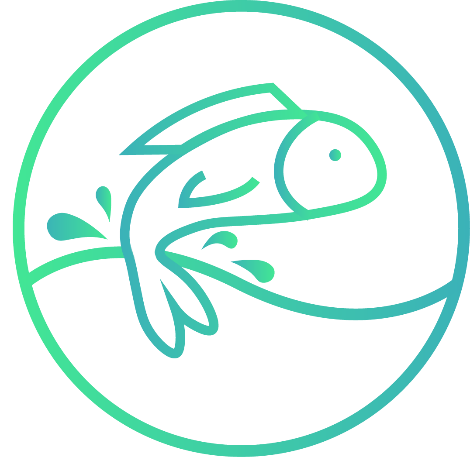
Home

**Projekt iz kolegija Programsko inženjerstvo**

**Naziv projekta:** Digitalna ribarnica



Logo aplikacije

**Oznaka tima:** PI20-001

Božo Kvesić, 45866/17-R

Nikola Muše, 46192/17-R

Anabela Pranjić, 45869/17-R

# Projektna dokumentacija

## O projektu

(Nikola Muše)

Digitalna ribarnica je projekt zamišljen kao aplikacija koja omogućava trgovcima ribe dnevno postavljanje ponude riba u kojoj navode količinu, cijenu, lokaciju i ostale elemente ponude. Korisnik aplikacije (kupac) može odabrati lokaciju i pretražiti različite vrste ribe te rezervirati količinu ribe. Unutar aplikacije izrađen je sustav obavijesti i razmjene poruka koji olakšava komunikaciju između trgovca i kupca.

Projektna dokumentacija sadrži bitne podatke o opisu metodike razvoja softvera, zaduženjima članova tima, terminskom planu projekta, popisu resursa, opisu alata korištenih u realizaciji projekta, proračun i budžet projekta, ponuda naručitelju, definirane metrike, izvješće o provedbi, analiza metrika i plan korekcije.

Osim projektne dokumentacije, izrađena je i tehnička dokumentacija koja sadrži raspis svih tehničkih specifikacija programskog proizvoda u svakoj fazi razvoja. U sadržaj tehničke dokumentacije su uključeni UML dijagrami slučajeva korištenja, dijagrami slijeda i aktivnosti, dijagram klasa, dijagram podataka, ERA model, detaljna specifikacija zahtjeva, wireframe korisničkog sučelja, implementacijski artefakti te podaci o testiranju.

Korisnička dokumentacija sadrži detaljne upute za korištenje programa. Dostupna je unutar dokumentacije te unutar aplikacije – odabirom opcije „Help“ ili pritiskom tipke F1.

## Metodika razvoja softvera

(Anabela Pranjić)

Za izradu aplikacije odabran je vodopadni model u kojem se sekvencijalno razvija faza po faza projekta te naglašava strukturirani napredak kroz definirane faze. Vodopadni model je iznimno koristan za manje projekte definiranih zahtjeva koji imaju određen rok i budžet što odgovara našem opisu projekta. Za vodopadni model je karakteristično da su zahtjevi sustava definirani dugo prije nego što započne sama implementacija te su promjene zahtjeva svedene na minimum kako se projekt razvija. Sastoji se od nekoliko faza čija imena mogu varirati ali iza svake postoji osnovna premisa.

Prva faza se odnosi na definiranje sustavnih i softverskih zahtjeva, druga faza definira dizajn softvera, treća faza je posvećena implementaciji, četvrta faza se odnosi na testiranje, a u petoj fazi se proizvod isporučuje i održava. Izrada projekta prati navedene faze uz dodatnu početnu fazu planiranja razvoja projekta u kojoj se formira tim i proučavaju inicijalne upute mentora kako bi se što bolje pripremilo za sljedeće faze. [prema SoftwareHut - Waterfall 101: an Introduction to Waterfall Software Development, preuzeto dana 10.4.2020.]

Sljedeća slika prikazuje vodopadni model. [prema XB Software Blog - Waterfall Model in SDLC, preuzeto dana 10.4.2020.]



## 1.3 Projektni tim i zaduženja

(Božo Kvesić)

Tim PI20-001 čine studenti 3. godine PDS Informacijski sustavi na Fakultetu Organizacije i Informatike u Varaždinu: Božo Kvesić, Nikola Muše i Anabela Pranjić. Sljedeća tablica prikazuje zaduženja svakog pojedinog člana tima.

|  |  |
| --- | --- |
| IME I PREZIME | ZADUŽENJA |
| Božo Kvesić | Registracija korisnika  Prijava korisnika  Administracija korisnika  Rezervacija  Komunikacija između korisnika  Administracija ponuda  Pregled izvršenih narudžbi  Sustav obavijesti  Pregled dnevnika  Rad s bazom podataka |
| Nikola Muše | Ocjenjivanje korisnika  Potvrđivanje rezervacije  Komunikacija između korisnika  Administracija postavki sustava  Administracija ponuda  Dodavanje kontakt podataka  Dodavanje pripadnih elemenata ponude  Sustav znački  Pregled dnevnika  Rad s bazom podataka |
| Anabela Pranjić | Registracija korisnika  Pregledavanje ponuda  Ocjenjivanje korisnika  Dodavanje ponude  Pregled izvršenih narudžbi  Sustav znački  Promjena količine dostupne ribe  Potvrđivanje preuzimanja ribe  Korisničke upute  Rad s bazom podataka |

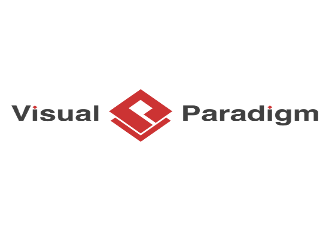
## 1.4 Alati za izradu projekta

(Nikola Muše)

Za izradu projekta korišteni su brojni alati poput MS Office paketa za oblikovanje dokumenata i tablica čiji je sadržaj potom uređen putem GitHub Wiki sustava. Nadalje, korišten je online alat Visual Paradigm za kreiranje UML dijagrama, Microsoft Project za izradu terminskog plana projekta i gantograma. Za izradu ERA modela korišten je MySQL Workbench te SQL Server Management Studio za izradu baze podataka. Implementacija je odrađena u programskom jeziku C#, u alatu Microsoft Visual Studio. Wireframe aplikacije je izrađen u online alatu Balsamiq. Kako bi kreirali i oblikovali logo aplikacije, koristili smo alate Adobe Illustrator i Adobe Photoshop koristeći besplatnu probnu verziju. Za neke alate imamo licence koje smo dobili tijekom školovanja, dok za pojedine alate nije bila potrebna licenca.





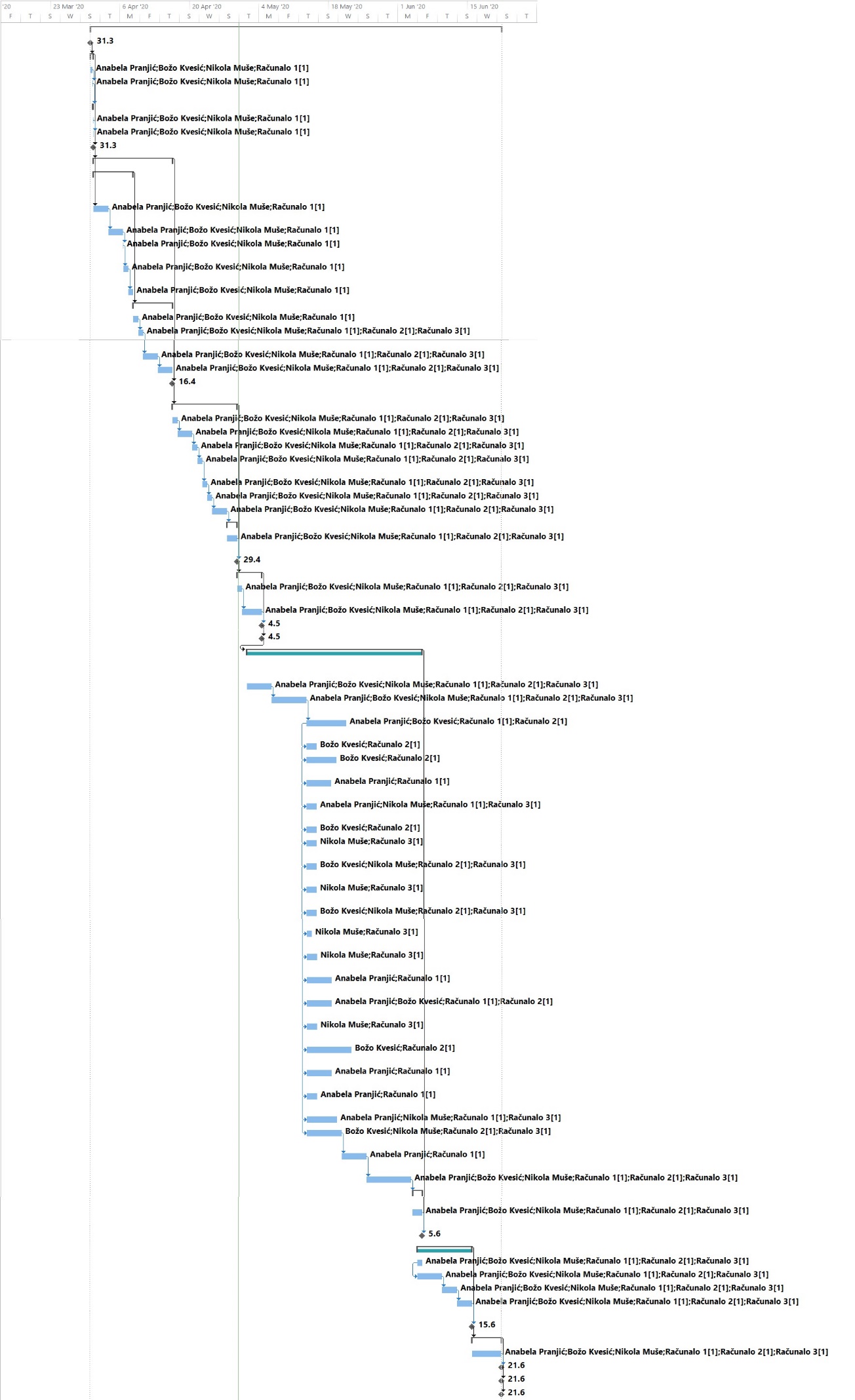


1.5. Terminski plan projekta

(Anabela Pranjić)

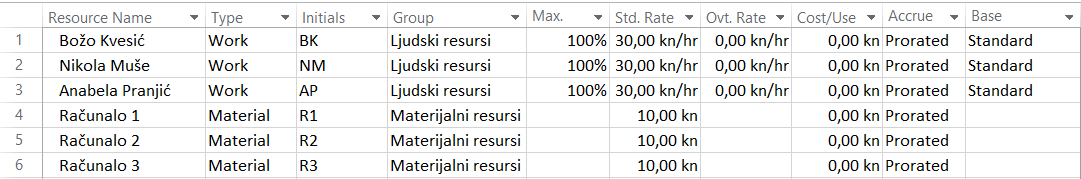
Za izradu terminskog plana projekta koristili smo alat MS Project. Rad na projektu započeo je 31.3.2020., a predviđen završetak rada je 21.6.2020. Projekt smo podijelili u nekoliko faza: Planiranje projekta, Analiza i specifikacija zahtjeva, Modeliranje, Dizajn, Implementacija, Testiranje i Isporuka. Ljudski resursi su članovi tima: Božo Kvesić, Nikola Muše i Anabela Pranjić. Materijalni resursi se odnose na 3 računala – Računalo 1, Računalo 2 i Računalo 3. Radno vrijeme članova tima je cijeli tjedan, a radno vrijeme iznosi pet sati dnevno. Sljedeće slike prikazuju faze projekta sa pripadnim zadacima i sudjelovanjem resursa te pripadnim gantogramom za izrađeni plan.



