

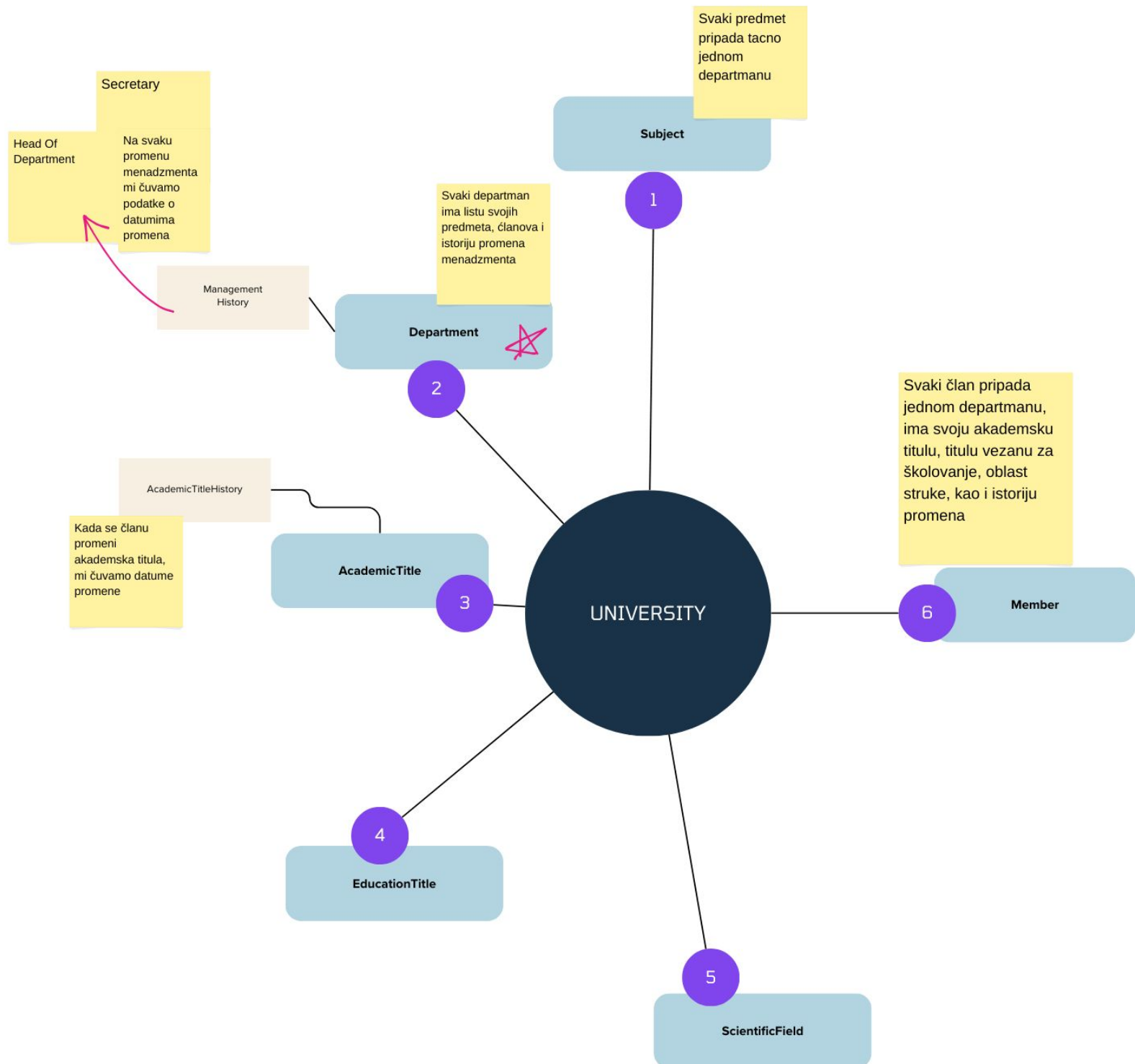


UNIVERSITY SIMULATION

DARKO SELAKOVIĆ

MAJ 2024

UNIVERSITY SIMULATOR KONCEPTI



DIJAGRAM KLASA

darkoo59/university-simulation



1

Contributor



7

Issues



0

Stars



0

Forks



university-simulation/src/main/resources/images/classDiagram.pn...

Contribute to darkoo59/university-simulation development by creating an account on GitHub.



GitHub

O APLIKACIJI

University simulator je aplikacija dizajnirana za sveobuhvatno upravljanje univerzitetskim operacijama. Ova simulaciona aplikacija omogućava korisnicima da nadgledaju različite aspekte univerzitetskog menadžmenta, uključujući departmane, predmete, članove fakulteta, njihove akademske titule, naučne oblasti i obrazovne kvalifikacije.

Aplikacija omogućava korisnicima da definišu i organizuju predmete ili kurseve koje univerzitet nudi, zajedno sa detaljima kao što su departmani kojima predmeti pripadaju ili ESPB bodovi koje pojedinačni predmeti obuhvataju.

