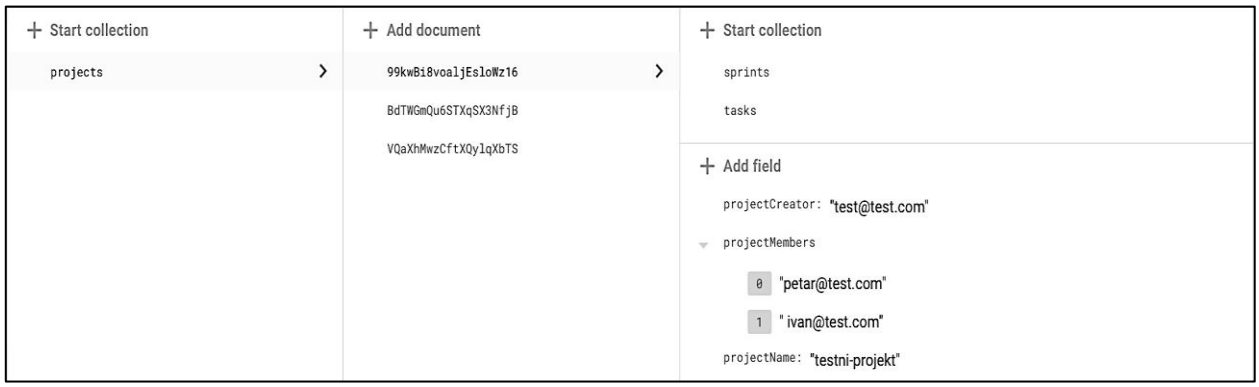
Zaslon pregleda i uređivanja svih zadataka svih tipova dizajnom je gotovo identičan nadzornoj ploči, samo što u ovom slučaju umjesto *sprinta*, prikazuju se svi zadaci te s pritiskom na zadatak moguće je dobiti više detalja o zadatku čiji se podaci mogu uređivati. Ovaj zaslon također nudi mogućnost brisanja zadatka iz baze podataka.

4.2. Izrada programskog rješenja

Poglavljem izrade programskog rješenja dan je uvid u razvoj mobilne aplikacije za vođenje IT projekata. U poglavlju je prikazano: struktura baze podataka korištena u razvoju, programska arhitektura, način na koji su implementirane funkcionalnosti programskog rješenja i validacija korisničkih zahtjeva kroz slučajeve korištenja.

4.2.1. Baza podataka

Način rada *Firebase* baze podataka korištene pri izradi programskog rješenja detaljno je prikazano u poglavlju 4.1.2. Ovim poglavljem dan je uvid u strukturu baze podataka koja se koristi u aplikaciji za vođenje IT projekata. Baza podataka podijeljena je u sljedeće kolekcije: projekti, zadaci i *sprintevi*. U hijerarhijskoj strukturi projekt predstavlja roditeljski čvor koji u svojim pod čvorovima sadrži zadatke i *sprinteve*. U slučaju hijerarhije koja je korištena za izradu programskog rješenja, kolekcija projekt sadrži dokument koji unutar sebe sadrži dvije kolekcije: *sprintevi* i zadaci. Kolekcije i njihova organizacija prikazana je na slici 4.12. Svaki dokument projekta sastoji se od informacija o: stvoritelju projekta, projektnim članovima i nazivu projekta.



Slika 4.12. Prikaz strukture baze podataka iz Firebase upravljačke ploče

Slika 4.13. Prikaz kolekcije sprintevi iz Firebase upravljačke ploče

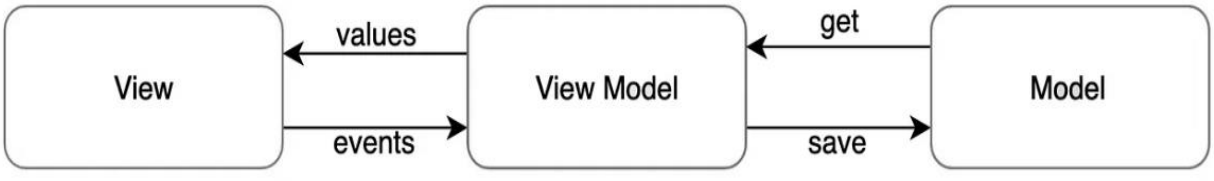
Kolekcija naziva sprintevi i njeni dokumenti koji predstavljaju pojedini stvoreni sprint u projektu sadrži informacije o početnom datumu, krajnjem datumu i identifikatoru sprinta. Kolekcija naziva sprintevi i njezini podaci vidljivi su na slici 4.13.

Slika 4.14. Prikaz kolekcije zadaci iz Firebase upravljačke ploče

Kolekcija naziva zadaci i njeni dokumenti koji predstavljaju pojedini zadatak stvoren u projektu sadrži informacije o: pridruženom korisničkom računu u obliku e-pošte, početni i krajnji datum zadatka, povezani zadatak više razine, sprint u kojem se nalazi, identifikator zadatka, naziv zadatka, opis, status, tip zadatka, prioritet i provedeno vrijeme na zadatku. Kolekcija naziva zadaci i njezini podaci vidljivi su na slici 4.14.

4.2.2. Programska arhitektura

Programska arhitektura aplikacije temelji se na Model-Pogled-Pogledni model arhitekturnom obrascu (engl. *Model-View-ViewModel, MVVM*). Razlog za korištenje *MVMM* ideje jest bolja struktura i snalaženje unutar koda uz poboljšane mogućnosti testiranja. Glavna zamisao *MVVM* arhitekture jest odvajanje poslovne logike od prezentacijskog sloja i razdvajanje odgovornosti između različitih komponenti vidljive na slici 4.12. *View* dio predstavlja korisničko sučelje, *View Model* predstavlja medijatora koji zahtjeva podatke od *Modela* i šalje podatke u *Model* te pruža podatke prema *View* dijelu. [28]



Slika 4.15. Općeniti prikaz MVVM arhitekture [28]

Implementacija *MVVM* arhitekture izvršena je putem obrasca pružatelja usluga (engl. *Provider*) unutar *Flutter* programskog okruženja. Paket naziva *Provider* koristi se za upravljanje stanjima i dijeljenjem podataka unutar aplikacije. *Provider* omogućuje dijeljenje i mijenjanje podataka kroz različite *widgete* aplikacije. *Widgeti* predstavljaju glavne komponente *Flutter* aplikacije. *Provider* uklanja potrebu ponovnim rekonstruiranjem aplikacije pri promjeni podataka tako što omogućuje komponentama korisničkog sučelja osvježavanje. [28]