1.6. Resursi

(Božo Kvesić)

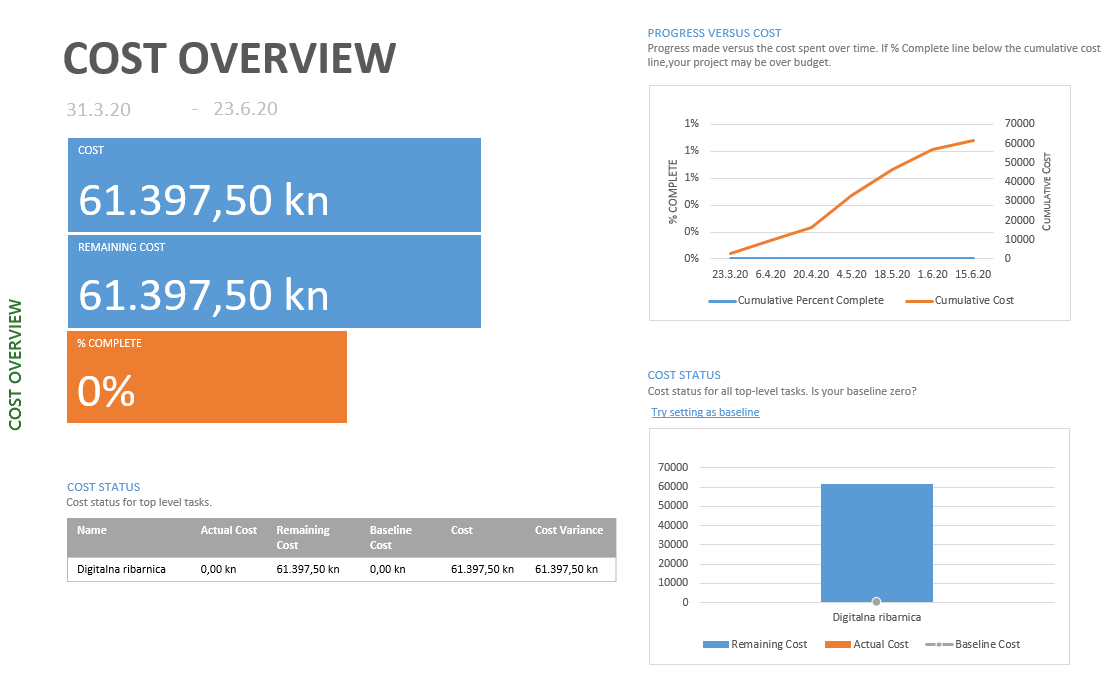
Na prethodnim stranicama su popisani zadaci te zaduženja članova tima. Navedeni su ljudski i materijalni resursi potrebni za uspješno izvršavanje zadataka u predviđenom roku. Sljedeća slika prikazuje popis resursa i njihove satnice.

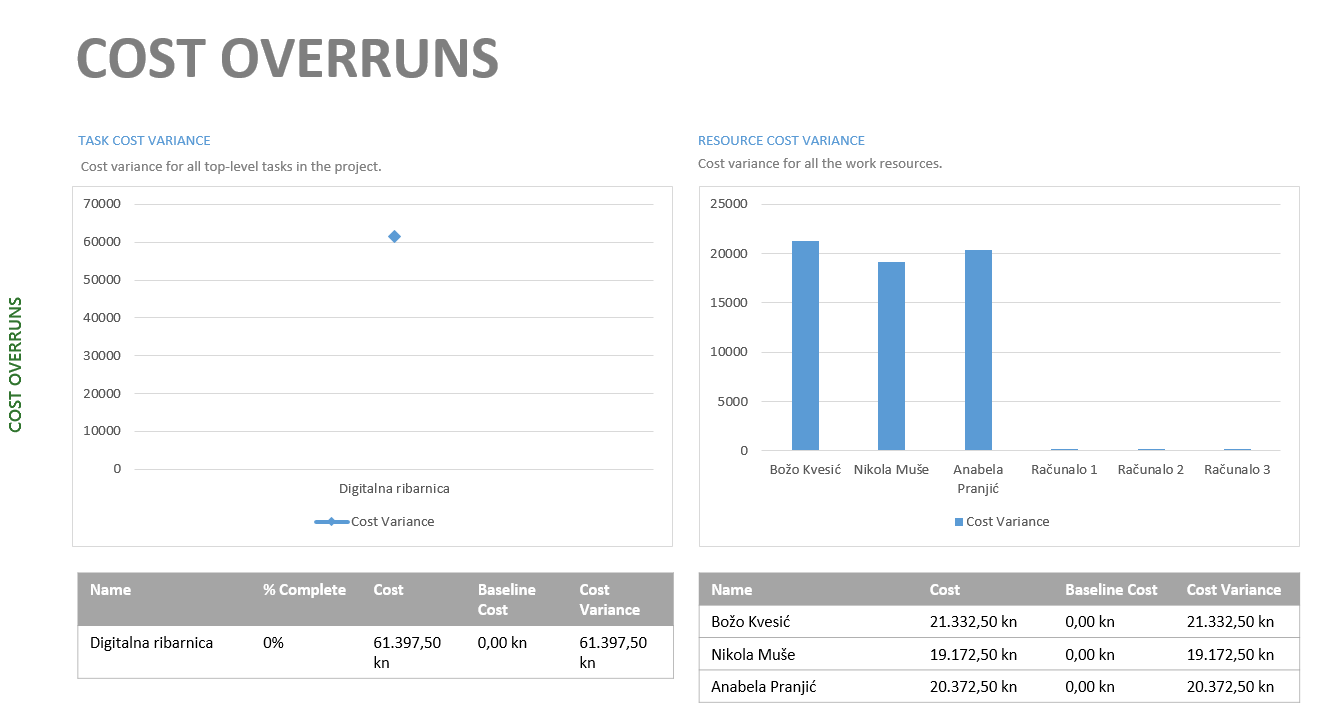


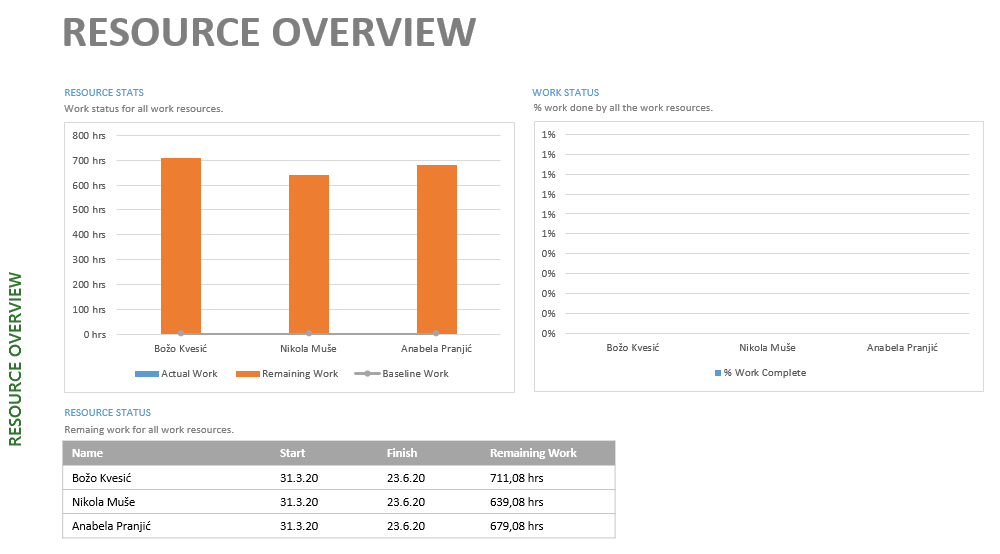
1.7. Proračun projekta

(Anabela Pranjić)

Unosom aktivnosti, resursa i pripadnih troškova, došli smo do cijene projekta koja iznosi 61.397,50 HRK. Cijena uključuje izradu dokumentacije i aplikacije. Nakon što u cijenu uračunamo PDV od 25% dobijemo 76.746,875 HRK. Proračun projekta smo također izradili u alatu MS Project te smo kreirali nekoliko izvještaja koji su priloženi na sljedećim stranicama.



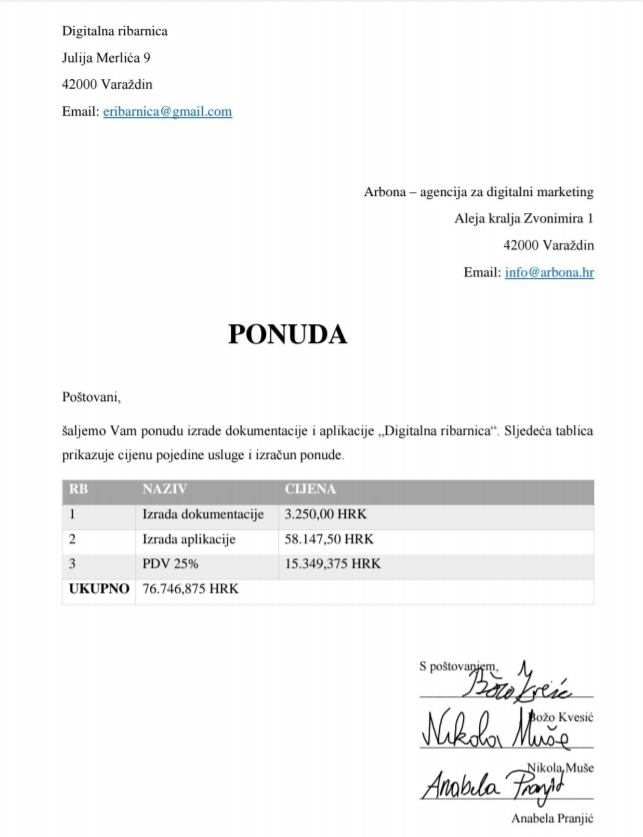




1.8. Ponuda naručitelju

(Nikola Muše)

Sljedeća slika sadrži ponudu naručitelju izrađenu prema podacima iz alata MS Project. Ponuda uključuje izradu dokumentacije i aplikacije.



1.9. Definirane metrike

(Anabela Pranjić)

Izvor: Project Management Metrics, KPIs and Dashboards, <https://www.academia.edu/26281895/Project_Management_Metrics_KPIs_and_Dashboards>

<http://computingtechnologies.blogspot.com/2009/12/software-quality-metrics.html>

<https://tensix.com/2019/02/7-risk-management-metrics-to-track/>

<https://uplandsoftware.com/psa/resources/blog/5-resource-management-kpis-you-can-use-today/>

<https://support.office.com/en-us/article/cost-variance-fields-10739faf-4b5e-4303-bed8-aa5663604526>

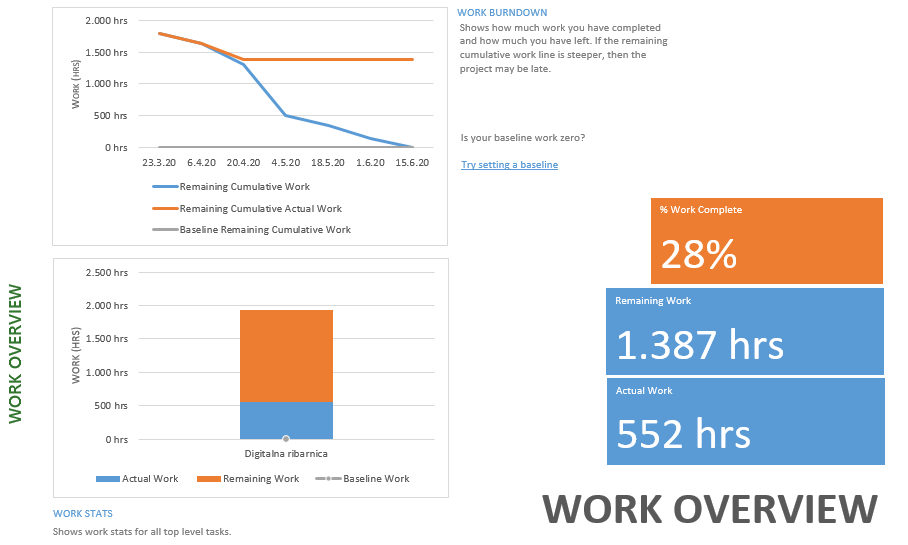
Status projekta se ne može točno odrediti korištenjem samo metrika u obliku vremena i troškova – potrebne su i druge metrike. Korištenje metrika zahtjeva promjenu, a bez vjerodostojnih metrika ne možemo dobiti pouzdane izvještaje. Postoji općenito pravilo – ono što se ne može mjeriti, time ne možemo upravljati. Sljedeća tablica prikazuje metrike kojima bismo pratili napredak i razvoj projekta.

