

**SVEUČILIŠTE U ZADRU**

**STUDIJ INFORMACIJSKIH TEHNOLOGIJA**

Andrej Markanjević

**ZAVRŠNI PROJEKT**

**Dokumentacija projekta**

Zadar, 2024.

**SVEUČILIŠTE U ZADRU**

**STUDIJ INFORMACIJSKIH TEHNOLOGIJA**

Andrej Markanjević

**ZAVRŠNI PROJEKT**

**Dokumentacija projekta**

Student: Andrej Markanjević

Status studenta: redovan

Kolegij, godina: Napredno objektno programiranje , II. Godina

Mentor: izv. prof. dr. sc. Ante Panjkota

Zadar, 2024.

## Sadržaj

1.	Opis problema .....	1
2.	Opis načina rješenja problema → konceptualni model predloženog rješenja .....	1
3.	Opis wireframe-a .....	2
4.	Opis i prikaz dijagrama klasa .....	6
5.	ERD dijagram .....	8
6.	Opis korištenih principa razvoja uz projekt .....	9
7.	Vanjske biblioteke .....	10
8.	Git aciklički graf .....	10

## 1. Opis problema

Problem koji projekt rješava je kreiranje jednostavnog web shopa za namirnice s funkcionalnošću korisničkih računa i pohranjivanja narudžbi u bazu podataka. Korištenje MVC (Model-View-Controller) arhitekture omogućuje odvajanje korisničkog sučelja, poslovne logike i pristupa podacima, čime se poboljšava održivost i modularnost aplikacije. S obzirom na povećanje broja online kupovina i potrebe za jednostavnim rješenjima koja omogućuju pohranu narudžbi, primarni cilj je stvoriti aplikaciju koja korisnicima omogućava registraciju korisničkih računa, prijavu u sustav i odabir namirnica koje žele kupiti. Iako kupnja proizvoda nije izvedena u stvarnom vremenu, podaci o odabranim proizvodima spremaju se u bazu podataka, pružajući osnovu za kasniju analizu i obradu.

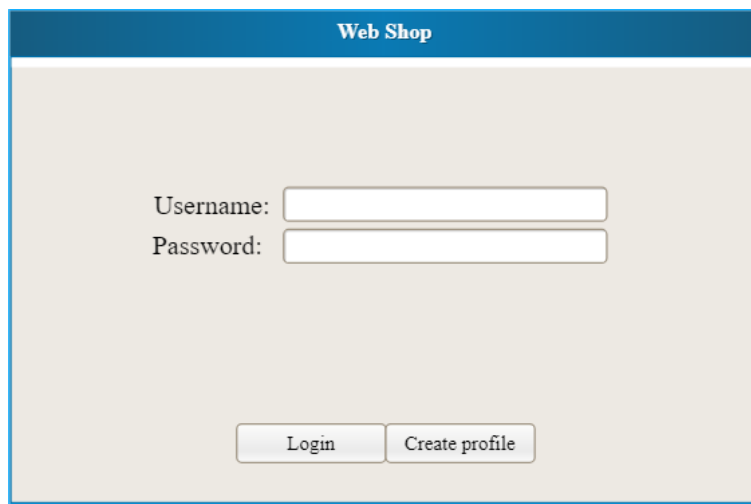
## 2. Opis načina rješenja problema → konceptualni model predloženog rješenja

Kako bi se problem riješio, predloženo je rješenje koje se temelji na sljedećim ključnim komponentama. **Model** predstavlja podatke aplikacije i poslovnu logiku povezanu s tim podacima. U ovom slučaju, to uključuje korisničke račune i namirnice dostupne u web shopu. Implementiran je pristup bazi podataka korištenjem odgovarajućih DAO (Data Access Object) klasa za spremanje i dohvaćanje podataka o korisnicima i njihovim narudžbama. Baza podataka može biti relacijska (npr. MySQL), a model osigurava sve potrebne operacije kao što su CRUD (Create, Read, Update, Delete) za namirnice i korisničke račune. **Pogled (View)** je grafičko sučelje putem kojeg korisnici komuniciraju s aplikacijom. Aplikacija koristi jednostavne GUI (Graphical User Interface) komponente, kao što su JFrame-ovi i JPanel-ovi za prikaz informacija korisniku. Korištenjem jednostavnih formi omogućena je registracija i prijava korisnika, pregled i odabir namirnica, prikazivanje potvrda o narudžbi., sve komponente su jasno odvojene kako bi se moglo lako proširiti aplikaciju u budućnosti. **Kontroler (Controller)** povezuje model i pogled te upravlja interakcijama korisnika. Kada korisnik unese podatke u formu (npr. prijava ili odabir proizvoda), kontroler prima te podatke, obrađuje ih i šalje