Korisnici pored toga što mogu kreirati nove departmane, mogu istim i dodeljivati i direktore, kao i sekretare, pri čemu se za svaku promenu u menadžmentu prate promene. Departmani čuvaju i podatke o svim predmetima i članovima koji mu pripadaju.

Članovi departmana pored svojih osnovnih podataka, čuvaju i podatke o departmanu kojem pripadaju, kao i o akademskoj tituli, zvanju i naučnoj oblasti kojoj pripadaju. Prilikom svake promene akademske titule u sistemu se obrađuje ta promena i čuva istorija svih promena za svakog člana.

Aplikacija nudi mogućnost dodavanja novih podataka u sistem, ažuriranje i brisanje postojećih, kao i pribavljanje različitog skupa podataka u zavisnosti od kriterijuma pretrage.

Unutar same aplikacije implementiran je sistem za obradu izuzetaka i svaki put kada korisnik pokuša da odradi akciju koja nije moguća iz bilo kog razloga, sistem će korisnika obavestiti standardizovanim porukama.



JAVA

Programski jezik

Programski jezik Java je korišćen za pisanje programskog koda.

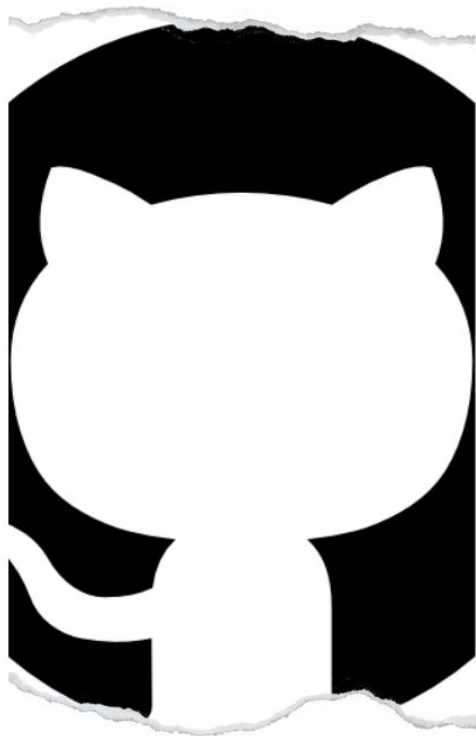


SPRING BOOT

Radni okvir

Spring Boot je korišćen kao osnova za razvoj aplikacije. Spring Boot je korišćen za konfigurisanje i upravljanje osnovnim karakteristikama aplikacije

ALATI



GITHUB

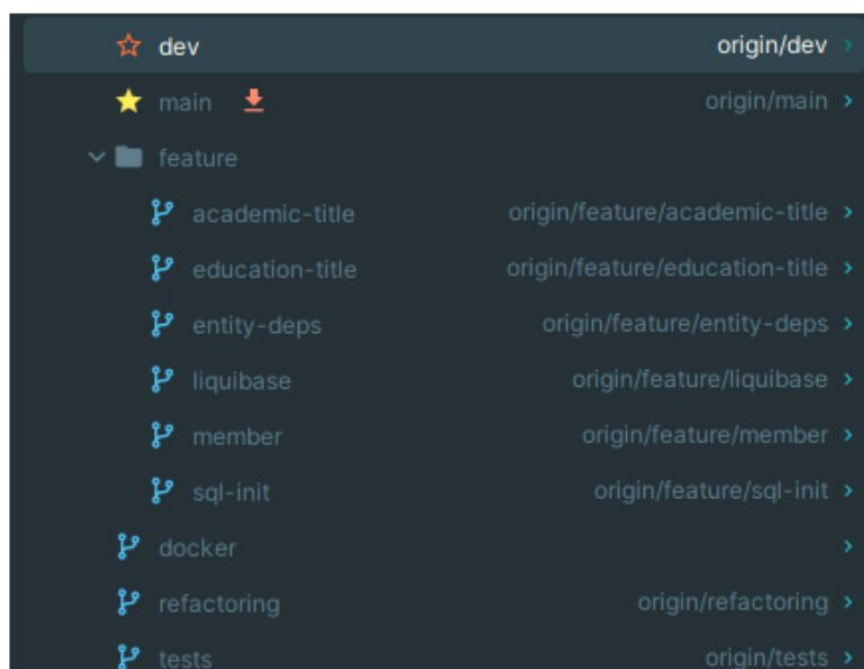
GitHub je popularna platforma za kontrolu verzija i hosting projekata koji koriste sistem Git. Omogućava efikasnu izradu projekata, praćenje istorije izmena, upravljanje zadacima i pregledanje koda. Pored toga, GitHub pruža alate za automatsko testiranje i integraciju sa raznim servisima.