Slika 4.16. Specifična programska struktura programskog rješenja

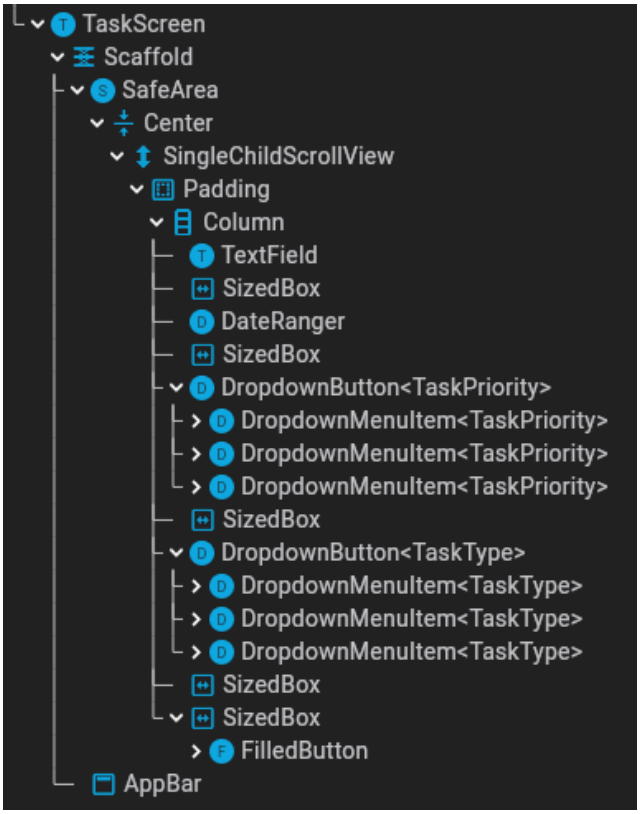
U programskom rješenju korišten je *Provider* kao *ViewModel* koji komunicira s bazom podataka, tj. sprema sve promjene vezane za zaslon na kojem se nalazi i na određene definirane akcije šalje informacije u bazu podataka. U primjeru prikazanom na slici 4.16. prikazana je modificirana *MVVM* struktura u kojoj *Provider* na zaslonu popisa projekta novi projekt izrađuje u bazi tek nakon što su izrađeni svi elementi projekta (osnovne informacije o projektu, zadatak i sprint). Princip toka podataka isti je za ostale zaslone. *Provider* omogućava spremanje podataka u *ViewModel* kroz više zaslona i slanje informacije prema krajnjem modelu koji je baza podataka. U povratnoj vezi prema zaslonu, elementi zaslona, točnije *widgeti* izvršavaju komunikaciju s bazom podataka i pomoću rade promjene na zaslonu.

4.2.3. Izrada i implementacija funkcionalnosti programskog rješenja

Ovim poglavljem dan je uvid u način na koji su izrađeni najvažniji segmenti aplikacije od grafičkih elemenata korisničkog sučelja, uloga i funkcionalnosti. Potpuni izvorni kod i pristup aplikaciji omogućen je pomoću poveznice u prilogu diplomskog rada.

Izrada i implementacija elemenata korisničkog sučelja

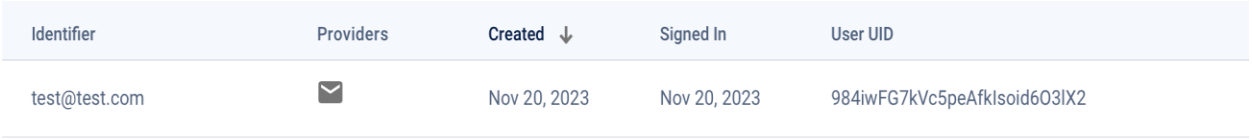
Prototipni dizajni grafičkih elemenata izrađeni su pomoću programa *Figma* koji omogućuje brzo vizualno prototipiranje. Zbog nemogućnosti izravnog kopiranja elementa iz *Figme* unutar *Fluttera* svaki element bilo je potrebno rekonstruirati kroz kod. Svaka vidljiva komponenta na grafičkom sučelju aplikacije predstavlja *widget* dok sveukupni zaslon predstavlja *widget* sastavljen od više *widgeta.* Svi *widgeti* korišteni u aplikaciji pripadaju skupini zadanih *Flutter widgeta* sadržani u paketu *material.dart.* Na slici 4.17. vidljivi su elementi od kojih je sadržan zaslon za izradu zadataka kao što su *SafeArea, Padding, Column, SizedBox, SingleChildScrollView, TextField i DropdownButton.* Prva četiri navedena *widgeta* predstavljaju elemente kojima se postiže vizualni raspored elemenata na zaslonu. Ostali navedeni *widgeti* predstavljaju elemente višestrukog odabira, upisa teksta i prikaza popisa. Cjelokupna aplikacija sastoji se od jedanaest zaslona.



Slika 4.17. Struktura grafičkih elemenata na zaslonu za izradu zadatka

Provjera korisničkog identiteta

Za provjeru korisničkog identiteta korišten je *Flutter* paket *firebase_auth.dart* koji pruža klase i metode za spajanje na uslugu *Firebase Authentication* koja omogućuje prijavu i registraciju u projekt. Svi korisnici spremaju se na oblak unutar baze korisnika. Metoda iz klase *FirebaseAuth* korištena za prijavu i registraciju unutar aplikacije je *signInWithEmailAndPassword* koja kao argumente prima e-poštu i zaporku kao i metoda *createUserWithEmailAndPassword* koja služi za registraciju korisnika. Svaki uspješno registrirani korisnik vidljiv je unutar *Authentication* sučelja na *Firebase* servisu. Na slici 4.18. prikazano je *Authentication* sučelje u kojemu su vidljivi korisnici i informacije kojima se može pristupiti kao što su: e-pošta korisničkog računa, datum stvaranja korisničkog računa, zadnja prijava i korisnički identifikator.



Slika 4.18. Prikaz korisnika unutar Authentication sučelja na Firebase servisu

Stvaranje i prikaz projekata

Programska logika za stvaranje projekta i prikaz projekata sadržana je na dva zaslona u kojemu se događa konstantna komunikacija s bazom podataka. Komunikacija je omogućena pomoću paketa *cloud_firestore.dart.* Prikaz projekta je zaslon koji se u redu izvođenja nalazi ispred zaslona izrade projekta. Prikaz projekata dostupnih u bazi omogućen je putem toka (engl. *Stream)* koji predstavlja asinkroni način dohvaćanja podataka. Svaki puta kada tok završi s provjerom javlja slušateljima te informacije (*widgetima*) kako je završio sa slanjem svih događaja iz baze podataka nakon čega se izvršava osvježavanje korisničkog sučelja ukoliko postoje promjene. Izrada projekta i sva logika spremanja podataka promijenjenih na zaslonu u bazu podataka izvršena je pomoću *providera* sadržanog u datoteci pm_alat_model.dart koji u ovom slučaju sprema sve promijenjene podatke na projektu lokalno te na pritisak tipke spremanja projekta sprema podatke u kolekciju projekti na oblak.