odgovarajuće zahtjeve modelu za pristup bazi podataka ili ažuriranje podataka. Kontroler osigurava da su svi podaci pravilni prije nego se proslijede modelu te koordinira prikazivanje rezultata u sučelju. Aplikacija koristi relacijsku bazu podataka za pohranu informacija o korisnicima i narudžbama. Svaki korisnik ima jedinstveni ID, a narudžbe su povezane s korisnicima pomoću odgovarajućih vanjskih ključeva. Podaci o proizvodima uključuju osnovne informacije poput naziva, cijene i dostupnosti. Maven je korišten za upravljanje ovisnostima i izgradnju projekta. Korištenje Maven-a osigurava da se sve potrebne biblioteke i vanjske komponente jednostavno dodaju u projekt. U ovom slučaju, Maven je korišten za dodavanje JDBC driver-a za povezivanje s bazom podataka i eventualno korištenje drugih alata kao što su biblioteke za GUI (npr. Swing ili JavaFX). Ovo rješenje omogućuje lako proširivanje sustava, jednostavno održavanje i modularnost. MVC arhitektura osigurava jasnu podjelu odgovornosti, čime se olakšava budući razvoj ili promjene u bilo kojem dijelu sustava (npr. promjena baze podataka ili sučelja).

### **3. Opis wireframe-a**

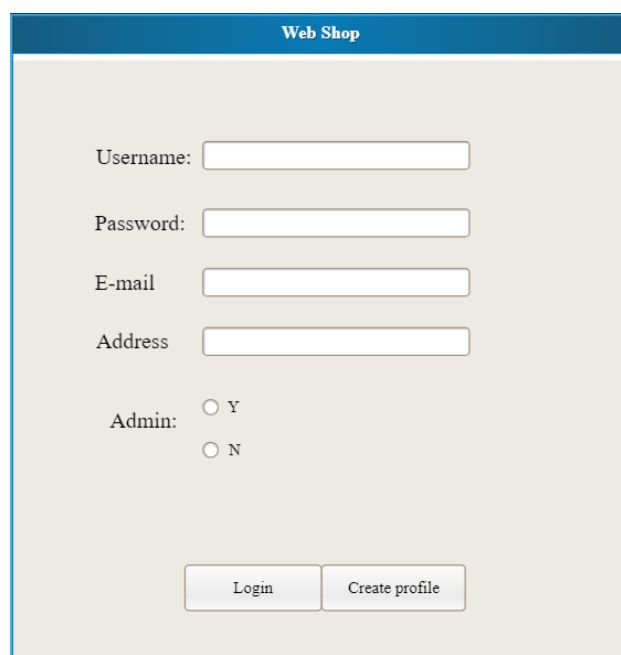
Pri paljenju aplikacije otvara se Login frame (Slika 1). Ova stranica je početni ekran za prijavu korisnika u web shop. Ima dva polja za unos podataka: jedno za korisničko ime i jedno za lozinku. Ispod polja za unos nalaze se dva gumba: Login za prijavu i Create profile za odlazak na stranicu za stvaranje profila. Ova stranica omogućava korisniku da se prijavi ili ga usmjeri na kreiranje računa.



The image shows a login frame for a 'Web Shop'. It has a blue header bar with the text 'Web Shop'. Below the header, there are two input fields: 'Username:' and 'Password:'. At the bottom, there are two buttons: 'Login' and 'Create profile'.