GitHub je korišćen kao platforma za kontrolu verzija tokom razvoja aplikacije. Na GitHub-u je kreiran repozitorijum koji je služio kao centralno mesto za čuvanje i upravljanje izvornim kodom aplikacije. Korišćenje GitHub platforme je unapredilo rad omogućavajući efikasnu kontrolu verzija i upravljanje izmenama tokom razvoja aplikacije.

UPOTREBA GRANA (BRANCHES)

Prilikom izrade aplikacije korišćene su grane (branches) u Git-u kako bi se radilo na različitim funkcionalnostima ili ispravkama nezavisno od glavne grane (main). Svaka funkcionalnost ili ispravka je razvijana u odvojenoj grani, što je omogućilo paralelni razvoj različitih funkcionalnosti.

U procesu razvoja aplikacije korišćene su određene prakse u organizaciji grana u Git-u, sa posebnim fokusom na glavnu granu (main), razvojnu granu (dev) i grane funkcionalnosti (feature branches).

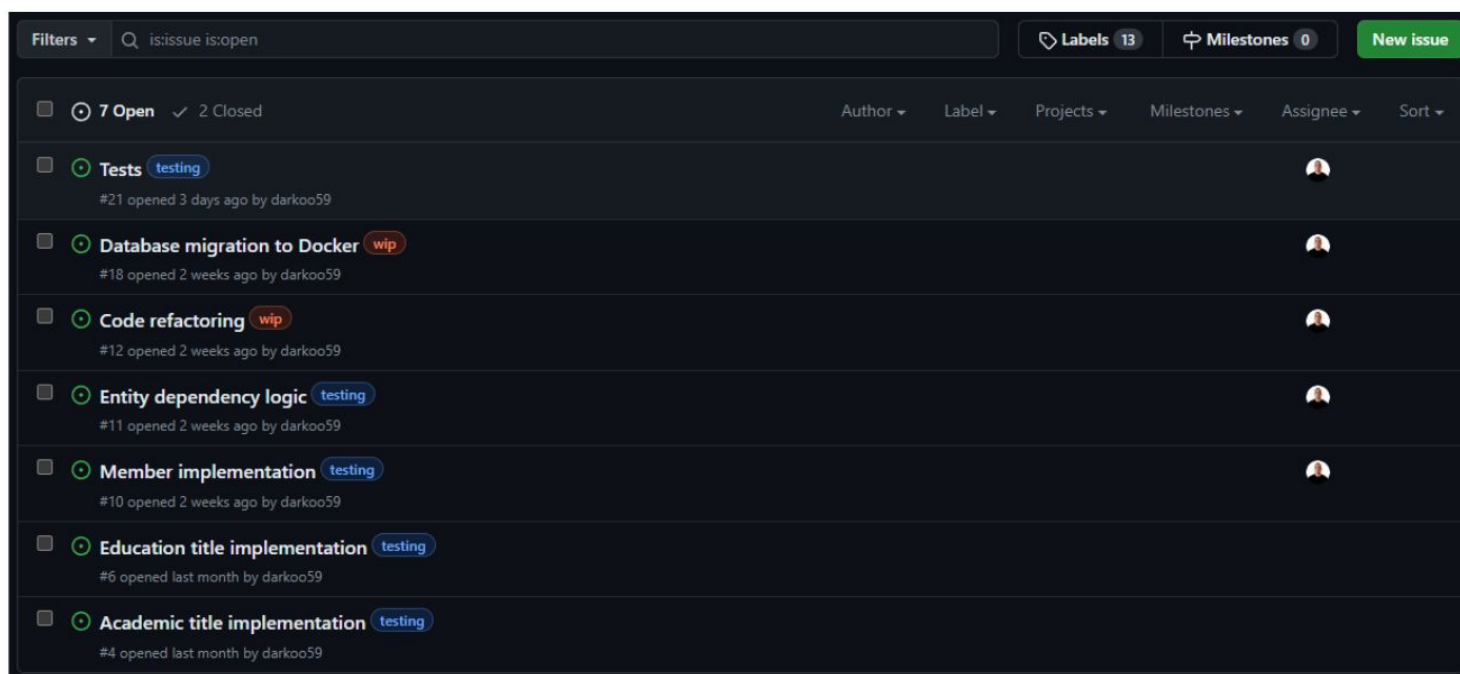


ALATI

ISSUES

Problemi, zahtevi i zadaci u vezi sa razvojem aplikacije su dokumentovani kao GitHub Issue. Issues su korišćeni za praćenje problema vezanih za kod, funkcionalnosti, poboljšanja ili greške koje treba rešiti. Issues su kategorizovani prema vrsti problema ili zadatka, kao što su problemi (bugs), funkcionalnosti, poboljšanja, ili drugi zadaci.

U procesu razvoja aplikacije, korišćeni su GitHub Issues za praćenje problema, zahteva i zadataka, uz dodatnu organizaciju kroz korišćenje različitih labela za identifikaciju i označavanje različitih vrsta kartica (issues).