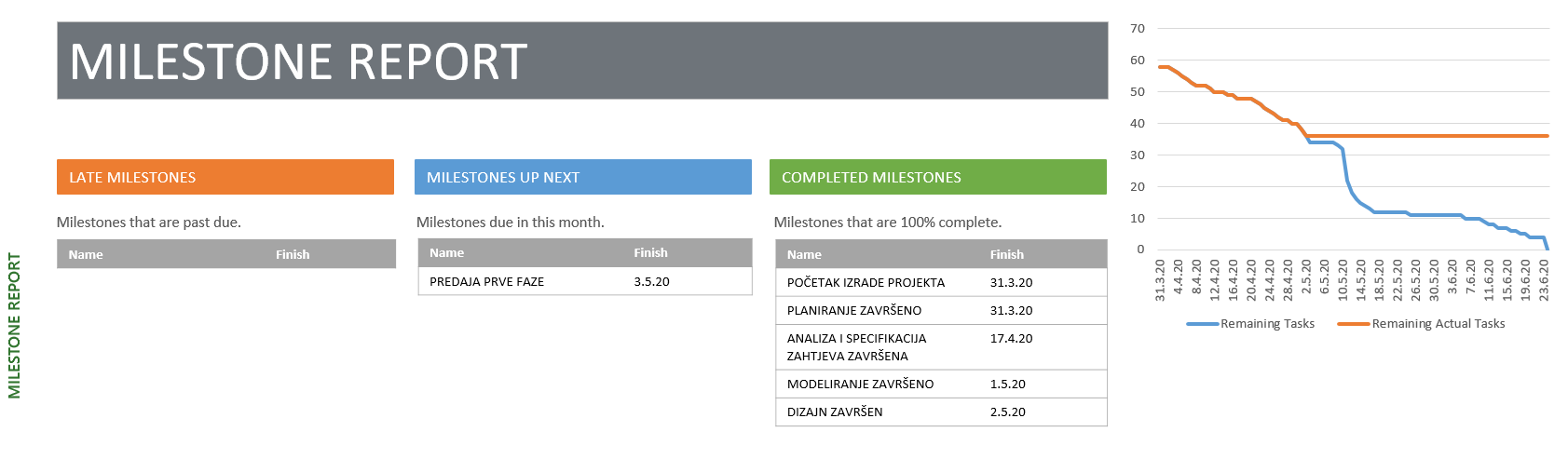
|  |  |
| --- | --- |
| Ciljevi | Metrike |
| Osiguravanje kvalitete zahtjeva | Dvosmislenost specifikacija zahtjeva – broj nezadovoljavajuće objašnjenih zahtjeva u dokumentaciji |
| Završenost – broj zahtjeva koji se tek moraju utvrditi (To Be Determined – TBD) i broj zahtjeva koji se moraju dodati (To Be Added – TBA) |
| Razumljivost – struktura i razumljivost dokumenta |
| Promjenjivost – broj zahtjeva koji se moraju mijenjati te broj promjena pojedinog zahtjeva |
| Usklađenost – broj zahtjeva koje programski kôd ne prati |
| Kvaliteta softvera | Struktura/Arhitektura – kompleksnost kôda |
| Održivost – otklanjanje novonastalih ili naknadno primijećenih pogrešaka |
| Iskorištavanje programskog kôda – primjena kreiranih klasa/metoda/sučelja i ostalih elemenata u drugim dijelovima programskog kôda |
| Interna dokumentacija – postotak komentiranja programskog kôda |
| Eksterna dokumentacija – objašnjenje koncepata uz pripadne UML dijagrame i korisničku dokumentaciju |
| Efektivnost implementacije | Korištenje resursa – broj sati koje resursi provedu za izvedbu zadataka |
| Stopa završenosti – broj završenih zadataka |
| Efektivnost testiranja | Korektnost – broj greški, vrijeme potrebno za pronalazak grešaka, vrijeme potrebno za ispravljanje greški |
| Omjer greški i broja linija programskog kôda |
| Reduciranje rizika | Broj identificiranih rizika – analiza i izdvajanje rizika po izvođenju zadataka koji se mogu pojaviti |
| Broj rizika koji su se pojavili i bili su identificirani – prema identificiranim rizicima; veći broj rizika koji su se pojavili i koji su prije identificirani rezultiraju dobrom analizom rizika |
| Broj rizika koji su se pojavili više puta – pojavljuju se kao posljedica rada koji ne prati smjernice za ispravljanje pogrešaka koje su se već dogodile |
| Broj rizika koji su se pojavili, a nisu bili identificirani |
| Trošak upravljanja rizicima – praćenje troška kojeg uzrokuje pojedini rizik, omogućava predviđanje troškova rizika |
| Broj rizika koji se nisu dogodili – bilo da su poduzete mjere sprečavanja događanja rizika ili ne |
| Efektivan/ produktivan rad resursa | Varijanca troškova za resurse (Resource Cost Variance) – računa se prema izrazu *Cost Variance = Cost - Baseline Cost*, gdje „*Cost*“ predstavlja trošak resursa utrošen prilikom obavljanja zadatka, a „*Baseline Cost*“ predstavlja predviđeni trošak resursa koji bi se pojavio prilikom obavljanja zadatka |
| Varijanca uloženog truda resursa (Resource Effort Variance) – računa se prema izrazu *Effort Variance = Forecasted Hours Worked – Actual Hours Worked*, gdje „*Forecasted Hours Worked*“ označava predviđene sate unutar kojih bi resurs trebao obaviti zadatak, a „*Actual Hours Worked*“ predstavlja stvaran broj sati potrebnih kako bi resurs obavio zadatak |

1.10. Izvješća o provedbi

(Božo Kvesić)

Pomoću alata MS Project kreirali smo dva izvješća koja su povezana sa provedbom projekta – Pregled obavljenog posla (Work Overview) i Izvještaj ključnih točki (Milestone Report). Prvo izvješće prikazuje grafove o završenosti poslova te koliko je posla ostalo do kraja projekta. Izvješće pokazuje da je otprilike 28% posla obavljeno. Obavljeni posao se odnosi na sve zadatke do predaje prve faze. Izvještaj ključnih točki prikazuje popis obavljenih točki te onih koje slijede u ovom mjesecu.





1.11. Analiza metrika

(Anabela Pranjić)

U prethodnim poglavljima popisane su brojne metrike koje služe praćenju napretka i razvoja projekta i vode nas k ostvarenju ciljeva poput – osiguravanja kvalitete zahtjeva i softvera, efektivnosti implementacije i testiranja, reduciranju rizika te osiguranju efektivnog, odnosno produktivnog rada resursa.

Ovo poglavlje je posvećeno detaljnijoj analizi pojedinih metrika te predstavljanju plana korekcije.

Za osiguravanje kvalitete zahtjeva postavljeno je nekoliko metrika koje se odnose na detaljnu, razumljivu specifikaciju zahtjeva, završenost zahtjeva, usklađenost i promjenjivost. Detaljna specifikacija zahtjeva je opisana u projektnoj dokumentaciji te sadrži popis utvrđenih funkcionalnosti koje je potrebno implementirati. Dvosmislenost specifikacije zahtjeva mjerimo na način da pobrojimo one zahtjeve u dokumentaciji koji ne daju dovoljno dobro objašnjenje, odnosno koje klijent nakon proučavanja dokumentacije shvati na krivi način. Nadalje, završenost se odnosi na broj zahtjeva koji se tek moraju utvrditi do kojih se dolazi u fazi implementacije kada tim odluči da bi trebalo dodati pojedine zahtjeve radi poboljšanja ili potrebe izrade nekog dijela aplikacije. Razumljivost predstavlja strukturu i razumljivost same dokumentacije na način da je u potpunosti razumljiva projektnom menadžeru i da se u svakom trenutku mogu iščitati zahtjevi. Promjenjivost broji one zahtjeve koji se moraju mijenjati tijekom implementacije te broj promjena pojedinog zahtjeva. Usklađenost se odnosi na broj onih zahtjeva koje programski kôd ne prati.

Kako bi osigurali kvalitetu softvera koristimo se sljedećim metrikama. Mjerimo kompleksnost programskog kôda na način da utvrđujemo koje komponente bi bile teže za održavanje i potencijalnu naknadnu izmjenu. Nadalje, u fazi održavanja ponekad je potrebno otklanjati novonastale ili naknadno primijećene pogreške. Stoga, kompleksnost programskog kôda tu ima veliku ulogu. Jedan od faktora uštede prilikom implementacije je iskorištavanje programskog kôda na način da se iznova primjenjuju programski artefakti i tako štede vrijeme i rad. Implementacijski dio zahtjeva izradu dokumentacije koju smo podijelili na internu i eksternu. Interna dokumentacija podrazumijeva komentiranje programskog kôda te na taj način mjerimo postotak komentiranja. Komentari su jasni i precizni. Eksterna dokumentacija se odnosi na objašnjenje koncepata koji su prethodno implementirani i prikazani na UML dijagramima i obrazloženi u korisničkoj dokumentaciji.

Efektivnost implementacije nastojimo ostvariti primjenom dvije metrike – korištenje resursa koje se odnosi na broj sati koje resursi provedu za izvedbu zadataka te stopa završenosti koja označava broj završenih zadataka.

Efektivnost testiranja se nastoji ostvariti brojanjem greški u programskom kôdu, vremenom potrebnom za pronalazak greški te vremenom potrebnom za ispravljanje grešaka. Također, omjer greški i broja linija programskog kôda je dobra mjera i pokazatelj korektno implementiranog zahtjeva.

Reduciranje rizika jedan je od ciljeva koji se mjere sa nekoliko metrika. Utvrđujemo broj identificiranih rizika koje dobijemo analizom i izdvajanjem rizika po izvođenju zadataka koji bi se mogli pojaviti tijekom razvoja projekta. Sljedeća metrika se odnosi na broj rizika koji su se pojavili i prethodno su bili identificirani. Upravo su takvi rizici dobar pokazatelj dobre analize rizika jer na taj način znamo da smo izradi dobru analizu i identifikaciju rizika koji bi se mogli pojaviti. Moguće je pojavljivanje istih rizika i nekoliko puta koji se pojavljuju kao posljedice rada koji ne prati smjernice za ispravljanje pogrešaka koje su se već dogodile. Postoje i rizici koji su se pojavili, a prije nisu bili identificirani. Upravljanje rizicima povlači i troškove, stoga je potrebno pratiti one troškove koje uzrokuje pojedini rizik. Na taj način možemo predvidjeti troškove rizika. Prilikom analize i identificiranja rizika, postoji mogućnost da se pojedini rizici ne dogode, bilo da su poduzete mjere njihovog sprečavanja ili ne. Stoga je broj takvih rizika još jedna metrika za osiguravanje što manjeg broja rizika.

Kako bi osigurali efektivan i produktivan rad resursa koristimo se metrikama varijance troškova za resurse i varijance uloženog truda resursa. Oboje metrike imaju formulu izračuna prema kojoj tumačimo rezultate.

1.12. Plan korekcije

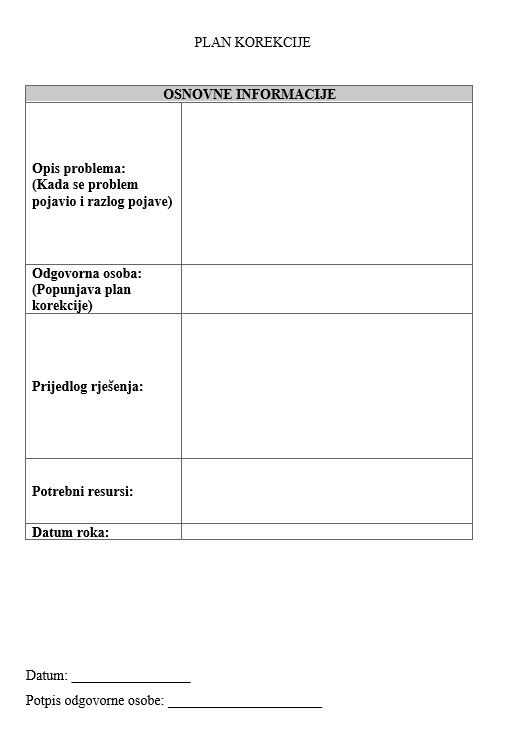
(Anabela Pranjić)

<https://www.cms.gov/research-statistics-data-and-systems/monitoring-programs/perm/downloads/2013correctiveactionpowerpoint.pdf>

<https://www.template.net/business/plan-templates/corrective-action-plan-template/>

Plan korekcije je dokument kojeg koriste poduzeća i organizacije kako bi spriječili i rješavali neželjene situacije i ponašanja. Služi identificiranju nepoželjnih problema i događaja te sadrži mjere za njihovo rješavanje. Plan korekcije svako poduzeće ili organizacija izrađuje po mjeri koja najbolje odgovara njihovom poslovanju. Svaki riješeni problem se dodaje na plan korekcije.