*Slika 1. Login frame*

Stranica za kreiranje korisnika (Slika 2) sadrži polja za unos korisničkog imena, lozinke, e-mail adrese i adrese korisnika. Također, tu su dvije opcije s radio gumbima koje pitaju je li korisnik admin (da ili ne). Na dnu su opet dva gumba: Back to login i Create profile, u ovom slučaju Create profile radi novi profil ako već ne postoji, a Back to login gumb nas vraća na početnu stranicu.



The image shows a 'Create user' frame for a 'Web Shop'. It has a blue header bar with the text 'Web Shop'. Below the header, there are four input fields: 'Username:', 'Password:', 'E-mail', and 'Address'. Below these fields, there are two radio buttons for 'Admin:' with options 'Y' and 'N'. At the bottom, there are two buttons: 'Login' and 'Create profile'.

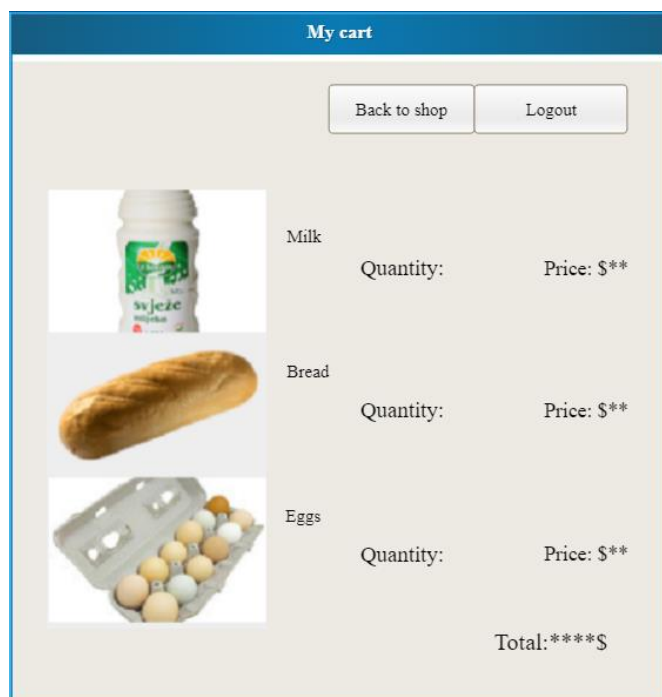
*Slika 2. Create user frame*

Na slici 3 se vidi popis proizvoda u web shopu. U primjeru su prikazani artikli poput mlijeka, kruha i jaja sa slikama svakog proizvoda. Sa strane svakog proizvoda nalazi se izbornik za odabir količine te gumb Add to cart za dodavanje odabranih proizvoda u košaricu. Na vrhu stranice nalaze se tri gumba: My profile, My cart i Logout, koji omogućavaju pregled korisničkog profila, košarice i odjavu.



*Slika 3. Shop frame*

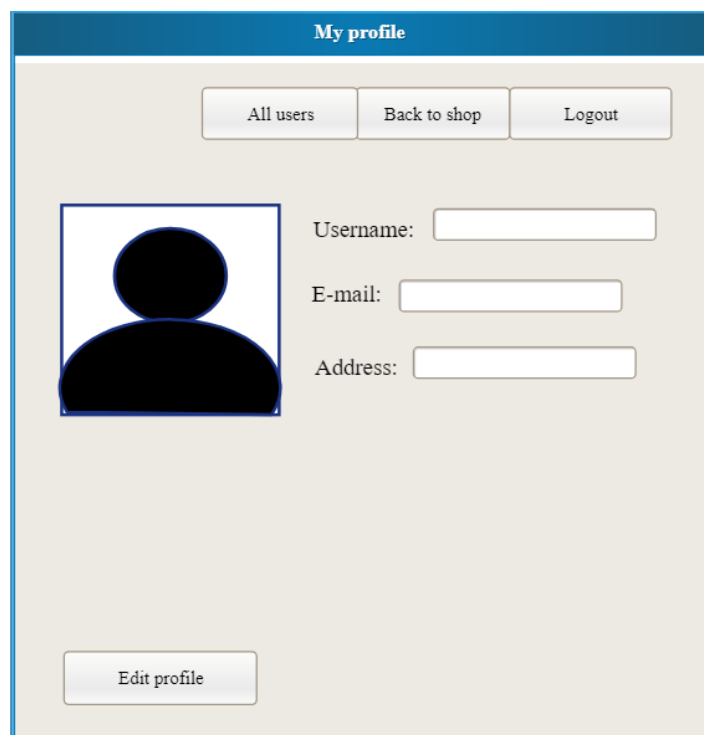
Slika 4 prikazuje sadržaj korisnikove košarice. Prikazani su artikli s informacijama o nazivu, količini i cijeni proizvoda. Na dnu se prikazuje ukupna cijena. Na vrhu stranice nalaze se gumbi Back to shop i Logout, koji omogućavaju povratak u trgovinu i odjavu.