Implementacija uloga

Dodjela uloga unutar aplikacije izvršava se automatizmom. Korisnički račun koji je stvorio projekt upisan je u bazu podataka, točnije, dokument projekta iz kolekcije projekti u čiji je ključ naziva *projectCreator* upisana informacija e-pošte korisničkog računa. Stvoritelj projekta na zaslonu stvaranja projekta definira korisničke račune (njihove e-pošte) koji će na tom projektu posjedovati

ulogu člana. Članovi su upisani u dokument projekta iz kolekcije projektu u čiji je ključ naziva *projectMembers* upisana informacija e-pošte korisničkih računa članova. Svi upisi informacija izvršeni su pomoću *providera.*

Prikaz, izrada i uređivanje zadataka

Programska logika iza tri mogućnosti upravljanja zadacima raspoređena je na četiri zaslona. Zasloni koji imaju veze sa zadacima su:

- nadzorna ploča i zaslon s popisom zadataka pri izradi projekta i pri upravljanju projektom koji prikazuje zadatke iz baze podataka za određeni projekt pomoću *ListView.builder widgeta,*
- izrada i evidentiranje rada na zadacima koji se može okarakterizirati kao uređivanje informacija zadatka predstavljaju zaslone koji se slažu u stog te se prilikom povratka na prethodni zaslon funkcijom *pop* izbacuju iz stoga. Prije izbacivanja iz stoga, baza podataka se puni podacima za zadatak pomoću *providera.*

Prikaz, izrada i uređivanje sprinteva

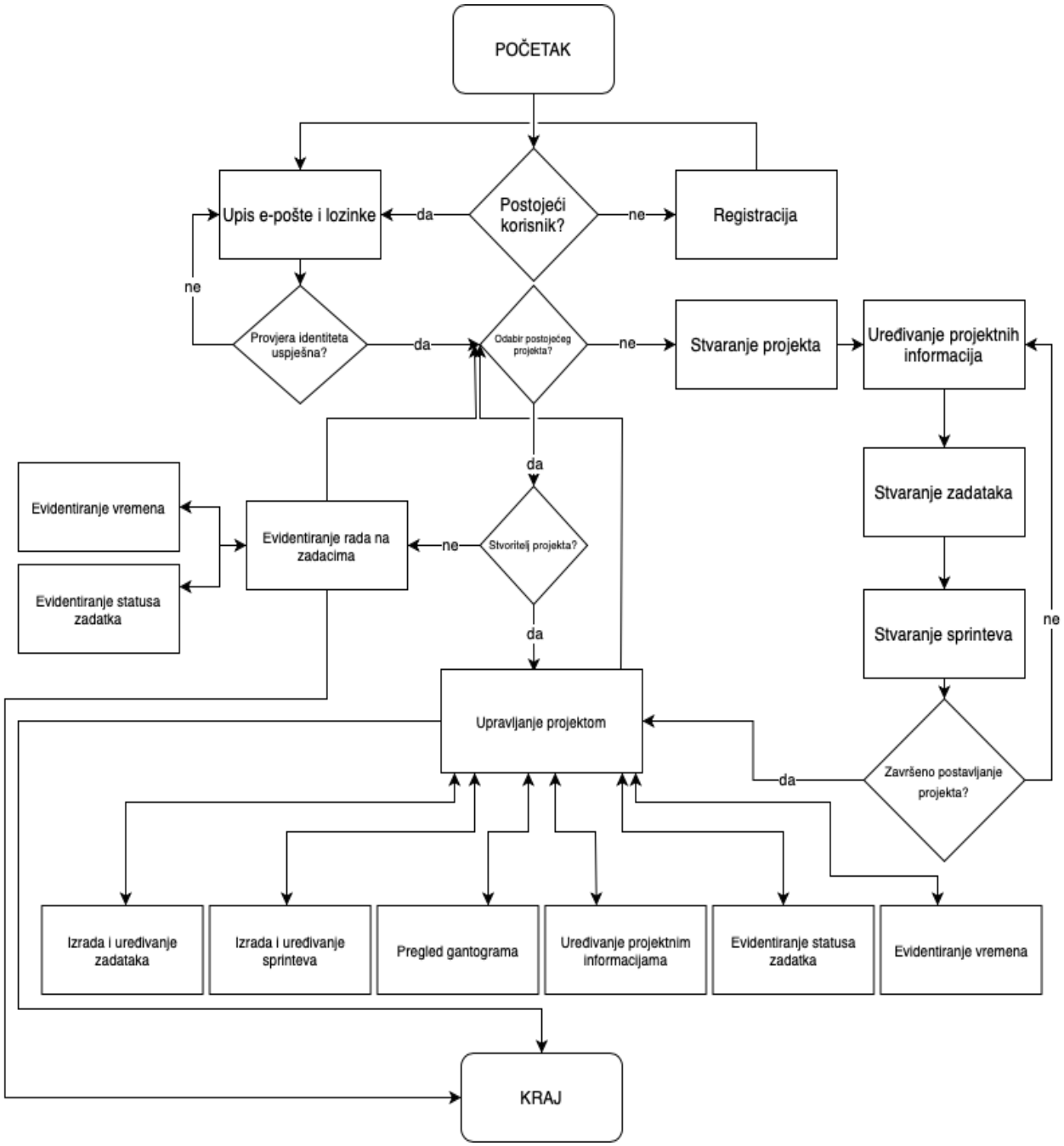
Programska logika za manipuliranje sprintevima identična je kao i programska logika za zadatke. Prilikom izrade sprinta, zaslon izrade slaže se u stog te se nakon izvršene akcije putem *providera* informacije u bazi podataka osvježavaju i zaslon za izradu sprinta se uklanja iz stoga putem funkcije *pop* koja kao argument prima trenutno aktivan zaslon. Izrada sprinta ovisi o postojanosti zadataka tako što pri izradi sprinta mogućnost ulaska u sprint ima korisnička priča. Odabir se izvršava putem *CheckboxListTile widgeta* koji omogućuje višestruki odabir korisničkih priča u popisu.