The screenshot shows the GitHub Issues interface for a repository. At the top, there's a search bar with the query 'is:issue is:open'. To the right, there are buttons for 'Labels 13' and 'Milestones 0', and a 'New issue' button. Below the search bar, there's a summary bar showing '7 Open' and '2 Closed' issues. The main list of issues includes:

- Tests** (testing label) - #21 opened 3 days ago by darkoo59
- Database migration to Docker** (wip label) - #18 opened 2 weeks ago by darkoo59
- Code refactoring** (wip label) - #12 opened 2 weeks ago by darkoo59
- Entity dependency logic** (testing label) - #11 opened 2 weeks ago by darkoo59
- Member implementation** (testing label) - #10 opened 2 weeks ago by darkoo59
- Education title implementation** (testing label) - #6 opened last month by darkoo59
- Academic title implementation** (testing label) - #4 opened last month by darkoo59

documentation	Improvements or additions to documentation
done	Task that is finished
duplicate	This issue or pull request already exists
enhancement	New feature or request
invalid	This doesn't seem right
question	Further information is requested
testing	Testing functionality
todo	Feature that needs to be done

Za organizaciju i označavanje Issues, korišćene su različite labele. Svaka labela je imala svoju boju i naziv koji su jasno identifikovali vrstu ili prioritet problema. Na primer, korišćena je labela "wip" (*work in progress*), "testing" (potrebno je testiranje), "done" (za zadatke koji su završeni) i tako dalje.

ALATI


























PULL REQUESTS

U procesu razvoja aplikacije, korišćeni su pull request-ovi (PR-ovi) za integrisanje promena između različitih grana kako bi se osigurala stabilnost, kvalitet i transparentnost uvođenja novih funkcionalnosti ili poboljšanja.

Kada bi se završio razvoj određene funkcionalnosti na svojoj grani funkcionalnosti (npr. "feature/department"), koristio bi se pull request (PR) za spajanje te grane nazad u razvojnu granu ("dev"). Za posebne grane koje su bile namenjene testiranju ili refaktorisanju (npr. "testing", "refactoring"), takođe bi se koristili PR-ovi za spajanje tih grana nazad u razvojnu granu ("dev").

Kada bi razvojna grana ("dev") bila spremna za objavljivanje nove verzije aplikacije, koristio bi se PR za spajanje razvojne grane ("dev") nazad u glavnu granu ("main"). Ovaj PR bi predstavljao korak ka objavljivanju stabilne verzije aplikacije, pa bi pregled PR-a bio posebno pažljiv kako bi se osigurala kvalitetna verzija na glavnoj grani.

Svi PR-ovi su prolazili kroz proces još jednog detaljnog pregleda pre odobravanja. Pregled PR-ova uključivao bi proveru koda, testiranja i provere funkcionalnosti kako bi se osigurao integritet aplikacije i izbegle moguće greške.

<input type="checkbox"/>	 0 Open	<input checked="" type="checkbox"/> 16 Closed	Author ▾	Label ▾	Projects ▾	Milestones ▾	Reviews ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>		Merge dev into main 							
	#25 by darkoo59 was merged 3 hours ago								
<input type="checkbox"/>		Fix actions 							
	#24 by darkoo59 was closed 3 hours ago								
<input type="checkbox"/>		Add actions							
	#23 by darkoo59 was merged 3 days ago								
<input type="checkbox"/>		Merge dev to main							
	#22 by darkoo59 was merged 3 days ago								
<input type="checkbox"/>		Tests							
	#20 by darkoo59 was merged 3 days ago								
<input type="checkbox"/>		Add sonarqube yml file							
	#19 by darkoo59 was merged last week								
<input type="checkbox"/>		Liquibase logic, entity deps and code refactoring							
	#17 by darkoo59 was merged 2 weeks ago								
<input type="checkbox"/>		Add liquibase and SQL scripts							
	#16 by darkoo59 was merged 2 weeks ago								
<input type="checkbox"/>		Code refactoring							
	#14 by darkoo59 was merged 2 weeks ago								
<input type="checkbox"/>		Add entity dependencies							
	#13 by darkoo59 was merged 2 weeks ago								
<input type="checkbox"/>		Add CRUD logic for member functionality							
	#9 by darkoo59 was merged 2 weeks ago								
<input type="checkbox"/>		Merging dev into main branch							
	#8 by darkoo59 was merged last month								
<input type="checkbox"/>		Add education title and scientific field CRUD logic							
	#7 by darkoo59 was merged last month								





ALATI

GITHUB ACTIONS

U procesu razvoja aplikacije, korišćeni su GitHub Actions kao deo kontinuirane integracije i kontinuirane dostave (CI/CD) prilikom izvršavanja određenih akcija, automatski pri push-u na dev i main grane. GitHub Actions omogućava automatizaciju različitih aktivnosti i procesa unutar GitHub okruženja.