Postoji nekoliko stavki koje se nalazi na većini planova korekcije poput navođenja problema, navođenja odgovorne osobe, prijedloga rješenja, navođenja potrebnih resursa i krajnjeg roka. Naš predložak plana korekcije upravo sadrži navedene stavke. Opis problema zahtjeva navođenje i precizno definiranje problema koji se pojavio te kako bi ostali koji čitaju plan korekcije mogli razumjeti problem. Sljedeća stavka plana korekcije se odnosi na navođenje odgovorne osobe (ili osoba) koja je odgovorna za korekciju problema. Prijedlog rješenja bi trebao biti realan i vezan uz problem koji se pojavio. Istraži se uzrok problema i predloži rješenje koje je specifično, mjerljivo, relevantno i ostvarivo u zadanom vremenskom periodu. Potrebni resursi za rješavanje problema mogu se odnositi na ljudske resurse i/ili materijalne resurse. Konačno, navodi se vremenski rok do kojeg bi se problem trebao riješiti. Rok bi također trebao biti realno postavljen.



# Tehnička dokumentacija

2.1. Uvod

Poglavlje tehničke dokumentacije sadrži specifikaciju zahtjeva prema IEEE Std 830-1998 standardu, pregled zahtjeva softvera (Software Requirements Specification). Uvodno poglavlje sadrži potpoglavlja koja se odnose na svrhu, opseg, definicije, reference i pregled.

2.1.1. Svrha

Svrha ovog dokumenta je opis funkcionalnosti aplikacije i uloga. Specifikacija zahtjeva je namijenjena budućim korisnicima kao pomoć u korištenju.

2.1.2. Opseg

Kao što je navedeno u opisu projekta u projektnoj dokumentaciji, projekt Digitalna ribarnica služi jednostavnoj prodaji ribe. Kupac može pregledavati, pretraživati, filtrirati i sortirati ponude te rezervirati određenu količinu ribe. Aplikacija omogućava i komunikaciju između kupca i ponuditelja kako bi uspješno obavili prodaju. Aplikacija će raditi u dvije konfiguracije, ovisno o tome tko se ulogirao u nju, administrator ili registrirani korisnik (Kupac/Ponuditelj).

2.1.3. Definicije, akronimi, kratice

Korisnik: predstavlja korisnika aplikacije koji se prethodno registrirao u aplikaciju te prijavio

Administrator: vlasnik aplikacije

2.1.4. Reference

IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

2.1.5. Pregled

Sljedeće poglavlje donosi opis aplikacije sa sljedećim potpoglavljima: perspektiva proizvoda, funkcionalnost aplikacije, korisnički zahtjevi i ograničenja.

2.2. Opis aplikacije

U nastavku slijedi opis proizvoda, funkcionalnosti aplikacije, korisnički zahtjevi i ograničenja.

2.2.1. Perspektiva proizvoda

Digitalna ribarnica je namijenjena svakome tko želi prodat ili kupiti ribu. Olakšava trgovanje ponuditeljima na način da jednostavno sastave ponudu unutar aplikacije gdje navode osnove podatke ponude poput vrste ribe, lokacije, cijene, količine, opisa i trajanja rezervacije te mogu prihvaćati ili odbijati rezervacije ostalih korisnika aplikacije. Razvijen je i sustav komunikacije koji omogućava razmjenu poruka između kupca i ponuditelja kako bi mogli dogovoriti prodaju. Sustav obavijesti korisniku daje uvid u razne obavijesti poput zahtjeva za rezervacijom ponude, prihvaćenom rezervacijom, dobivenom porukom te ostalim sustavskim obavijestima. Korisnici se mogu međusobno ocjenjivati nakon provedene transakcije, ostavljati komentare te dobivati značke.

2.2.2. Funkcionalnosti aplikacije

U ovom potpoglavlju su popisane funkcionalnosti aplikacije uz kratke opise koje sustav treba zadovoljiti. Popis svih funkcionalnosti aplikacije, kratkog opisa i zaduženog člana tima može se pronaći u README.md datoteci.

Administrator:

* **Administracija korisnika** – administrator može dodavati nove korisnike, blokirati i deblokirati korisnike
* **Administracija postavki** – administrator može mijenjati postavke koje su definirane unutar sustava poput minimalne količine i minimalne cijene
* **Administracija ponude** – administrator može administrirati ponudu ribe (vrsta, količina, cijena, lokacija) te dodavati fotografije i lokacije ribe
* **Pregled dnevnika –** sustav unutar dnevnika u bazi podataka bilježi sve radnje koje se događaju u aplikaciji. Administrator ima uvid u te radnje

Korisnik:

* **Registracija korisnika –** korisnik se može registrirati preko forme unutar aplikacije gdje unosi osobne podatke (ime, prezime, email adresu, korisničko ime, broj telefona, mjesto, datum rođenja te lozinku i potvrdu lozinke). Nakon unosa podataka, korisniku na uneseni email stiže verifikacijski kod u obliku peteroznamenkastog broja kojeg potom unosi u za to predviđeno polje aplikacije. Slijedi prihvaćanje uvjeta korištenja aplikacije nakon kojeg se korisnik može služiti aplikacijom i urediti vlastiti korisnički profil
* **Prijava korisnika -**  prijava u aplikaciju se odvija preko forme za prijavu koja zahtijeva unos korisničkog imena ili email adrese te pripadne lozinke. Nakon prijave u aplikaciju, korisniku se otvara sučelje aplikacije primjereno njegovoj ulozi (Administrator/Ponuditelj/Kupac). U formi prijave postoji i poveznica „Zaboravljena lozinka“ koja omogućava ponovno postavljanje lozinke u slučaju da je korisnik istu zaboravio
* **Postavke korisničkog računa –** prijavljeni korisnik može unutar aplikacije upravljati svojim korisničkim računom na način da može dodati svoju fotografiju, dodatne podatke te ažurirati postojeće
* **Pregledavanje ponuda –** korisnik može pregledavati, pretraživati, filtrirati i sortirati ponude koje su vidljive unutar aplikacije te su prilagođene mjestu stanovanja korisnika.
* **Ocjenjivanje korisnika** **-** korisnik može dodati ocjenu za ponuditelja nakon što kupi nešto od njega ali i ponuditelj može ocijeniti kupca
* **Rezerviranje ponuda -** korisnik može rezervirati ponudu koju onda ponuditelj može prihvatiti ili odbiti. Ponuditelj određuje vremenski rok, ali postoji predefinirano (default) vrijeme za rezervaciju od 30 minuta
* **Potvrđivanje/odbijanje rezervacije -** ponuditelj mora unutar tih 30 minuta (ili više ako je to ponuditelj odredio) prihvatiti rezervaciju ili ona propada. Isto tako ponuditelj može i odbiti rezervaciju
* **Kontaktiranje –** korisnik može kontaktirati ponuditelja nakon što je odobrena njegova rezervacija za ponudu
* **Dodavanje nove ponude –** korisnik može dodati ponudu u kojoj navodi količinu, cijenu, vrstu ribe, lokaciju te po potrebi opis, dodatnu fotografiju i trajanje rezervacije. Nakon dodavanja ponude može ju naknadno uređivati ili izbrisati
* **Pregled izvršenih rezervacija –** korisnik može pregledavati svoje izvršene rezervacije
* **Preuzimanje ribe –** ponuditelj treba potvrditi preuzimanje ribe kako bi se zatvorila transakcija. Nakon toga se korisnici također mogu međusobno ocjenjivati

Sustav:

* **Povezivanje i čitanje s bazom podataka -** baza podataka sadrži korisničke podatke, podatke o ponudama, ribama, lokacijama, rezervacijama, dnevniku rada, ocjenama, značkama, obavijestima i porukama. Aplikacija se povezuje na bazu podataka u koju upisuje podatke preuzete iz korisničkog sučelja te preuzima podatke potrebne za popunjavanje ponude poput predefiniranih vrsta riba i lokacija, uloga korisnika i sl.
* **Omogućavanje različitih uloga korisnika u aplikaciji (administrator, ponuditelj, kupac) –** kao što je navedeno, aplikacija ima više uloga (administrator, ponuditelj i kupac) koje imaju uvid u postavke i sadržaj aplikacije definirane odgovarajućom ulogom
* **Sustav obavijesti –** sustav može poslati obavijesti korisniku o rezervaciji, pristiglim porukama, prihvaćanju ili odbijanju rezervacije ili ostalim sustavskim obavijestima koje korisnik dobiva unutar aplikacije
* **Sustav znački –** sustav može dodijeliti značke korisnicima ovisno o brojnim pokazateljima (povijest kupnje i prodaje, dobivene ocjene i sl.) te na taj način može nagraditi vjernost kupca i pouzdanost ponuditelja
* **Korisničke upute –** unutar aplikacije se nalaze upute za korištenje aplikacije kojima se pristupa pritiskom tipke F1 ili odabirom opcije za pomoć

2.2.3. Korisnički zahtjevi

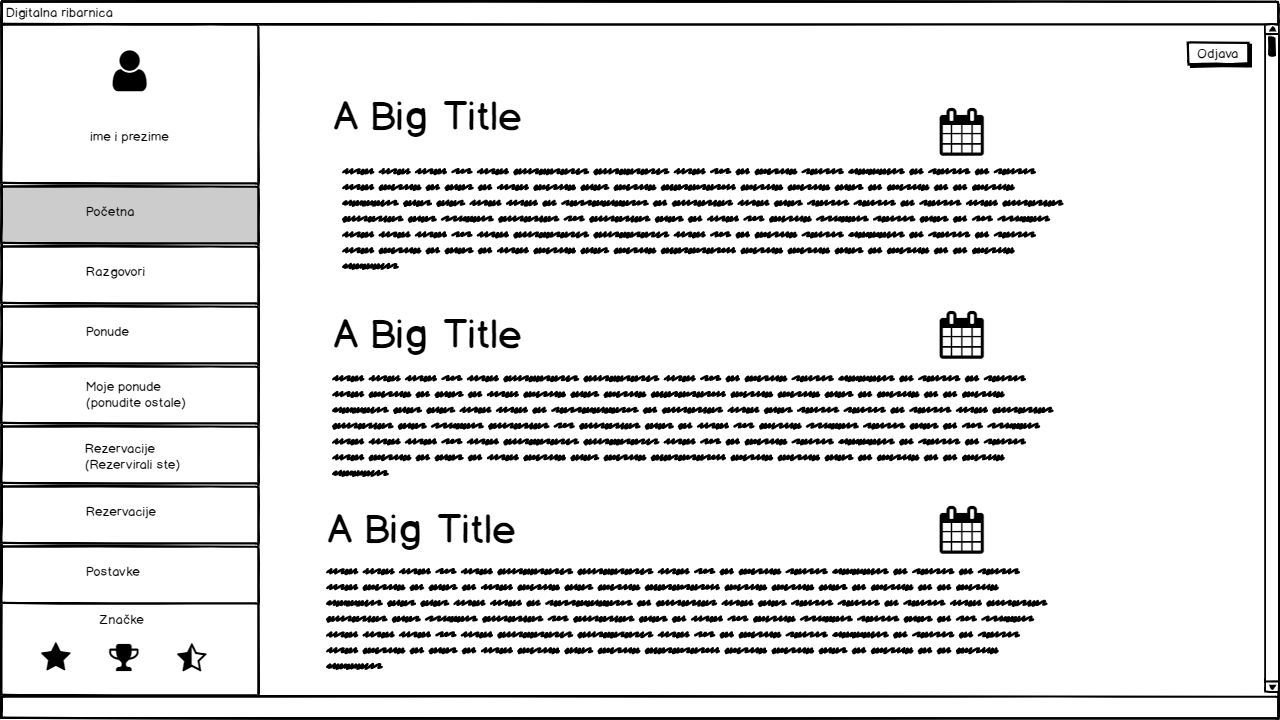
Za korištenje aplikacije Digitalna ribarnica potrebni su računalo i računalna periferija.