Izrada gantograma

Prikaz gantograma riješen je pomoću postojećeg paketa dostupnog na repozitoriju *Flutter* i *Dart* paketa. Paket koji je korišten naziva je *dynamic_timeline.dart* koji pruža jednostavno prilagođavanje vizualnog dojma putem promjene svojstava klase *DynamicTimeline.* Dodatna programska logika korištena u slučaju izrade gantograma jest korištenje *providera* prilikom svakog pokretanja gantograma koji pruža informacije o promjenama koje se rezultiraju kao osvježenje vizualnog prikaza. Gantogramom je isključivo omogućen pregled inicijativa i *epica* bez mogućnosti prikaza kraćih zadataka.

4.3. Analiza programskog rješenja

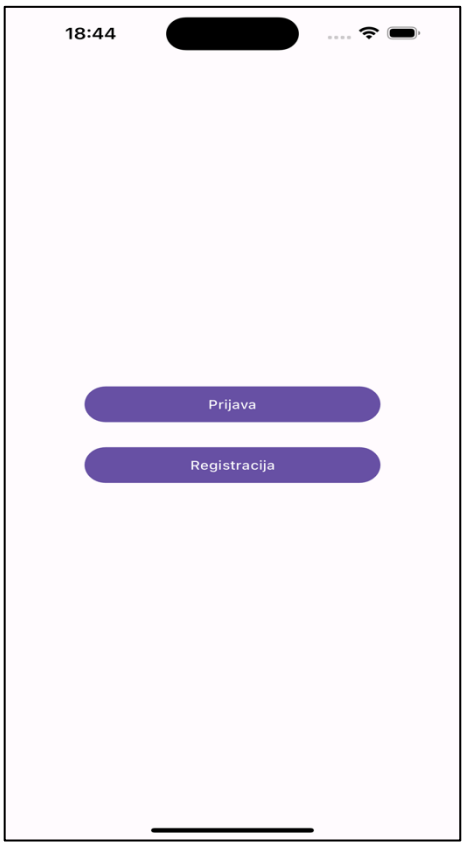
U poglavlju analize programskog rješenja dan je uvid u način rada aplikacije kroz dijagram tijeka aplikacije, korisnički doživljaj koji predstavlja krajnje vizualno rješenje i validaciju aplikacije kroz više općenitih scenarija korištenja. Dijagram tijeka aplikacije za vođenje IT projekata prikazan je na slici 4.19.



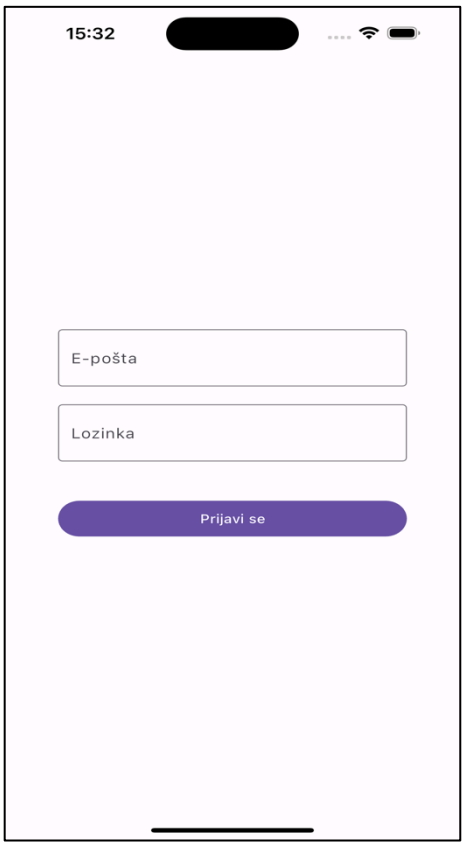
Slika 4.19. Dijagram tijeka aplikacije za vođenje IT projekata

4.3.1. Korisnički doživljaj

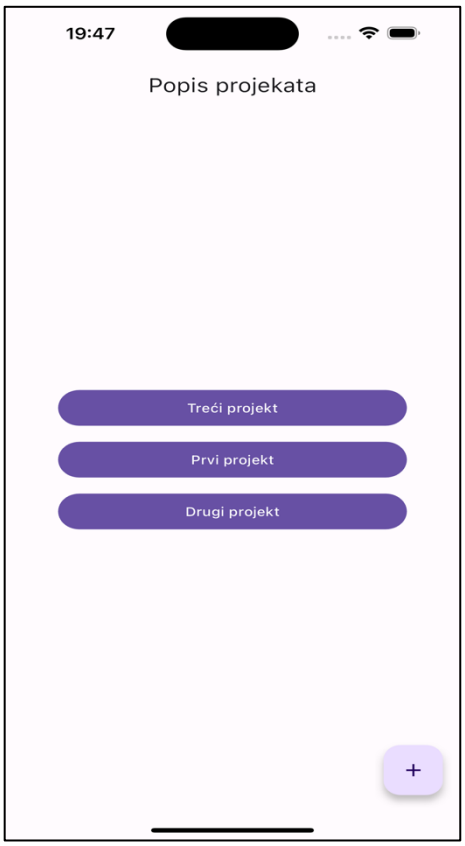
Korisnički doživljaj predstavlja finalizirani grafički prikaz programskog rješenja u koji krajnji korisnik ima uvid. U nastavku, na slikama (od 4.20. do 4.33.) dan je uvid u sve zaslone koje posjeduje aplikacija za upravljanje IT projektima. Zasloni nisu identični prototipnim rješenjima iz poglavlja 4.1.3. iz razloga evoluiranja aplikacije kroz fazu izrade u kojoj su uočeni tehnički i vizualni nedostaci prototipnih prijedloga. Zasloni prikazani u nastavku poglavlja snimljeni su u emulatoru *Flutter* programskog rješenja na računalu te predstavljaju identičan izgled aplikacije na mobilnim uređajima. U primjerima u nastavku prikazan je imaginarni slučaj projekta pod nazivom „Jednostavna aplikacija“ u kojoj na razvoju sudjeluju dva stručnjaka za razvoj softvera na zadacima vezanim uz izradu grafičkog rješenja za aplikaciju. Slika 4.33. prikazuje gantogram u horizontalnoj orijentaciji umjesto portretnog načina definiranog u tehničkim zahtjevima iz razloga uvida u više informacija u vremenskoj lenti.