Kada bi bilo izvršeno slanje (push) izmena na dev ili main granu, GitHub Action bi automatski pokrenuo skripte i testove kako bi se osiguralo da nove izmene nisu prouzrokovale probleme ili greške. Rezultati testova bi bili prikazani na GitHub interfejsu, omogućavajući da se brzo otkriju i reše potencijalni problemi.

```
6   on:
7     push:
8       branches:
9         - main
10        - dev
11
12    jobs:
13      build:
14
15        runs-on: ubuntu-latest
16
17        services:
18          postgres:
19            image: postgres
20            env:
21              POSTGRES_PASSWORD: postgres
22              POSTGRES_USER: postgres
23            ports:
24              - 5432:5432
25            # Set health checks to wait until postgres has started
26            options: >-
27              --health-cmd pg_isready
28              --health-interval 10s
29              --health-timeout 5s
30              --health-retries 5
31
32        steps:
33          - uses: actions/checkout@v3
34          - name: Set up JDK 17
35            uses: actions/setup-java@v3
36            with:
37              java-version: '17'
38              distribution: 'oracle'
39              cache: maven
40          - name: Check PostgreSQL Status
41            run: |
42              docker ps
43              docker exec $(docker ps -q --filter ancestor=postgres) psql -U postgres -c "\l"
44          - name: Build with Maven
45            run: |
46              mvn clean install
```

4 workflow runs	Event ▾	Status ▾	Branch ▾	Actor ▾
 Merge pull request #25 from darkoo59/dev Maven Package #19: Commit 59062bc pushed by darkoo59			main	3 hours ago 1m 4s
 Add class diagram image Maven Package #18: Commit ae57276 pushed by darkoo59			dev	3 hours ago 1m 3s
 Fix workflow Maven Package #17: Commit dc5580d pushed by darkoo59			dev	10 hours ago 53s
 Fix action yml Maven Package #15: Commit 3f583b2 pushed by darkoo59			main	3 days ago 1m 1s

ALATI

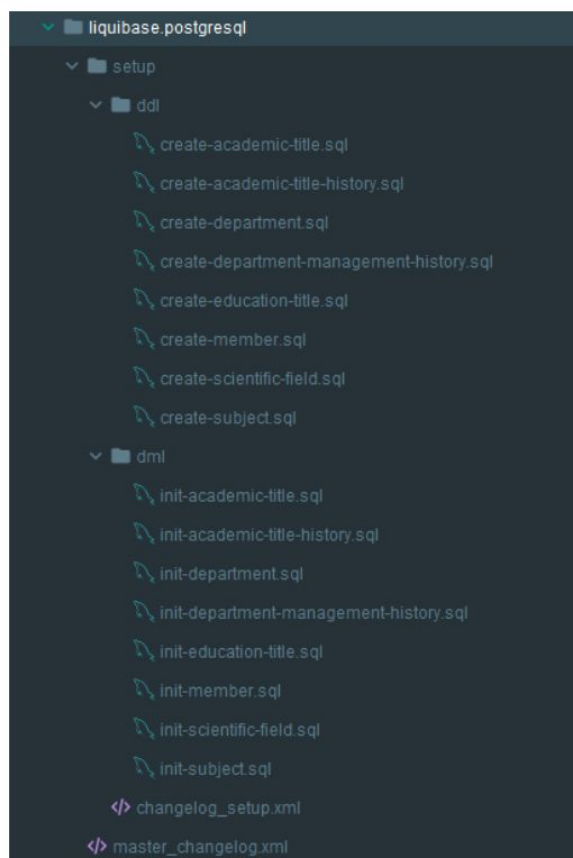
LIQUIBASE



U procesu razvoja projekta, korišćen je alat Liquibase za upravljanje migracijama šeme baze podataka, prilagođen za korišćenje sa PostgreSQL bazom podataka. Liquibase omogućio je efikasno upravljanje strukturom baze podataka putem definisanja i primene promena na bazu kroz verzionisane migracije.

Za svaku promenu u strukturi baze podataka, kao što su dodavanje novih tabela ili modifikacija postojećih tabela definisane su migracije. Liquibase se integriše u proces aplikacije tako da automatski primenjuje migracije na bazu podataka prilikom pokretanja aplikacije. Prilikom pokretanja, Liquibase proverava trenutno stanje baze podataka i primenjuje samo one migracije koje još nisu primenjene ili su se promenile od poslednjeg pokretanja.

Korišćenje Liquibase-a za upravljanje migracijama šeme baze podataka omogućilo je da se održava konzistentnost i integritet baze podataka tokom razvoja i evolucije aplikacije. Automatizacija primene migracija pomaže u održavanju baze podataka sinhronizovanom sa trenutnim stanjem aplikacije i olakšava upravljanje promenama u šemi baze podataka tokom vremena.