*Slika 4. User cart frame*

Na slici 5 je profil korisnika, gdje se prikazuju informacije poput korisničkog imena, e-mail adrese i adrese. S lijeve strane nalazi se avatar ili simbolična slika korisnika. Na vrhu stranice su tri gumba: All users (prikaz svih korisnika), Back to shop (povratak na trgovinu), i Logout (odjava). Na dnu se nalazi gumb Edit profile za uređivanje profila korisnika.

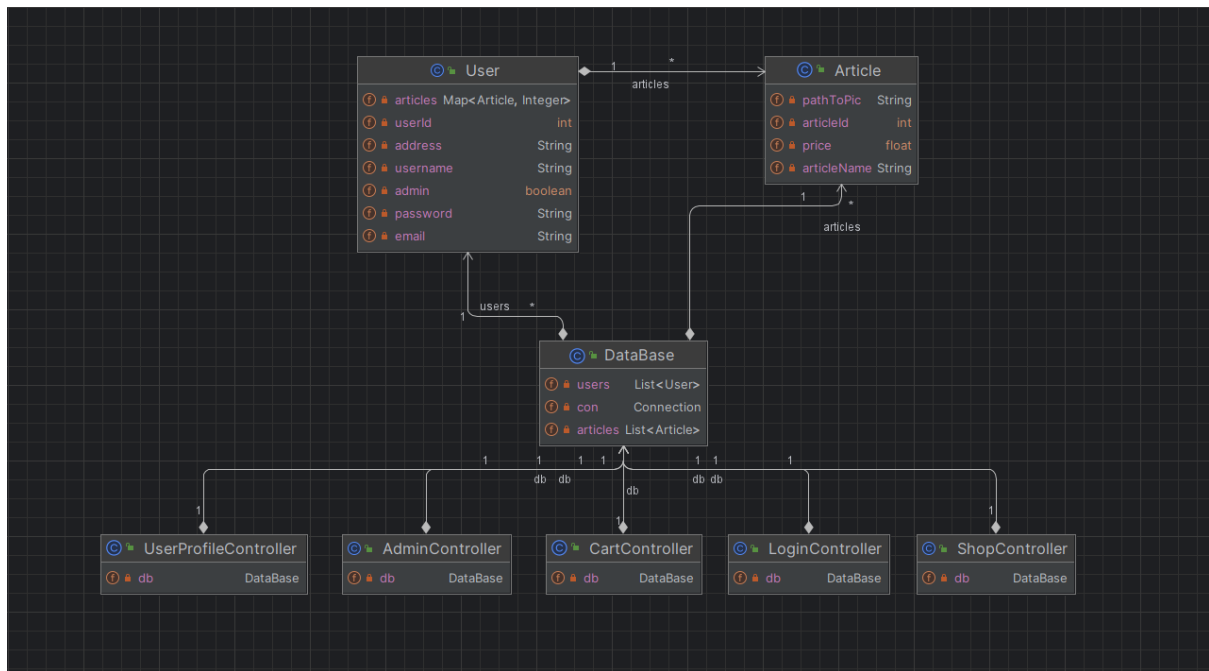


The image shows a web interface for a user profile. At the top, there is a blue header bar with the text "My profile". Below this, there are three buttons: "All users", "Back to shop", and "Logout". In the center, there is a placeholder for a user profile picture, represented by a black silhouette of a person's head and shoulders. To the right of the profile picture, there are three input fields labeled "Username:", "E-mail:", and "Address:". At the bottom left, there is a button labeled "Edit profile".