Slika 4.20. Prikaz početnog zaslona

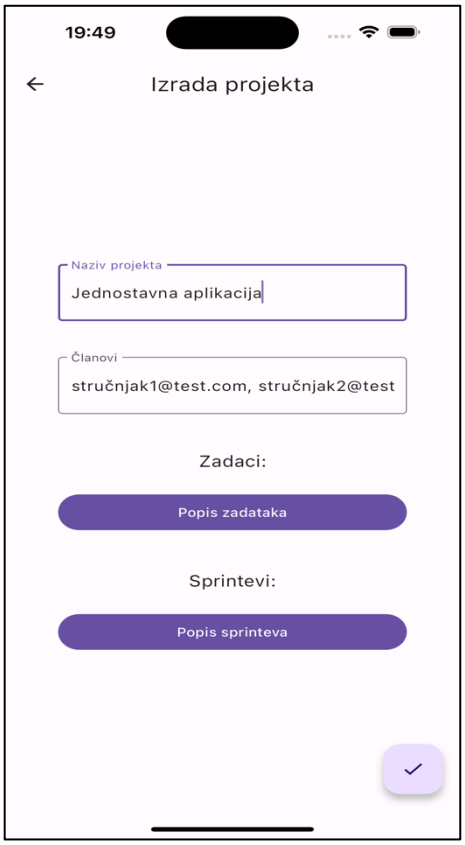


Slika 4.21. Prikaz zaslona za prijavu u sustav



Slika 4.22. Prikaz popisa projekata

Slika 4.24. Izrada i uređivanje zadatka

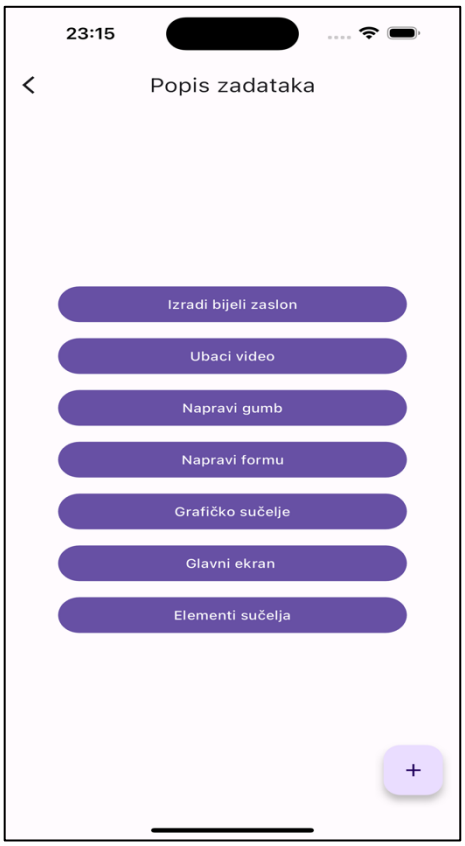


Slika 4.23. Prikaz zaslona za izradu projekta

Slika 4.25. Prikaz izrade epica i povezivanja zadatka s inicijativom

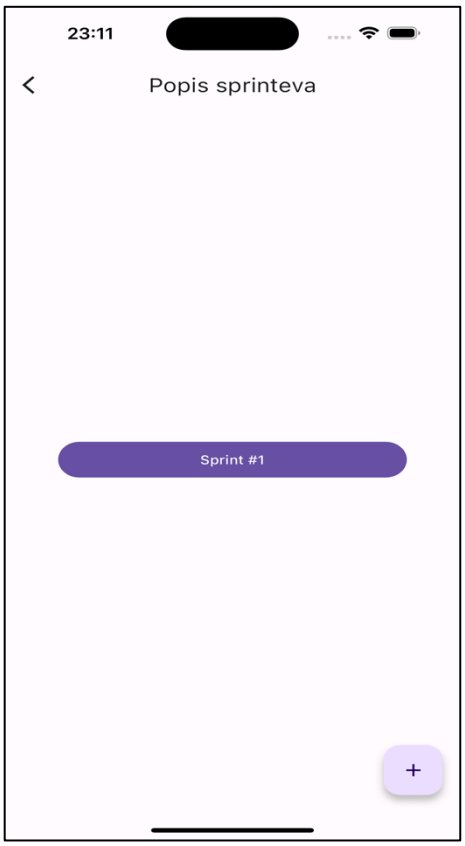
Slika 4.26. Prikaz izrade korisničke priče i povezivanja s epicom

Slika 4.27. Prikaz izrade korisničke priče

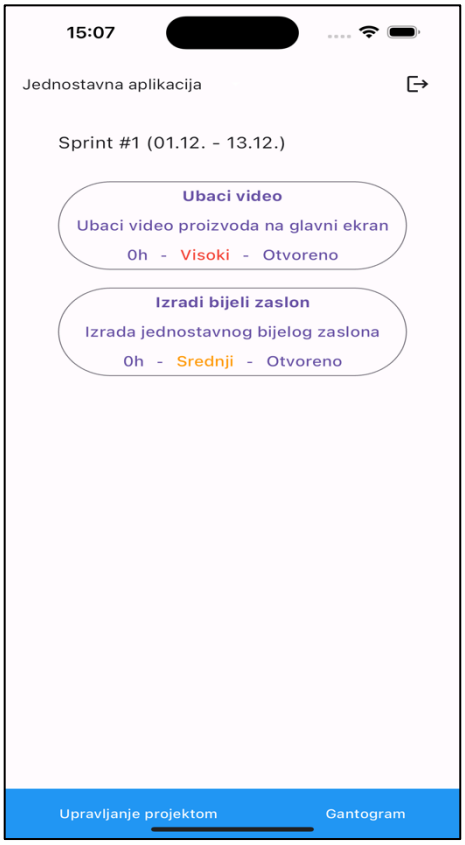


Slika 4.28. Prikaz popisa svih zadataka

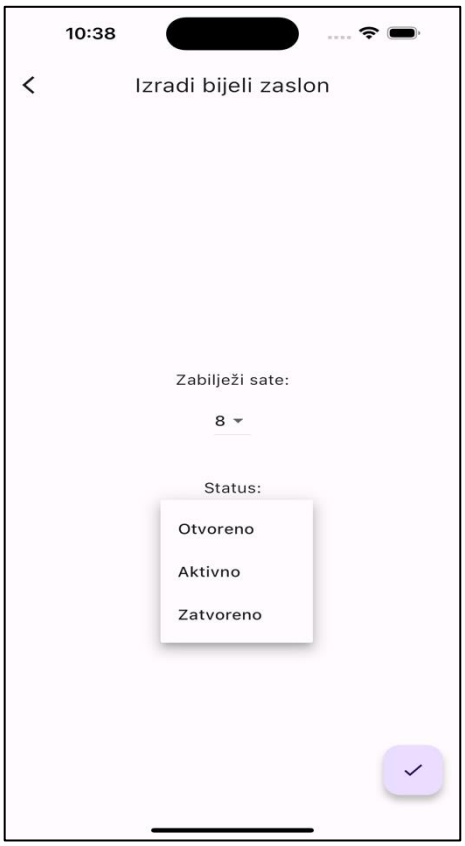
Slika 4.29. Prikaz izrade sprinta s popisom korisničkih priča