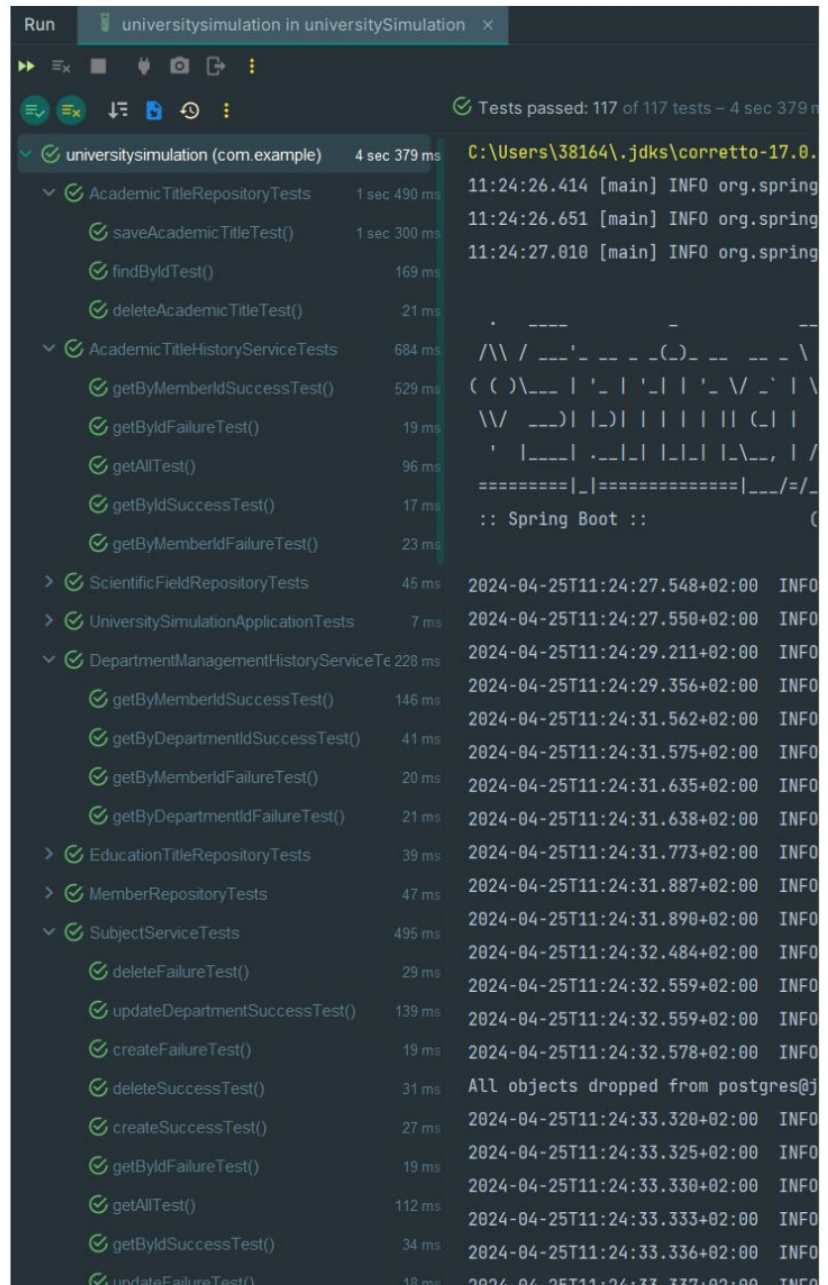
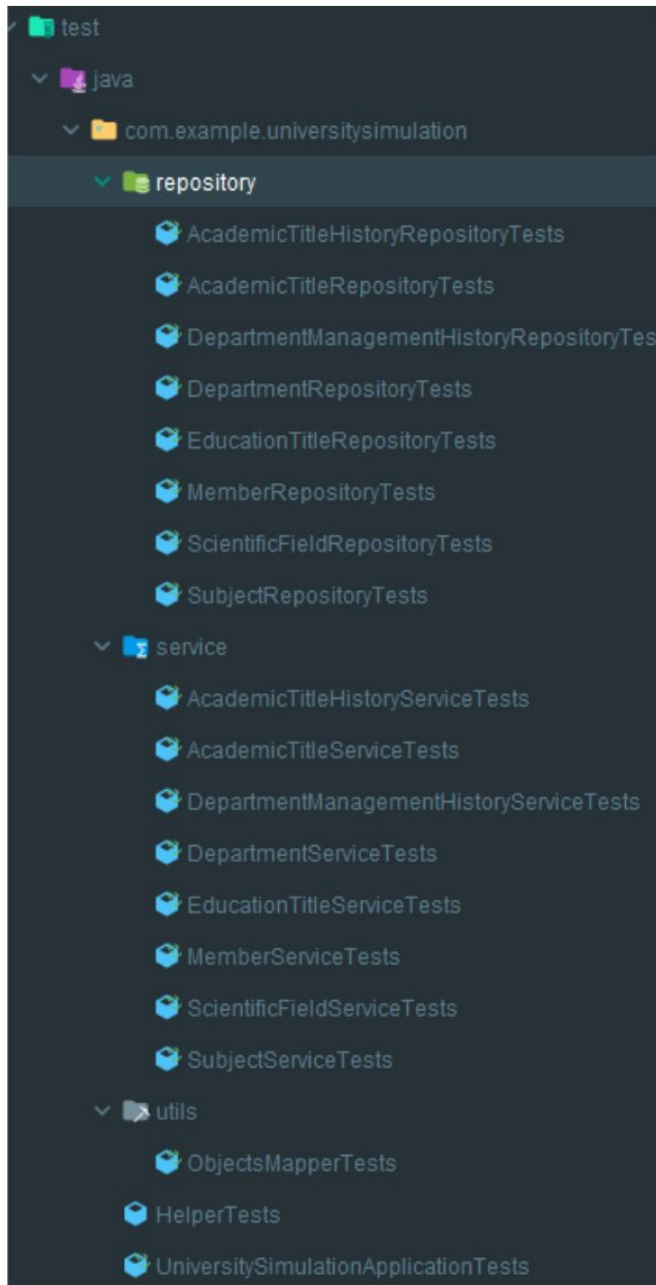
ALATI