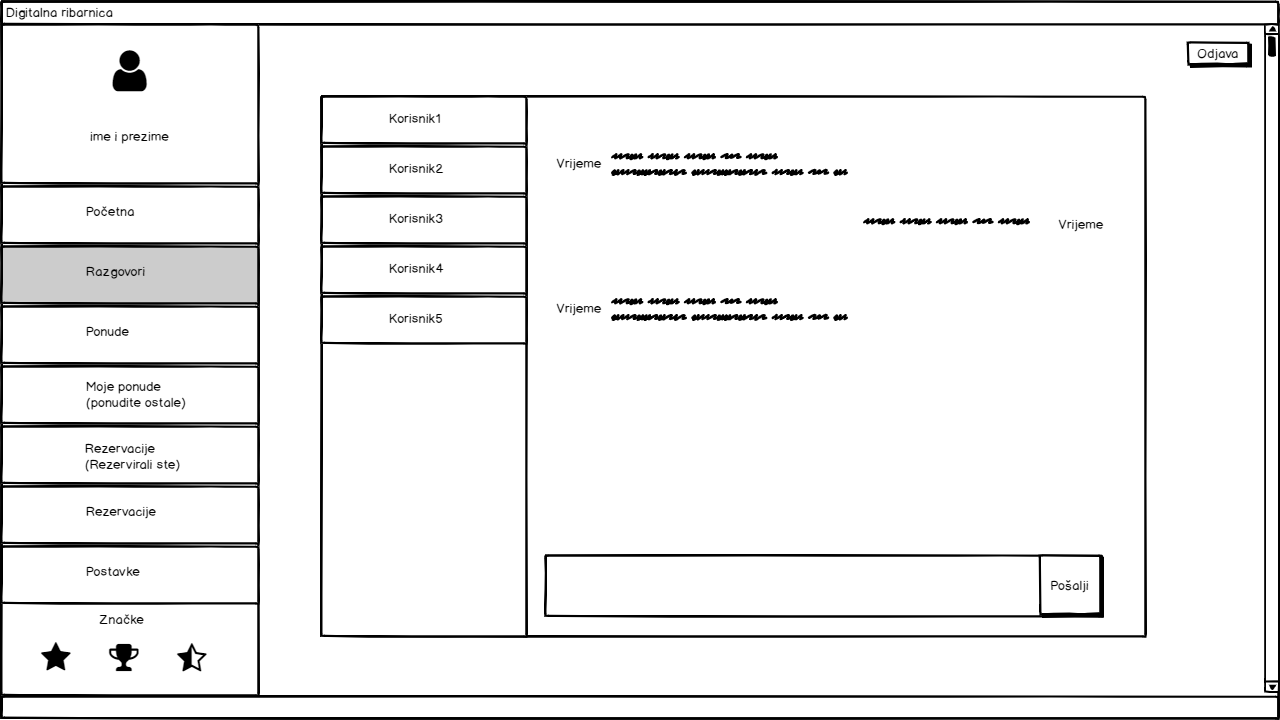
2.2.4. Ograničenja

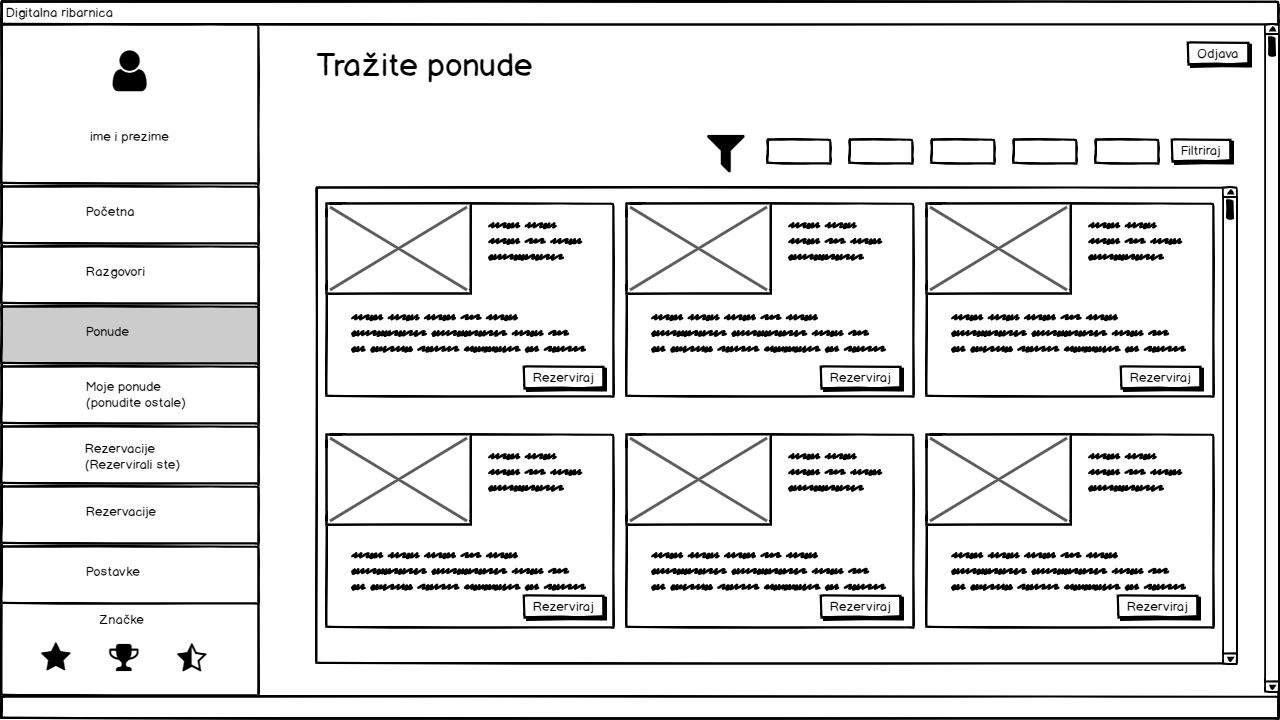
Aplikacija je vezana za Windows tehnologije i razvijena je za Windows operacijske sustave. Za korištenje aplikacije je potrebno imati instaliran .Net 4.5 Framework.

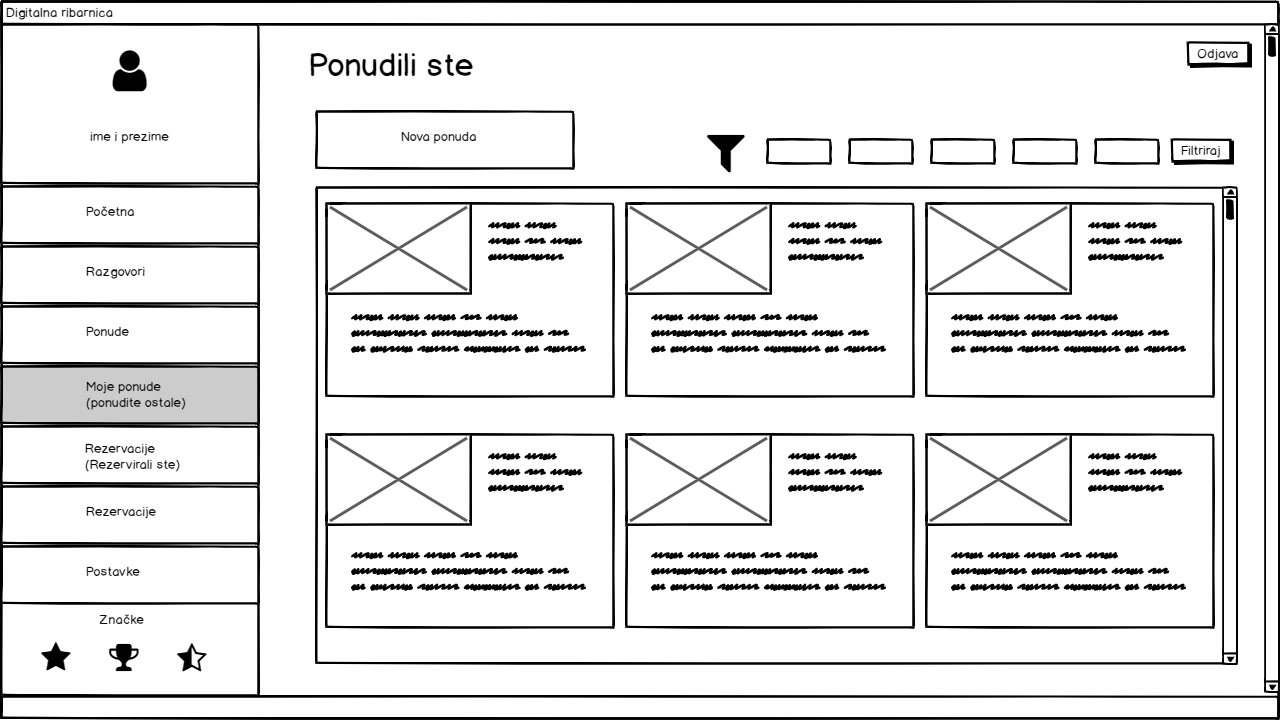
2.2.5. Korisničko sučelje – wireframe

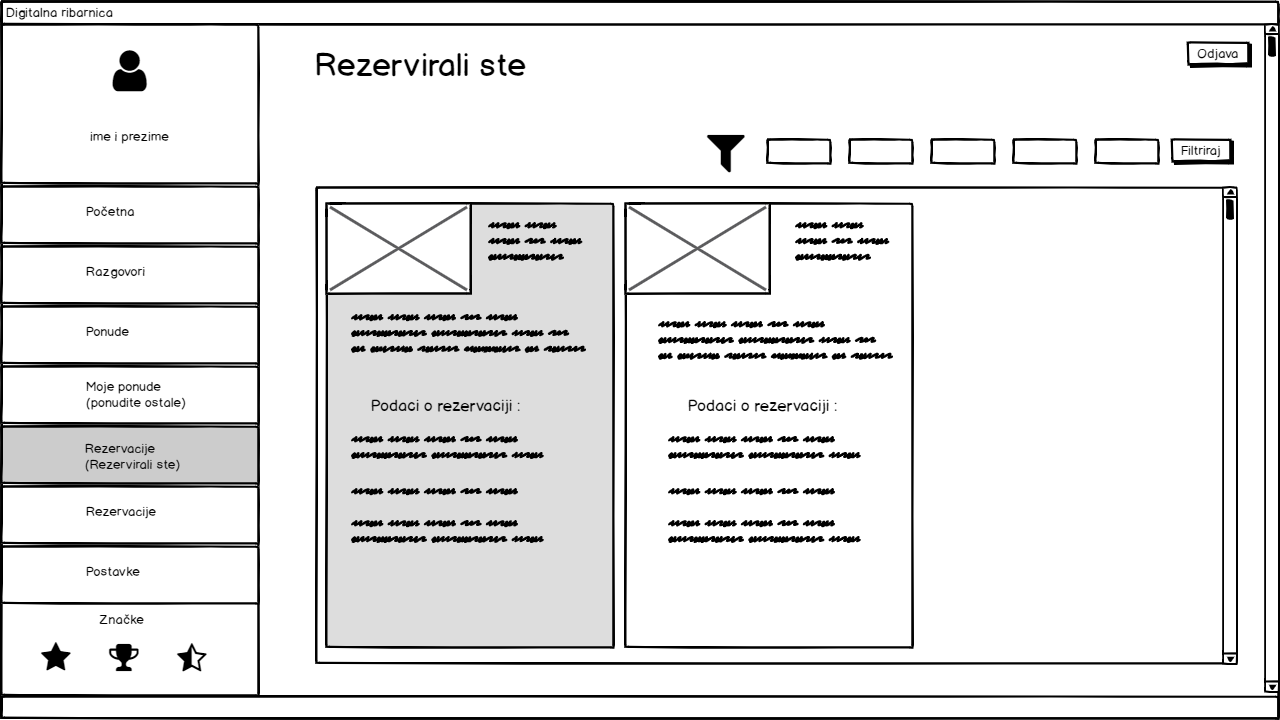
U online alatu Balsamiq izradili smo prijedlog korisničkog sučelja, odnosno wireframe. Sljedeće slike prikazuju nekoliko formi aplikacije – Početna stranica – novosti, Razgovori, Ponude, Moje ponude, Rezervacije, Moje rezervacije, Postavke, Značke.