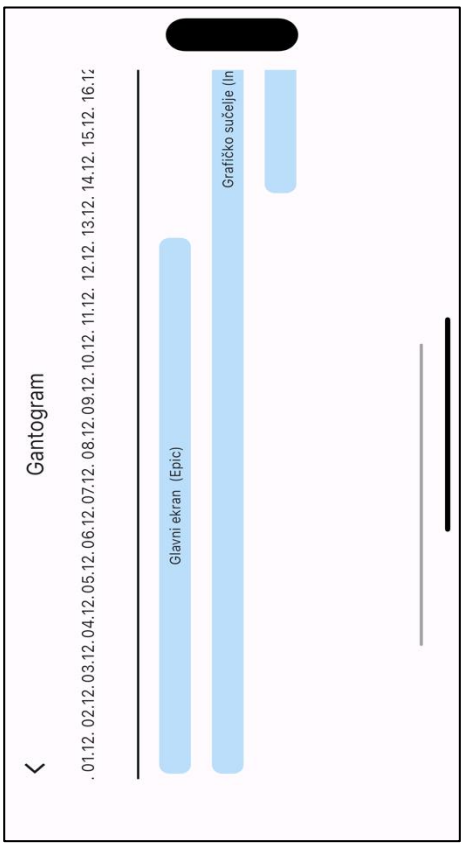
Slika 4.30. Prikaz popisa sprinteva



Slika 4.31. Prikaz popisa zadataka na nadzornoj ploči



Slika 4.32. Prikaz zaslona za evidentiranje rada na zadatku



Slika 4.33. Prikaz gantograma

4.3.2. Validacija odabranog skupa korisničkih zahtjeva

U okviru validacije korisničkih zahtjeva provedena su tri općenita korisnička scenarija nad odabranim skupom kojim su predstavljene glavne funkcionalnosti aplikacije i najočekivaniji načini korištenja. Provedba korisničkih scenarija i uspješan stvarni rezultat izvedbe scenarija pridonosi validaciji programskog rješenja. Korisnički scenariji raspisani su u obliku tablice i nalaze se u nastavku u tablicama (4.4. do 4.6.). Svaki korisnički scenarij sadrži identifikator, hipotetski scenarij, preduvjet, akcijske korake koji pridonose ostvarenju ciljeva, očekivani krajnji cilj i stvarni krajnji rezultat.

Tablica 4.4. Korisnički scenarij KS1

Identifikator	KS1
Opis	Novi korisnik želi stvoriti projekt naziva u trajanju mjesec dana. Također, korisnik želi dodati dva člana koja će raditi na dvije korisničke priče i stvoriti sprint u trajanju mjesec dana koji sadržava sve zadatke raspisane u prvoj iteraciji raspisivanja.
Preduvjet	Korisnik ima pristup aplikaciji
Korisnički koraci	1. Korisnik pokreće aplikaciju 2. Korisnik odrađuje registraciju u aplikaciju koristeći e-poštu i zaporku 3. Korisnik odabire stvaranje novog projekta 4. Korisnik upisuje / odabire: i. Odabire datum početka i očekivanog kraja projekta ii. Upisuje e-pošte dva člana projekta 5. Korisnik pritišće na gumb „Popis zadataka“ 6. Korisnik odabire izradu novog zadatka 7. Korisnik: i. Upisuje naziv i kratki opis zadatka ii. Odabire datum početka i očekivanog kraja zadatka iii. Odabire prioritet zadatka iv. Odabire korisničku priču kao tip zadatka v. Odabire osobu kojoj pridružuje zadatak

	vi. Pritišće gumb za završetak izrade zadatka 8. Korisnik ponavlja korak 6. i 7. 9. Korisnik pritišće gumb za povratak na prethodni zaslon 10. Korisnik pritišće na gumb „Popis sprinteva“ 11. Korisnik: i. Odabire datum početka i kraja trajanja sprinta ii. Odabire oba zadatka dostupna u popisu dostupnih korisničkih priča iii. Pritišće gumb za završetak izrade sprinta 12. Korisnik pritišće gumb za povratak na prethodni zaslon 13. Korisnik pritišće gumb za završetak izrade projekta
Očekivani cilj	Prikaz oba zadatka u sprintu na nadzornoj ploči
Stvarni rezultat	Prikaz oba zadatka u sprintu na nadzornoj ploči

Tablica 4.5. Korisnički scenarij KS2

Identifikator	KS2
Opis	Voditelj projekta na postojećem projektu unutar aplikacije želi dodati jednu inicijativu, dva *epica* i četiri korisničke priče od kojih su dvije vezane za prvi *epic*, dok su druge dvije vezane za drugi. U jedan sprint trajanja mjesec dana želi dodati dvije korisničke priče vezane za prvi *epic*. Svaki tip zadataka želi međusobno hijerarhijski povezati te želi generirati gantogram.
Preduvjet	Korisnik je voditelj projekta koji postoji u bazi podataka
Korisnički koraci	1. Korisnik pokreće aplikaciju 2. Korisnik se prijavljuje pomoću e-pošte i zaporke 3. Korisnik odabire postojeći željeni projekt 4. Korisnik pritišće na gumb „Popis zadataka“ 5. Korisnik odabire izradu novog zadatka 6. Korisnik: i. Upisuje naziv i kratki opis zadatka ii. Odabire datum početka i očekivanog kraja zadatka iii. Odabire prioritet zadatka

	iv. Odabire inicijativu kao tip zadatka v. Pritišće gumb za završetak izrade zadatka 7. Korisnik odabire izradu novog zadatka 8. Korisnik: i. Upisuje naziv i kratki opis zadatka ii. Odabire datum početka i očekivanog kraja zadatka iii. Odabire prioritet zadatka iv. Odabire *epic* kao tip zadatka v. Odabire prethodno izrađenu inicijativu kao povezani zadatak vi. Pritišće gumb za završetak izrade zadatka 9. Korisnik ponavlja korak 7. i 8. 10. Korisnik odabire izradu novog zadatka 11. Korisnik: i. Upisuje naziv i kratki opis zadatka ii. Odabire datum početka i očekivanog kraja zadatka iii. Odabire prioritet zadatka iv. Odabire *korisničku priču* kao tip zadatka v. Odabire *epic* kojem pripada vi. Odabire osobu kojoj pridružuje zadatak vii. Pritišće gumb za završetak izrade zadatka 12. Korisnik ponavlja tri puta korake 10. i 11. 13. Korisnik pritišće gumb za povratak na prethodni zaslon 14. Korisnik pritišće na gumb „Popis sprinteva“ 15. Korisnik: i. Odabire datum početka i kraja trajanja sprinta ii. Odabire dvije korisničke priče koje pripadaju prvom *epicu* iii. Pritišće gumb za završetak izrade sprinta 16. Korisnik pritišće gumb za povratak na prethodni zaslon 17. Korisnik pritišće gumb za završetak izrade projekta