TESTIRANJE - JUNIT 5

U procesu testiranja aplikacije, korišćen je JUnit 5 kao glavni radni okvir za pisanje i izvršavanje testova. JUnit 5 je popularni alat za testiranje Java aplikacija koji pruža bogat set funkcionalnosti i poboljšanja u odnosu na prethodne verzije.

Za glavne delove aplikacije, kao što su servisi i repository klase, ili druge komponente, razvijeni su odgovarajući JUnit 5 testovi. Testovi su pisani kao metode u posebnim klasama koje se nazivaju test klasama. U testovima se koriste JUnit 5 anotacije za definisanje testova, podešavanje početnih uslova, i proveru očekivanih rezultata.

JUnit 5 dolazi sa raznovrsnim setom *assert* metoda za proveru očekivanih rezultata i stanja. Korišćene su metode poput *assertEquals*, *assertTrue*, *assertNotNull* i druge za proveru da li se stvarni rezultati testa podudaraju sa očekivanim rezultatima.



SONARQUBE I SONARLINT

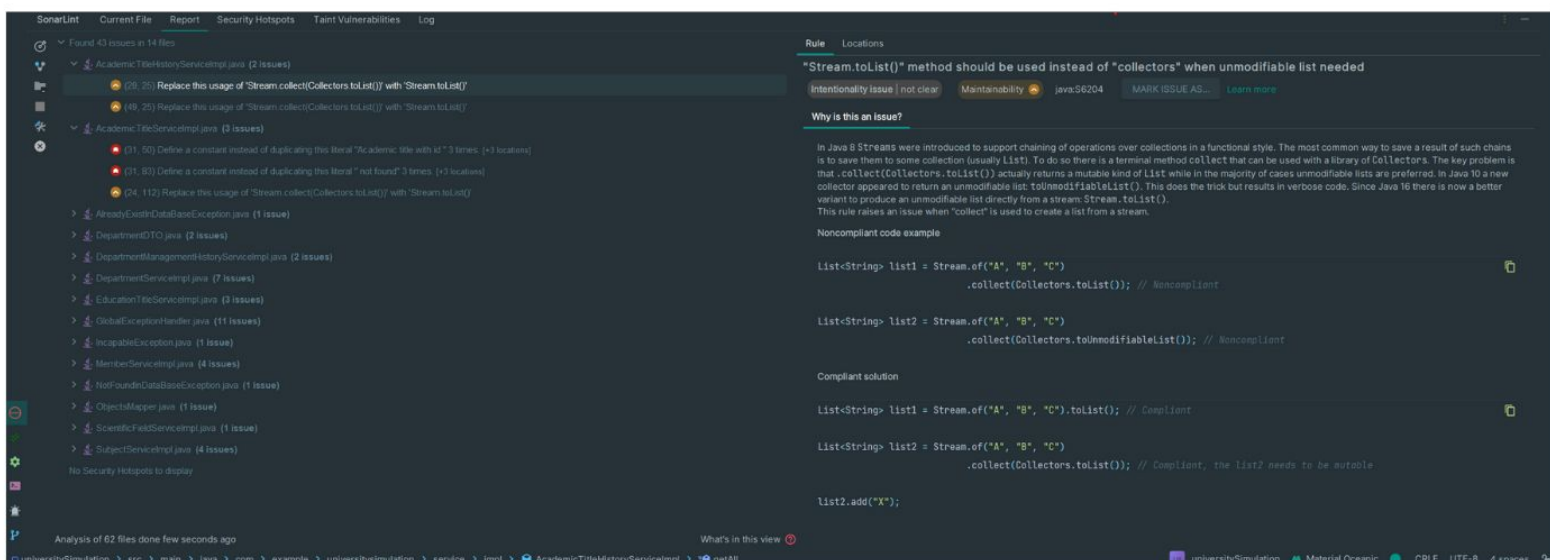


U procesu razvoja aplikacije, korišćeni su SonarQube i plugin SonarLint kako bi se obezbedila analiza kvaliteta koda i otkrivanje potencijalnih problema ili grešaka u kodu. Ovi alati su integrisani u razvojno okruženje IntelliJ IDEA radi kontinuirane analize i poboljšanja kvaliteta koda.