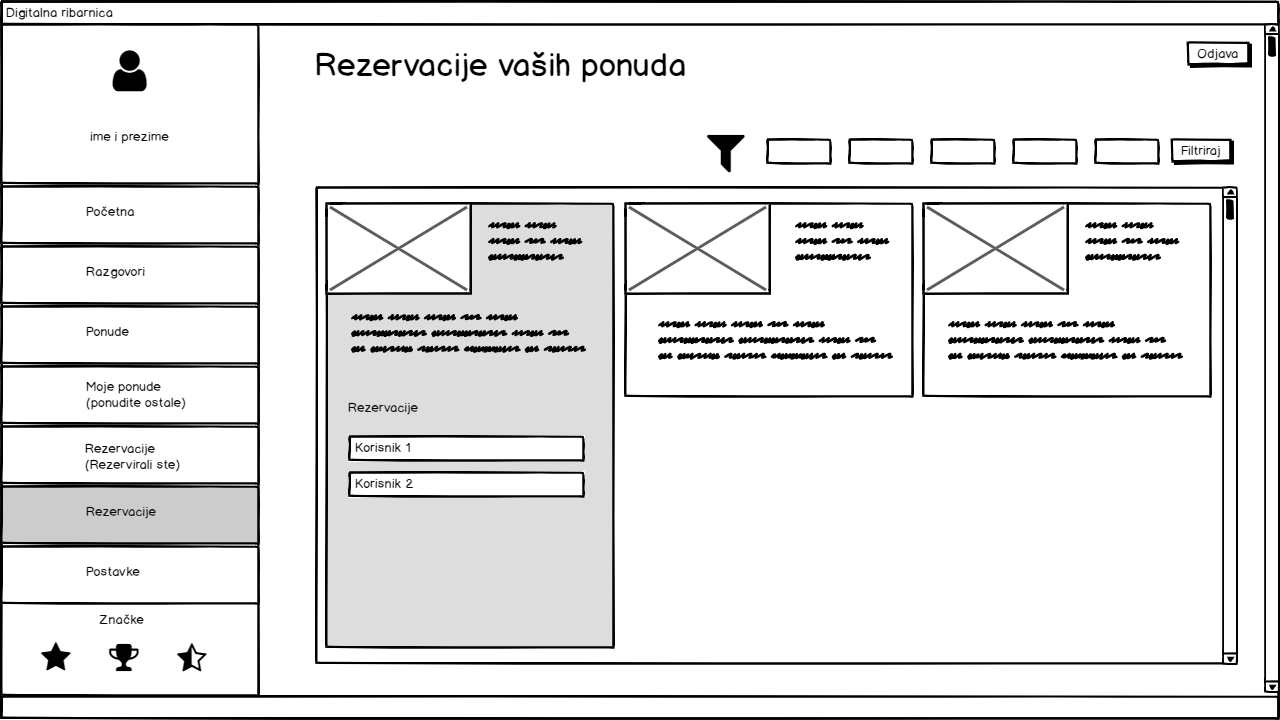


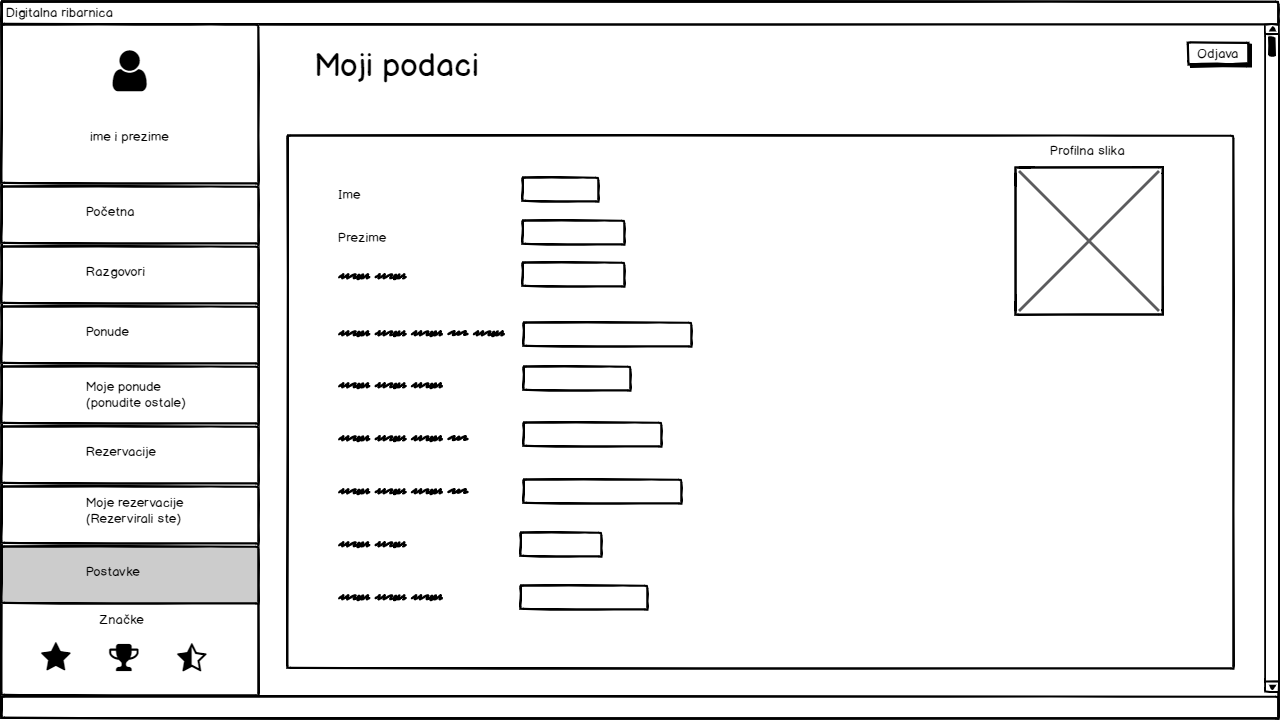


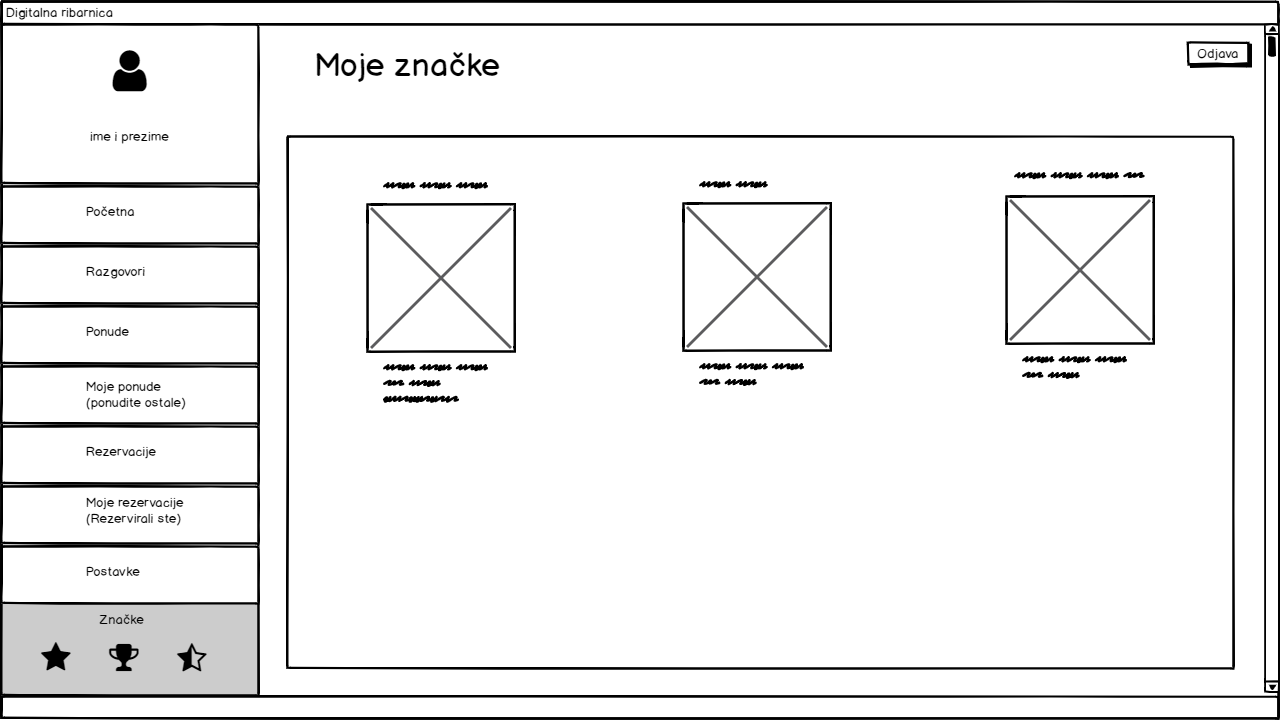










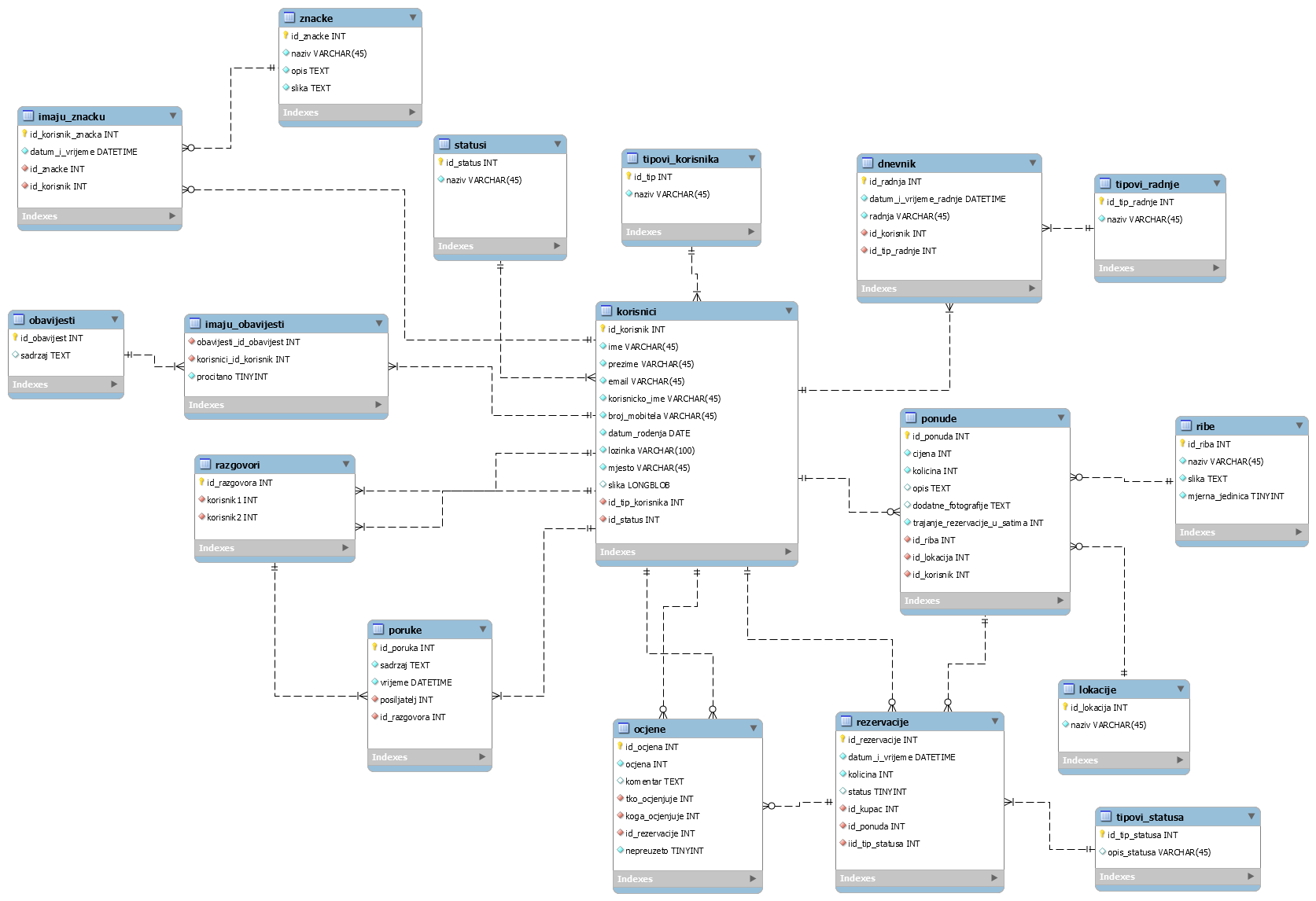


# Specifični zahtjevi

U poglavlju specifični zahtjevi prikazane su prethodno popisane i objašnjene funkcionalnosti aplikacije uz opise i pripadne dijagrame.