	18. Korisnik pritišće na gumb „Gantogram“
Očekivani cilj	Prikaz sprinta na nadzornoj ploči i prikaz gantograma
Stvarni rezultat	Prikaz sprinta na nadzornoj ploči i prikaz gantograma

Tablica 4.6. Korisnički scenarij KS3

Identifikator	KS3
Opis	Stručnjak za razvoj softvera radi na projektu i zadacima te želi evidentirati rad na dodijeljenim mu zadacima tako što na dva zadatka koja ima dostupna u tekućem sprintu želi jedan prebaciti u status „aktivno“ dok drugi želi prebaciti u status „zatvoreno“. Na oba zadatka želi zabilježiti četiri radna sata.
Preduvjet	Korisnik je uključen u projekt i posjeduje dodijeljene zadatke
Korisnički koraci	1. Korisnik pokreće aplikaciju 2. Korisnik se prijavljuje pomoću e-pošte i zaporke 3. Korisnik odabire projekt na kojem radi 4. Korisnik odabire prvi zadatak u popisu zadataka tekućeg sprinta 5. Korisnik bilježi sate odabirom broja četiri u padajućem popisu sati rada 6. Korisnik odabire status aktivno u padajućem izborniku statusa zadatka 7. Korisnik pritišće gumb za završetak evidentiranja rada 8. Korisnik odabire drugi zadatak u popisu zadataka tekućeg sprinta 9. Korisnik bilježi sate odabirom broja četiri u padajućem popisu sati rada 10. Korisnik odabire status zatvoreno u padajućem izborniku statusa zadatka 11. Korisnik pritišće gumb za završetak evidentiranja rada
Očekivani rezultat	Prikaz točnih informacija o statusu i radnim satima na zadatku u nadzornoj ploči
Stvarni rezultat	Prikaz točnih informacija o statusu i radnim satima na zadatku u nadzornoj ploči

5. ZAKLJUČAK

U suvremenom okruženju malih IT poduzećima dinamičnost predstavlja značajan izazov u vođenju projekata, budući da se zahtjevi i okolnosti mogu brzo mijenjati, stalne potrebe za nadogradnjama dodatno povećavaju kompleksnost vođenja projekata, izlažući projekte i poduzeća izazovima održavanja stabilnosti i konkurentnosti. Dodatni izazovi proizlaze iz korištenja alata za vođenje projekata, koji, iako korisni, mogu biti ponekad komplicirani, skupi i zahtijevati dodatno vrijeme za obuku.

U ovome radu uspoređene su sekvencijske i agilne metodologije vođenja IT projekata u kojoj je zaključeno kako za dinamično tržište najviše odgovara agilna metodologija. Pošto je *Scrum* agilni radni okvir te jedan od najpopularnijih u IT svijetu poslužio je kao podloga pri izradi aplikacije za vođenje IT projekata za mobilne uređaje. Aplikacija izrađena za potrebe diplomskog rada predstavlja jednostavno rješenje za upravljanje projektom s mogućnostima: razdvajanja pogleda za voditelja i članove na projektu, izrade zadataka i *sprinteva*, evidentiranja rada na zadacima, stvaranja gantograma te hijerarhijskog povezivanja zadataka. Aplikacija izrađena pomoću *Flutter* kompleta za razvoj softvera omogućava krajnjim korisnicima, poduzećima, prilagođavanje pregleda i funkcija s obzirom kako je izvorni kod programskog rješenja dostupan javno. Aplikacija je validirana izvršavanjem korisničkih scenarija te validacija aplikacije na taj način predstavlja ključnu fazu u potvrđivanju njezine funkcionalnosti i korisničke prihvatljivosti.

Proučavajući trenutnu ponudu alata za vođenje IT projekata, evidentna je jaka konkurencija na tržištu. Identificirana su moguća unaprjeđenja koja obuhvaćaju dodavanje više razina uloga na projektu, implementaciju sustava obavijesti, dodavanje detaljnijih statistika projekta te integraciju drugih agilnih radnih okvira. Ova prepoznata poboljšanja imaju potencijal unaprijediti funkcionalnost aplikacije, stvarajući podlogu i mogućnosti za povećanje njezine konkurentnosti.

LITERATURA

[1] M, Esteves, T, Rodrigues, L.R, Sanjuliao and V, Hugo, Project Management In Industry 4.0: Technologies and skills supporting project managers, Revista Mundi Engenharia, br.8, sv.5, str. 2-12, 2020.,

[2] E, Rahim, M, Dawson, IT Project Management Best Practices in an Expending Market, Journal of Information Systems Technology and Planning, br.5, sv.3, str. 27-34, 2010.

[3] Y. H, Kwak, F. T, Anbari, E. G, Carayannis, The Story of Managing Projects, Greenwood Publishing Group, Westport, 2005.

[4] D. T, Seymour, S, Hussein, The History Of Project Management, International Journal of Management & Information Systems (IJMIS), br.4, sv.18, str. 233-238, 2014.

[5] T, Efran, L, Volonino, G, Wood, Information technology for management - Ninth Edition, Willey Global Education, 2012.

[6] M, Jeremiah, B, Kabey, M, Kabey, Evolution of Project Management, Monitoring and Evaluation, with Historical Events and Projects that Have Shaped the Development of Project Management as a Profession, International Journal of Science and Research, br.12, sv.8, str. 63-77, 2019.

[7] M. A, Omazić, S, Baljkas, Projektni menadžment, Sinergija nakladništvo, Zagreb 2005..

[8] I. Sobočan, Upravljanje portfeljem projekata, Sveučilište Jurja Dobrile u Puli, Pula, 2016..

[9] PM Institute, Vodič kroz znanje o upravljanju projektima (Vodič kroz PMBOK) - četvrto izdanje, Mate d.o.o., Zagreb, 2011..