*Slika 5. User profile frame*

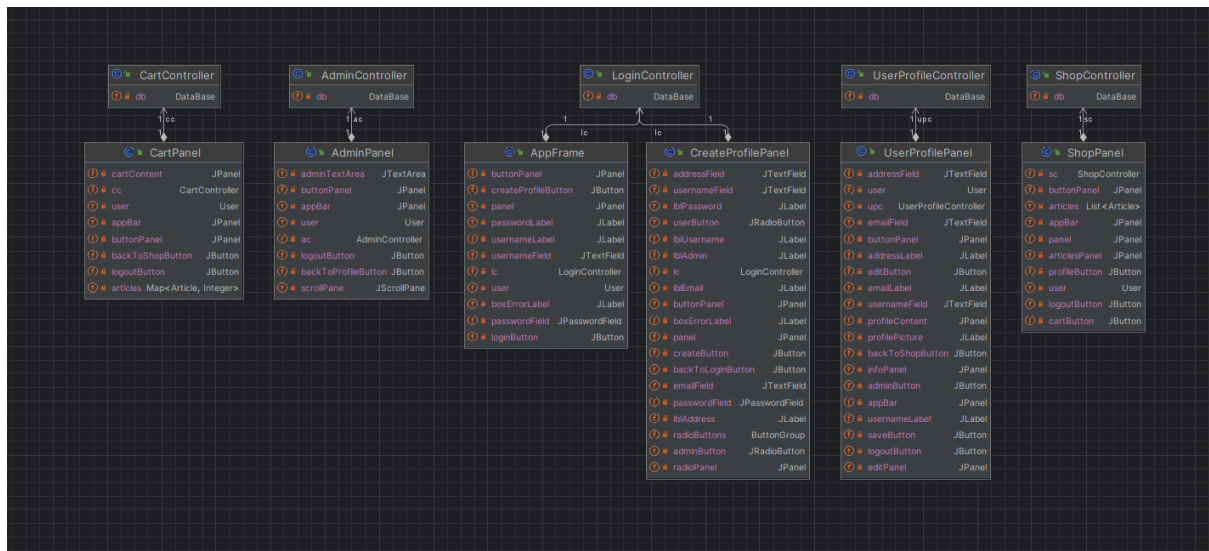
#### **4. Opis i prikaz dijagrama klasa**

Na dijagramu klasa (Slika 6) prikazani su odnosi među klasama koje upravljaju korisnicima, artiklima i bazom podataka, odnosno odnosi klasa modela i klasa controllera. Klasa User sadrži podatke o korisnicima, uključujući njihove artikle, dok je povezana s klasom Article, koja predstavlja pojedinačne artikle s atributima poput naziva, cijene i putanje do slike. Klasa DataBase upravlja popisom korisnika i artikala te održava vezu s bazom podataka. Svaki od kontrolera (UserProfileController, AdminController, CartController, LoginController, ShopController) povezan je s bazom podataka, omogućujući im pristup korisnicima i artiklima te upravljanje specifičnim funkcijama aplikacije, poput profila korisnika, košarice i trgovine. Relacije među klasama su strukturirane tako da jedan korisnik može imati više artikala, a svi kontroleri dijele jednu instancu baze podataka.