SonarQube je platforma otvorenog koda koja pruža statičku analizu koda radi identifikacije različitih problema u kodu, kao što su bug-ovi, loše prakse, bezbednosni propusti, nepokriveni kod ili loša arhitektura. SonarLint je plugin za IntelliJ IDEA koji omogućava lokalnu analizu koda direktno unutar IDE-a.

Kombinacija SonarQube platforme za analizu kvaliteta koda i SonarLint plugina za IntelliJ IDEA omogućila je da se održava visok nivo standarda u kodu, identifikuju potencijalni problemi i greške u realnom vremenu tokom razvoja, i unapređuje ukupan kvalitet aplikacije.

U projektu, SonarQube je korišćen za redovnu analizu izvornog koda kako bi se identifikovali potencijalni problemi i poboljšao kvalitet koda. Problemi su kategorizovani prema ozbiljnosti (Critical, Major, Minor) i prioritetu, omogućavajući da se fokusira na najvažnije aspekte kvaliteta koda. SonarLint je korišćen kako bi se dobijale real-time povratne informacije o kvalitetu koda dok se radi na implementaciji.



ALATI

JACOCO



U procesu razvoja aplikacije, korišćen je JaCoCo (Java Code Coverage) alat za merenje pokrivenosti koda testovima. JaCoCo je popularan alat za generisanje izveštaja o pokrivenosti koda koji omogućava da se utvrdi koliki deo izvornog koda je pokriven testovima

JaCoCo je integrisan u build proces projekta kako bi generisao detaljne izveštaje o pokrivenosti koda testovima. Nakon izvršavanja testova, JaCoCo analizira izvorni kod i prati koji delovi koda su izvršeni (pokriveni) tokom testiranja. JaCoCo generiše metrike pokrivenosti kao što su linije koda, grane, instrukcije i metode koje su pokrivenne testovima.

Metrike se prikazuju u izveštaju u procentima, omogućavajući uvid u ukupnu pokrivenost koda testovima i identifikuje delove koda koji nisu adekvatno pokriveni testovima.

U projektu JaCoCo se konfiguriše kao plugin u pom.xml datoteci kako bi se automatski generisali izveštaji o pokrivenosti tokom build procesa.

universitySimulation > com.example.universitysimulation.service.impl > EducationTitleServiceImpl

EducationTitleServiceImpl

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods
update(EducationTitleRequest, Long)	<div><div></div></div>	100%	<div><div></div></div>	100%	0 2	0 6	0 1
findById(Long)	<div><div></div></div>	100%	<div><div></div></div>	100%	0 2	0 4	0 1
delete(Long)	<div><div></div></div>	100%	<div><div></div></div>	100%	0 2	0 4	0 1
create(EducationTitleRequest)	<div><div></div></div>	100%	<div><div></div></div>	n/a	0 1	0 3	0 1
getAll()	<div><div></div></div>	100%	<div><div></div></div>	n/a	0 1	0 5	0 1
getById(Long)	<div><div></div></div>	100%	<div><div></div></div>	n/a	0 1	0 2	0 1
EducationTitleServiceImpl(EducationTitleRepository)	<div><div></div></div>	100%	<div><div></div></div>	n/a	0 1	0 1	0 1
Total	0 of 100	100%	0 of 6	100%	0 10	0 25	0 7

File | D:\Projekti\university-simulation\universitySimulation\target\site\jacoco\com.example.universitysimulation.service.impl\index...

universitySimulation > com.example.universitysimulation.service.impl

com.example.universitysimulation.service.impl

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
DepartmentServiceImpl	<div><div></div></div>	43%	<div><div></div></div>	33%	15 27	37 63	9 18	0 1
MemberServiceImpl	<div><div></div></div>	90%	<div><div></div></div>	75%	6 27	7 92	0 15	0 1
SubjectServiceImpl	<div><div></div></div>	96%	<div><div></div></div>	90%	1 13	1 36	0 8	0 1
AcademicTitleHistoryServiceImpl	<div><div></div></div>	90%	<div><div></div></div>	75%	1 6	1 18	0 4	0 1
EducationTitleServiceImpl	<div><div></div></div>	100%	<div><div></div></div>	100%	0 10	0 25	0 7	0 1
AcademicTitleServiceImpl	<div><div></div></div>	100%	<div><div></div></div>	100%	0 9	0 19	0 6	0 1
DepartmentManagementHistoryServiceImpl	<div><div></div></div>	90%	<div><div></div></div>	100%	1 6	2 17	1 4	0 1
ScientificFieldServiceImpl	<div><div></div></div>	100%	<div><div></div></div>	100%	0 9	0 22	0 7	0 1
Total	240 of 1,353	82%	20 of 76	73%	24 107	48 292	10 69	0 8