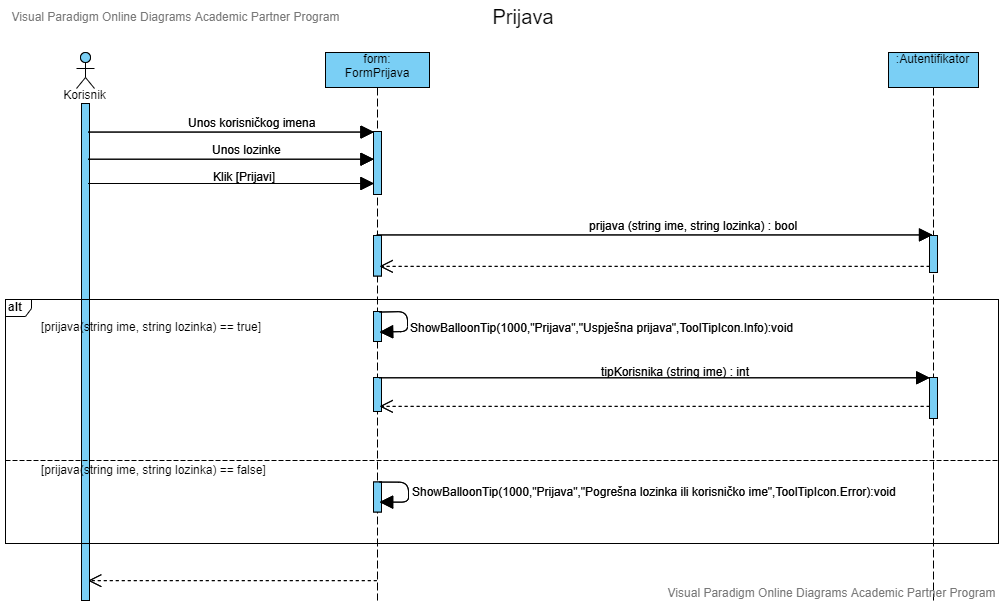
2.3.1. ERA model

ERA model čini ukupno 17 tablica u koje se spremaju i iz koje se čitaju podaci potrebni za rad aplikacije. Tablica korisnici sadrži podatke o svim korisnicima aplikacije: identifikacijska oznaka (id), ime, prezime, email, korisničko ime, broj mobitela, datum rođenja, lozinka, mjesto i slika. Svakom korisniku je pridružen točno jedan tip korisnika koji može biti administrator, kupac ili ponuditelj, dok je jedan tip korisnika pridružen jednom ili više korisnika. Postoji i tablica dnevnik u kojoj se bilježe radnje, datum i vrijeme radnje, identifikacijska oznaka radnje, identifikacijska oznaka korisnika koji je izvršio radnju te identifikacijska oznaka tipa radnje. Tablica *tipovi\_radnje* sadrži identifikacijske oznake i nazive radnji koje su dopuštene u aplikaciji poput dodavanja ponude, registracije, prijave, uređivanja ponude i sl. Svaki korisnik ima jedan status koji može aktivan (trenutno aktivan u aplikaciji), blokiran (ne može se više prijaviti u aplikaciju) ili neaktivan (nije blokiran, trenutno nije aktivan u aplikaciji). Jedan status može odgovarati jednom ili više korisnika. Ranije smo spomenuli sustav znački koji omogućava svakom korisniku osvajanje jedne ili više znački, te jednu značku može osvojiti više korisnika. Vidimo da se ovdje radi o binarnoj vezi M:N, stoga je potrebno uključiti pomoćnu tablicu *imaju\_značku* koja bilježi informacije o datumu i vremenu osvajanje značke, tipu značke koja se dodjeljuje te identifikacijskoj oznaci korisnika kojem se dodjeljuje značka. Tablica znački sadrži identifikacijsku oznaku značke, naziv, opis i sliku značke. Korisnik nakon osvajanja značke odlaskom na svoj korisnički profil ima uvid u sve osvojene značke. Također, ostali korisnici vide značke drugih korisnika. Korisnik može dobivati više obavijesti koji se odnose na obavijesti o zahtjevu za rezervacijom ponude koju dobiva ponuditelj nakon što objavi ponudu i netko je rezervira, zatim obavijesti o potvrđenoj ili odbijenoj rezervaciji koje dobiva kupac kada rezervira ponudu te ponuditelj reagira na rezervaciju. Postoje i sustavske obavijesti koje korisnici mogu dobivati poput dodatnih informacija i sl. Ovdje se također radi o binarnoj vezi M:N jer korisnici mogu dobivati više obavijesti, te jednu obavijest može dobiti više korisnika. Tablica obavijesti sadrži identifikacijsku oznaku te sadržaj obavijesti. Pomoćna tablica *imaju\_obavijesti* sadrži identifikaciju oznaku obavijesti, identifikacijsku oznaku korisnika koji prima obavijest te atribut *procitano* koji je tipa boolean i po defaultu je false, što označava da korisnik nije pročitao obavijest. Kada ju pročita, *procitano* poprimi vrijednost true. Spomenuta je i implementacija komunikacije između korisnika putem poruka. Svaki korisnik može poslati više poruka i primiti više poruka. U tablici poruke nalaze se podaci o identifikacijskoj oznaci poruke, sadržaju, datumu i vremenu slanja te identifikacijskoj oznaci pošiljatelja i razgovora. Jedan korisnik može postaviti više ponuda u kojoj navodi cijenu, količinu, opis, dodatne fotografije, trajanje rezervacije u satima, ribu koju prodaje i lokaciju. U bazi podataka postoje predefinirane lokacije i ribe. Tablica riba sadrži identifikacijske oznake riba, naziv, sliku i mjernu jedinicu koja se odnosi na količinu za prodaju jer se neke ribe prodaju po komadu, a neke po težini i sl. Pojedina riba se može nalaziti u više ponuda, a svaka ponuda ima točno jednu vrstu ribe. Tablica lokacija sadrži identifikacijsku oznaku lokacije i naziv. U ponudi se navodi jedna lokacija, dok se prema jednoj lokaciji može sastaviti više ponuda. Korisnik (kupac) može rezervirati više ponuda te isto tako jednu ponudu može rezervirati više korisnika. Za rezervaciju se navode datum i vrijeme, količina, kupac, identifikacijska oznaka ponude i status. Status rezervacije može biti: čeka se odobrenje ponuditelja, odobrena rezervacija, odbijena rezervacija i rezervacija izvršena do kraja (prodana i preuzeta riba). Status rezervacije odgovara više rezervacija, a svaka rezervacija u određenom trenutku ima jedan tip statusa. Kupac i ponuditelj se mogu međusobno ocjenjivati kada kupac rezervira ponudu ponuditelja. Tablica ocjene sadrži identifikacijsku oznaku ocjene, ocjenu, komentar, identifikacijsku oznaku rezervacije, kupca, ponuditelja te atribut *nepreuzeto*. Atribut ima po defaultu vrijednost true, a mijenja vrijednost u false ako kupac nije preuzeo ribu što ponuditelj unosi u aplikaciju. Kupac može dodijeliti više ocjena, odnosno na temelju svake njegove rezervacije može dodijeliti ocjenu ponuditelju. Isto tako ponuditelj može ocjenjivati svoje kupce.



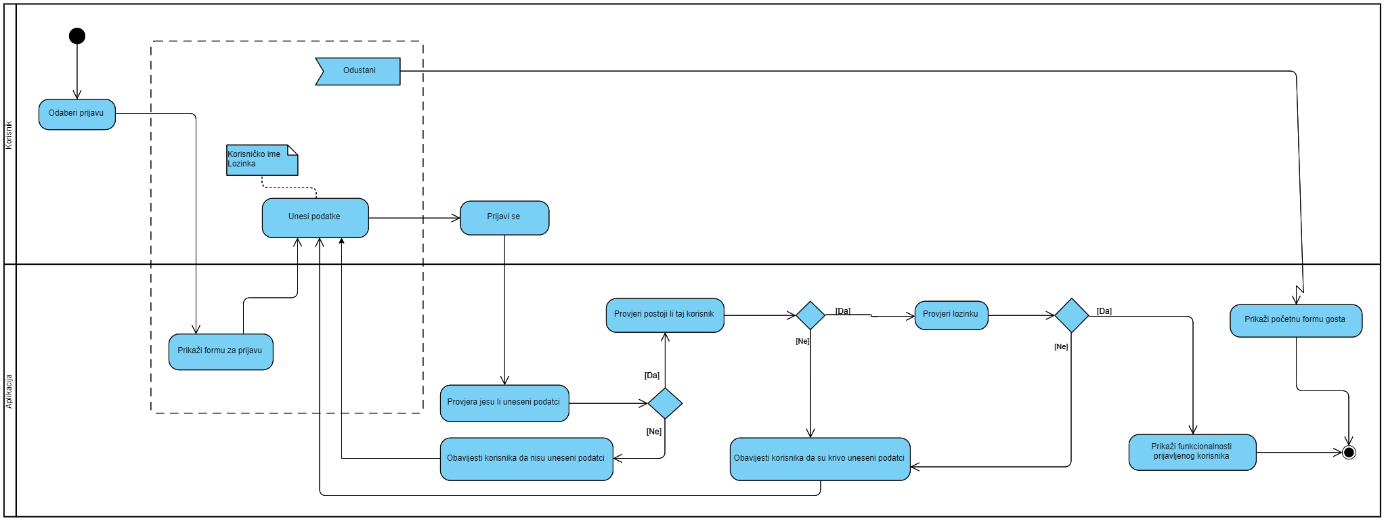
2.3.2. Dijagram slijeda za prijavu u aplikaciju

Korisnik unosi korisničko ime i lozinku te klikne na gumb prijavi. Forma za prijavu FormPrijava poziva funkciju prijava (string ime, string lozinka) tipa bool iz klase Autentifikator. Ukoliko funkcija vrati vrijednost true, FormPrijava poziva funkciju ShowBalloonTip(1000,"Prijava","Uspješna prijava",ToolTipIcon.Info) : void koja služi kako bi prikazali korisniku da se uspješno prijavio. Nakon toga se poziva funkcija tipKorisnika (string ime) : int klase Autentifikator koja nam vraća vrijednost ovisno o tome kakav tip korisnika je taj korisnik koji se prijavljuje. Ukoliko funkcija prijava vrati vrijednost false FormPrijava poziva funkciju ShowBalloonTip(1000,"Prijava","Pogrešna lozinka ili korisničko ime",ToolTipIcon.Error) : void koja služi kako bi prikazali korisniku da je pogriješio pri prijavi.