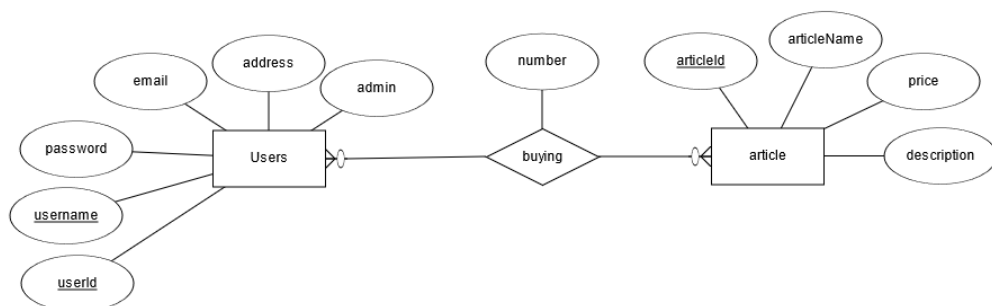
Slika 6. Model - Controller class diagram

Na dijagramu (Slika 7) su prikazani kontroleri koji upravljaju različitim panelima korisničkog sučelja. CartController upravlja CartPanelom, koji omogućuje korisnicima pregled artikala u košarici i navigaciju. AdminController kontrolira AdminPanel, koji pruža administrativno sučelje za pregled i uređivanje korisnika i artikala. LoginController povezan je s klasama AppFrame i CreateProfilePanel, koje upravljaju prijavom i kreiranjem novih korisničkih profila. UserProfileController koristi UserProfilePanel za pregled i uređivanje korisničkih profila, dok ShopController upravlja ShopPanelom, koji prikazuje artikle dostupne za kupnju. Svaki kontroler komunicira s bazom podataka kako bi podržao različite funkcionalnosti aplikacije i omogućio interakciju s korisnicima.



Slika 7. Controller - Views diagram class

## 5. ERD dijagram



Slika 8. ERD diagram of the database

Na temelju prikazanog ERD dijagrama, mogu se vidjeti dva glavna entiteta: Users i Article. Entitet Users ima attribute za ID, korisničko ime, e-mail, adresu i za administratora. Entitet Article također ima attribute za ID i naziv artikla, te opis artikla i njegova cijena. Između entiteta Users i Article postoji veza nazvana buying, koja predstavlja proces kupovine članka od strane korisnika. Ova veza ima atribut number koji označava broj kupljenih artikala. Kardinalnost veze pokazuje da jedan korisnik može kupiti više različitih artikala, a isto tako, jedan članak

može biti kupljen od strane više različitih korisnika (many-to-many), ta kardinalnost je i razlog zašto tablica buying postoji.

## **6. Opis korištenih principa razvoja uz projekt**

U ovome projektu se primjenjuje MVC arhitektura, ona ovaj projekt dijeli u tri jadne komponente: Model, Pogled (View) i Kontroler (Controller). Ovaj obrazac omogućio je modularnost i razdvajanje odgovornosti, čineći aplikaciju preglednom i lakom za održavanje, posebno u kontekstu funkcionalnosti web shopa s korisničkim računima i pohranom narudžbi u bazu podataka. Model je ključan za rad s bazom podataka, jer upravlja pohranom korisničkih računa, informacijama o narudžbama i proizvodima. U ovom slučaju, DAO (Data Access Object) klase povezane s bazom podataka omogućile su stvaranje, dohvaćanje, ažuriranje i brisanje podataka. Time je postignuto da su svi podaci centralizirani i izolirani od prikaza, što omogućuje lakšu promjenu logike ili strukture baze bez potrebe za izmjenom sučelja. Nedostatak ovog pristupa mogao bi biti složenost Modela s rastom poslovne logike, osobito u sustavu koji se planira proširivati. Pogled (View) u projektu predstavlja grafičko sučelje putem kojeg korisnici komuniciraju s aplikacijom, npr. koristeći forme za prijavu, registraciju i pregled namirnica. Ovo jasno odvajanje omogućilo je jednostavno dodavanje novih elemenata sučelja, poput dodatnih prozora za pregled narudžbi. Velika prednost je to što su sve promjene u sučelju moguće bez dodirivanja poslovne logike, a nedostatak je što pogled ovisi o kontroleru kako bi funkcionirao, jer samostalno ne može raditi s podacima ili reagirati na korisničke akcije. Kontroler u aplikaciji upravlja interakcijom između pogleda i modela. Kada korisnik unese podatke, kao što je odabir proizvoda ili prijava u sustav, kontroler obrađuje taj unos i putem modela ažurira bazu podataka te odgovarajuće ažurira prikaz u Pogled-u. Prednost je što je logika koja upravlja interakcijom korisnika centralizirana, što omogućuje bolje testiranje i proširivanje sustava. Međutim, kako se broj funkcionalnosti povećava, postoji rizik da Kontroler postane preopterećen i teško ga je održavati. Prednosti MVC-a u ovome projektu su jasna podjela odgovornosti. Na primjer, odvajanje poslovne logike (kao što su pravila pohrane i manipulacije podacima) u model omogućilo je da pogled i kontroler budu jednostavni i čisti, usmjereni na sučelje i korisničke akcije.