[10] P, Podreka, Model životnog ciklusa softvera – sekvencijske i agilne metode, Sveučilište Jurja Dobrile u Puli, Pula, 2018.

[11] R, Pressman, Software Engineering: A Practitioner's Approach.7th Edition, McGraw Hill, New York, 2010.

[12] C, Reaiche, S, Papavasiliou, The Traditional, Sequential Methodologies, [online], dostupno na: https://jcu.pressbooks.pub/pmmethods/chapter/traditional-sequential-methodologies/, 2022., [30.6.2023.]

[13] D, Young, Software Development Methodologies, Alabama Supercomputer Center, 2013.

[14] R, Magner, Poglavlje 4: Verifikacija i validacija, [online], Sveučilište u Zagrebu, Zagreb, 2022., dostupno na: http://web.studenti.math.pmf.unizg.hr/~manger/si/SI-4.pdf, [30.10.2023]

[15] L, Wang, Research of Software Development Model Based on the Theory of Software Engineering, *ICCIA 2012*, str. 1171-1173, 2012.

[16] J, Sutherland, K, Schwaber , THE SCRUM PAPERS: NUT, BOLTS, AND ORIGINS OF AN AGILE FRAMEWORK, Scrum Inc., Cambridge, 2011.

[17] SCRUM agilna metoda, [online], Info Novitas, Zagreb, dostupno na: https://www.info-novitas.hr/o-nama/metodologije-rada/scrum-procesni-framework/ [1.11.2023.]

[18] M, Rehkopf, Stories, epics, and initiatives, [online], Atlassian, dostupno na: https://www.atlassian.com/agile/project-management/epics-stories-themes, [31.10.2023.]

[19] ClickUp - Manage Teams & Tasks, ClickUp, [online], dostupno na: HYPERLINK "https://apps.apple.com/us/app/clickup-manage-teams-tasks/id1535098836" https://apps.apple.com/us/app/clickup-manage-teams-tasks/id1535098836, [4.11.2023.]

[20] Trello: organize anything!, Trello, [online], dostupno na: HYPERLINK "https://apps.apple.com/us/app/trello-organize-anything/id461504587" https://apps.apple.com/us/app/trello-organize-anything/id461504587, [4.11.2023.]

[21] Asana: Work in one place, Asana, [online], dostupno na: HYPERLINK "https://apps.apple.com/us/app/asana-work-in-one-place/id489969512" https://apps.apple.com/us/app/asana-work-in-one-place/id489969512, [4.11.2023.]

[22] Jira Cloud by Atlassian, Atlassian, [online], dostupno na: https://apps.apple.com/hr/app/jira-cloud-by-atlassian/id1006972087, [4.11.2023.]

[23] Microsoft Planner, Microsoft, [online], dostupno na: https://apps.apple.com/us/app/microsoft-planner/id1219301037 , [5.11.2023.]

[24] Flutter, [online], dostupno na: https://flutter.dev, [1.11.2023.]

[25] Firebase, [online], dostupno na: https://firebase.google.com, [1.11.2023.]

[26] JSON (format za razmenu podataka), [online], Web Programiranje, dostupno na: https://www.webprogramiranje.org/json/, [5.11.2023.]

[27] I, Chinedu, Firebase Authentication Using a Custom Token, [online], Progress Telerik, 2023., dostupno na: https://www.telerik.com/blogs/firebase-authentication-using-custom-token, [5.11.2023]

[28] J, Khambhayta, Flutter: Provider Pattern (Blueprint), [online], Medium, 2021., dostupno na: https://medium.com/@khambhaytajaydip/flutter-provider-pattern-blueprint-4ab847e247e9, [3.11.2023.]

SAŽETAK

Projektni menadžment u IT industriji i razvoju softvera ključan je za uspješno vođenje projekata. Ovaj proces obuhvaća planiranje, praćenje i upravljanje resursima, vremenom i zadacima kako bi se izvršio cilj. Korištenje agilnih metoda, kao i kontinuirano prilagođavanje promjenama u zahtjevima, igraju ključnu ulogu u postizanju ciljeva u dinamičnom IT okruženju. Kao rezultat rada izrađena je više platformska mobilna aplikacija koja omogućuje upravljanje IT projektima agilnim radnim okvirom *scrum* unutar organizacija koje si ne mogu priuštiti napredne alate. Korisnicima je omogućena izrada projekta, uključivanje više sudionika projekta, stvaranje i uređivanje zadataka, upravljanje *sprintevima*, stvaranje gantograma i evidentiranje rada na zadacima. Aplikacija je stvorena unutar *Flutter* kompleta za razvoj softvera koristeći *Dart* programski jezik. Pomoću *Firebase* baze podataka omogućena je provjera identiteta i prijava korisnika u sustav, spremanje i dohvaćanje projektnih podataka. Validacija aplikacije provedena je korištenjem testnih slučajeva na glavnim funkcionalnostima aplikacije.

Ključne riječi: agilna metodologija, informacijska tehnologija, mobilna aplikacija, projektni menadžment, scrum

ABSTRACT

Methods and mobile application for IT project management

Effective project management is crucial for success in the IT industry and software development. This process involves planning, monitoring, and managing resources, time, and tasks to accomplish the project goal. The use of agile methods, as well as continuous adaptation to changes in requirements, play a key role in achieving goals in a dynamic IT environment. As a result, a multi-platform mobile application was created that enables the management of IT projects with an agile scrum framework within organizations that cannot afford advanced tools. Users can create a project, include multiple project participants, create and edit tasks, manage sprints, create a Gantt chart, and record work on tasks. The application was created within the Flutter software development kit using the Dart programming language. Utilizing the Firebase database allows for user identity verification, user login, and the storage and retrieval of project data. The application underwent validation through test cases focusing on its core functionalities.

Key words: agile methodology, information technology, mobile application, project management, scrum

ŽIVOTOPIS

Petar Biočić rođen je 27.1.1999. u Osijeku. Osnovno obrazovanje započinje 2005. godine u Osnovnoj školi Višnjevac. Po završetku osnovne škole, 2013. godine nastavlja obrazovanje u Elektrotehničkoj i prometnoj školi Osijek gdje upisuje smjer za elektrotehničara. Srednjoškolsko obrazovanje završava 2017. godine te upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija u Osijeku, smjer Računarstvo. Nakon završenog preddiplomskog studija 2021. godine upisuje diplomski studij Automobilsko računarstvo i komunikacije na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku.

PRILOZI

Prilog 1: Pristup izvornom kodu na *GitHubu*: https://github.com/pbiocicorqa/diplomski_pm.git