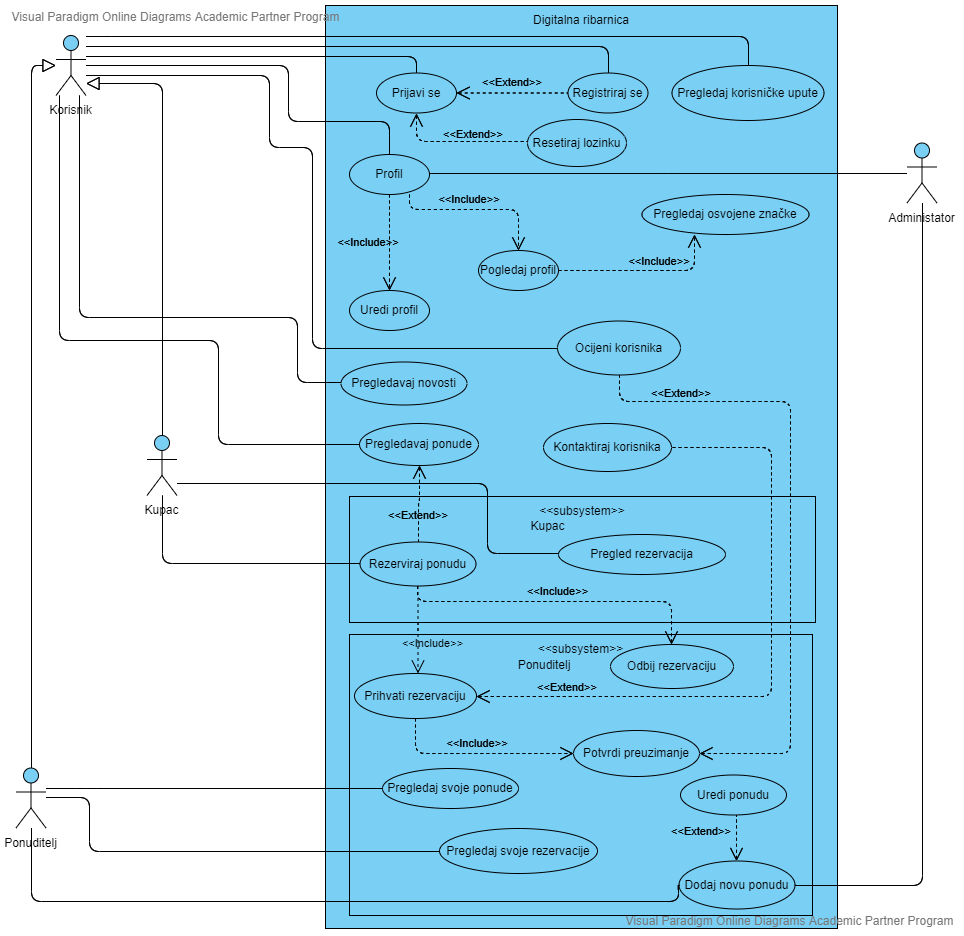
2.3.3. Dijagram aktivnosti za prijavu u aplikaciju

Ovdje je prikazana aktivnost prijavljivanja korisnika. Korisnik se nalazi na početnoj stranici i odabire prijavu. Nakon toga mu se prikazuje forma za prijavu. Korisnik unosi podatke (korisničko ime i lozinka) i pritisne na gumb prijavi se. Podaci se prosljeđuju aplikaciji na provjeru. Prvo se provjerava jesu li podaci uopće uneseni. Ako podatci nisu uneseni korisnika se obavještava da podaci nisu unijeti i ponovo se vraća na korak za unos podataka. Ako je korisnik unio podatke provjerava se postoji li u bazi podataka korisnik s navedenim korisničkim imenom. Ako ne postoji obavještava se korisnika da je krivo unio podatke i vraća ga se na korak za unos podataka. Ukoliko postoji korisnik s tim korisničkim imenom ide se na provjeru lozinke. Ukoliko se lozinka ne poklapa sa korisničkim imenom obavještava se korisnika da su krivo uneseni podatci i vraća ga se na korak za unos podataka. Ukoliko je kombinacija korisnikom imena i lozinke dobra to znači da je korisnik uspješno logiran i prikazuju mu se funkcionalnosti koje su svojstvene tipu korisnika kojem on pripada. U bilo kojem trenutku od prikaza forme za prijavu pa sve do unosa podataka korisnik može odustati od prijave pritiskom na gumb "Odustani" što rezultira prikazom početne forme, ali bez dodatnih funkcionalnosti, dakle početne forme za gosta.



2.3.4. Dijagram slučajeva korištenja za prikaz odnosa sustava i okoline

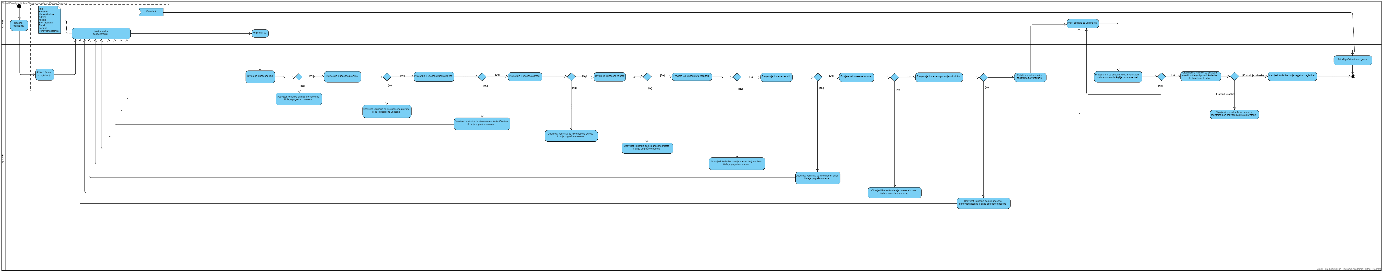
Sljedeća slika prikazuje dijagram slučajeva korištenja koji je izrađen u online alatu Visual Paradigm. Kako postoji više razina apstrakcije za oblikovanje modela slučajeva korištenja, odlučili smo se za konceptualnu razinu apstrakcije koja prikazuje osnovne specifikacije funkcionalnosti sustava i odnosa sa okolinom koje razumiju i korisnici. Dijagram slučajeva korištenja opisuje što sustav radi, s motrišta vanjskog promatrača te nije bitno kako sustav funkcionira iznutra. Slučaj korištenja je priča koja opisuje kako učesnici koriste sustav da bi postigli određene ciljeve ili obavili poslove. Učesnici su korisnik koji može biti ponuditelj ili kupac te administrator. Učesnici se nalaze izvan promatranog sustava i s njim su u međudjelovanju. Sa lijeve strane promatranog sustava se nalaze primarni učenici – korisnik koji može biti ponuditelj ili kupac. Primarni učesnici pokreću slučaj korištenja da bi ostvarili neki svoj cilj ili obavili neki zadatak. Sa desne strane promatranog sustava se nalazi sekundarni učesnik – administrator koji nadzire rad slučaja korištenja ili obavlja nešto nakon što ga slučaj korištenja pozove. Korisnik se može prijaviti u aplikaciju te ako nema kreiran korisnički račun, može se registrirati ili otvaranjem aplikacije može odabrati opciju registracije. Pritiskom tipke F1 ili odabirom opcije Help, korisnik može pročitati korisničke upute kako bi savladao korištenje aplikacije i dobio dodatne informacije o aplikaciji. Otvaranjem korisničkog profila korisnik ima opcije uredi profil i pogledaj profil. Prilikom pregledavanja profila, ima uvid u osvojene značke. U aplikaciji može pregledavati novosti i ponude. Za rezervaciju ponude potrebna je uloga kupca koji osim rezervacije ponude ima i uvid u povijest narudžbi. Kada kupac rezervira ponudu, ponuditelj je može prihvatiti ili odbiti. Prihvaćanjem ponude mora navesti je li kupac preuzeo ribu kako bi transakcija bila završena. Nakon preuzimanja ribe, korisnici (kupac i ponuditelj) se mogu međusobno ocjenjivati. Ponuditelj može dodavati nove ponude te ih po potrebi uređivati ili izbrisati. Također ima uvid u povijest svojih ponuda. Administrator može administrirati korisnike na način da ih može blokirati ili deblokirati. Osim ponuditelja, administrator također može administrirati ponude.



2.3.5. Dijagram aktivnosti za registraciju korisnika

[Nikola Muše]

Na slici je prikazana aktivnost registracije korisnika. Na početnoj stranici se nalazi opcija registracije čijim klikom se otvara forma za registraciju koja zahtjeva unos imena, prezimena, korisničkog imena, adrese, mjesta stanovanja, broja telefona, email-a, lozinke i ponovljene lozinke. Pritiskom gumba „Registriraj se“, provjerava se jesu li unesene sve vrijednosti. Ako neka od vrijednosti nije unesena, aplikacija šalje obavijest korisniku kako nije unio pojedinu vrijednost ili ju je pogrešno unio te korisnik mora ispraviti pogrešno unesenu vrijednost ili unijeti ispravnu vrijednost. Nakon što su svi podaci ispravno uneseni, korisnik može pritiskom gumba „Registriraj se“ primiti na uneseni email verifikacijski kod u obliku peteroznamenkastog broja kojeg treba unijeti u za to predviđeno polje. Ukoliko korisnik ne unese verifikacijski kod ili unese krivi, aplikacija ga obavještava kako je unio neispravni verifikacijski kod ili nije uopće unio kod. Ispravnim unosom verifikacijskog koda, korisniku se otvara forma na kojoj su prikazani uvjeti korištenja koje korisnik može prihvatiti ili odbiti. Odbijanjem uvjeta korištenja onemogućuje se registracija, a prihvaćanjem završava postupak registracije. Korisnik može odustati od registracije u trenutku popunjavanja podataka u formi za registraciju. Odustajanjem od registracije korisniku se prikaže početna forma gosta.



2.3.6. Dijagram klasa za projekt Digitalna ribarnica

[Božo Kvesić, Anabela Pranjić]

Na dijagramu klasa možemo vidjeti trenutno implementirane klase te njihove odnose. Kako daljnja implementacija bude tekla, tako ćemo proširivati i dijagram klasa, odnosno kako budemo dodavali funkcionalnosti. Trenutno se ovdje nalaze svi commitovi koda do 2.5.2020. Danas kasno je objavljen jedan commit koji će biti nadodan na dijagram klasa 4.5.2020. ili 5.5.2020. Ovdje se trenutno nalazi funcionalnost prijave i registracije te njihovi odnosi s početnom formom. FormPocetna sadrži buttone koji vode na pojedine funkcionalnosti te se sve te funkcionalnosti prikazuju na početnoj formi. Započet ćemo s prijavom koja je puna jednostavnija od registracije. Prijava sadrži jedan autentifikator povezan slabom kompozicijom iz razloga ako iz nekog razloga prijava ne bude radila onda se ne može dogoditi da i autentifikator ne bude dostupan. Autentifikator sadrži listu korisnika te sadrži metode kojim provjeravamo validaciju korisničkog unosa. Korisnik sadrži property-a koja će kasnije biti potrebno proširiti iz razloga što funkcionalnost registracije zahtjeva brojne podatke o korisniku koje trenutno još uvijek nigdje ne spremamo. Funkcionalnost registracije kreće pritiskom na button „Registracija“ te se nakon toga pojavljuje forma registracije koja zahtjeva od korisnika unos podataka. Autentifikacija registracije je zadužena za validaciju tih podataka te u slučaju određenih nevalidnih podataka baca prijavaException (u slučaju da ne valja ili postoji korisničko ime) ili registrationException (za sva ostala polja).Nakon uspješnog unosa podataka korisniku se na mail šalje verification code kojeg mora unijeti unutar 15 minuta te nakon unosa toga koga mora prihvatiti Terms and service kako bi se uspjšno registrirao. Iz toga razloga smo stavili slabu agregaciju izmedu Code i VerificationCode jer ukoliko iz nekog razloga Verification code forma ne bude radila, Code klasa će moći biti pozivana u ostalim funkcionalnostima, dok recimo jaku agregaciju smo stavili između VerificationCode forme i Registracije te forme Terms and Service i forme Registacije iz razloga što će one biti u potpunosti nedostupne ukoliko funkcionalnost registracije ne bude radila. Klasa autentifikacijaRegistracije.cs je povezana slabom agregacijom jer će ona biti dostupna drugim funkcionalnostima (iako za sada to neće biti potrebno). Postoji mogućnost od promjena, ali za sada smo raspravili te mislimo kako je ovo u redu za trenutno implementaciju. Spremni smo i očekujemo brojne promjene koje će kasnije dovesti do dijagrama klasa koji prati u potpunosti naše funkcionalnosti te rad same aplikacije.