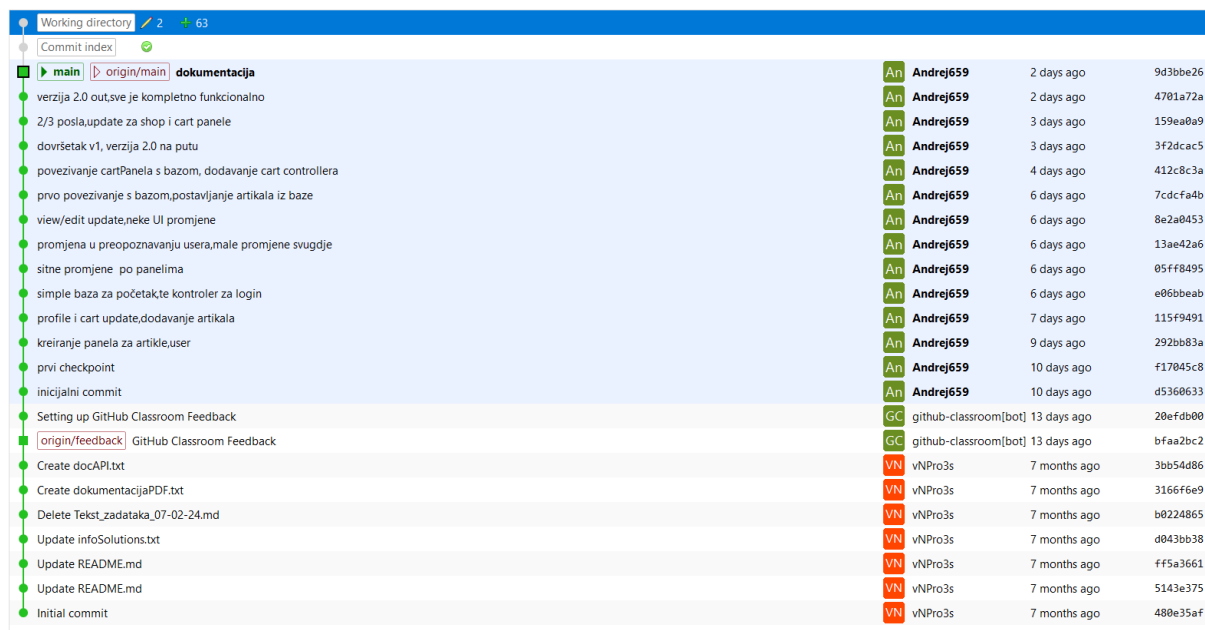
## 7. Vanjske biblioteke

Od vanjskih biblioteka u projektu je korišten samo JDBC (Java Database Connectivity). On omogućuje aplikaciji da šalje SQL upite i upravlja bazom podataka. JDBC driver se koristi za specifičnu bazu podataka, npr. MySQL JDBC driver ili PostgreSQL JDBC driver. U ovom projektu je korišten MySQL JDBC Driver jer je stvorena MySQL baza podataka.

Lokacija preuzimanja: <https://dev.mysql.com/downloads/connector/j/>

Dokumentacija biblioteke: <https://dev.mysql.com/doc/connector-j/en/>

## 8. Git aciklički graf



Slika 9. Git aciklički graf