



UNIVERZITET U SARAJEVU
ELEKTROTEHNIČKI FAKULTET
ODSJEK ZA RAČUNARSTVO I INFORMATIKU
PRAKTIKUM – NAPREDNE WEB TEHNOLOGIJE

Menadžment sistem za kazнено-popravni zavod

DOKUMENTACIJA ZA PROJEKTNII ZADATAK
- DRUGI CIKLUS STUDIJA -

Profesor:

Doc. dr. Anel Tanović

Asistent:

Mr. dipl. ing. Irfan Prazina

Studenti:

Halilović Kemal

Mehmedović Faris

Memić Miralem

Omić Kenan

Sarajevo,
juni 2022.

Sadržaj

Popis slika	iv
1 Zadatak 1	1
1.1 Opis teme:	1
1.2 Funkcionalnosti:	1
1.3 Skice korisničkih interfejsa za funkcionalnosti:	3
1.4 Entity Relationship (ER) diagram:	7
1.5 Primjeri aplikacija koje već postoje sa istom/sličnom temom:	7
2 Zadatak 2	8
2.1 Upravljanje nad korisničkim računima	8
2.2 Upravljanje nad zatvorenicima	8
2.3 Servis za obavijesti	9
2.4 Servis za poruke	9
2.5 Administracija nad kazнено-popravnom zavodu	9
3 Zadatak 3	10
3.1 Testiranje putem Postman-a za user-service	11
3.2 Testiranje putem Postman-a za prisoner-service	14
3.3 Testiranje putem Postman-a za message-service	17
3.4 Testiranje putem Postman-a za notification-service	19
4 Zadatak 4	22
4.1 Netrivijalni webservisi	22
4.2 Service discovery (Eureka) i centralizirana konfiguracija	24
5 Zadatak 5	26
5.1 Dijagram jail → prisoner	26
5.2 Dijagram user → jail → notification	26
5.3 Dijagram user → message	27
5.4 Dijagram notification → user	27
6 Zadatak 6	28
7 Zadatak 7	29
7.1 Analiza rješenja za sigurnost sistema	29
7.2 Prikaz implementirane sigurnosti	30

8	Zadatak 8	32
8.1	Implementacija asinhronne komunikacije	32
8.2	Implementacija saga pattern-a	33
9	Zadatak 9, 10 i 11	34

Popis slika

1.1	Prikaz login-a na aplikaciju	3
1.2	Prikaz početne stranice sa obavijestima	3
1.3	Prikaz stranice za slanje zathjeva	4
1.4	Prikaz stranice za registraciju korisnika (popunjavanje podataka)	4
1.5	Prikaz stranice za brisanje/ažuriranje profila od strane upravnika	5
1.6	Prikaz stranice za dodavanje profila zatvorenika od strane upravnika	5
1.7	Prikaz stranice za pregled profila zatvorenika od strane upravnika	6
1.8	Prikaz stranice za pregled svih profila zatvorenika od strane upravnika	6
1.9	Prikaz Entity Relationship Diagram (ERD) za aplikaciju	7
2.1	ERD mikroservisa: upravljanje nad korisničkim računima	8
2.2	ERD mikroservisa: upravljanje nad zatvorenicima	8
2.3	ERD mikroservisa: servis za obavijesti	9
2.4	ERD mikroservisa: servis za poruke	9
3.1	Prikaz MySQL baze podataka podignute od strane Google Cloud SQL instance	10
3.2	GET all users	11
3.3	GET user by id	11
3.4	POST user	12
3.5	PUT user	12
3.6	DELETE user	13
3.7	GET user by id (slučaj neuspješnog zahtjeva-odgovora)	13
3.8	GET all prisoners	14
3.9	GET prisoner by id	14
3.10	POST prisoner	15
3.11	PUT prisoner	15
3.12	DELETE prisoner	16
3.13	GET prisoner by id (slučaj neuspješnog zahtjeva-odgovora)	16
3.14	GET all messages	17
3.15	GET message by id	17
3.16	POST message	18
3.17	DELETE message	18
3.18	GET message by id (slučaj neuspješnog zahtjeva-odgovora)	18
3.19	GET all notifications	19
3.20	GET notification by id	19
3.21	POST notification	20
3.22	DELETE notification	20
3.23	GET notification by id (slučaj neuspješnog zahtjeva-odgovora)	21

4.1	GET prisoners by cell id	22
4.2	GET prisoners/cells	23
4.3	GET prisoners/most-common-offense	23
4.4	GET prisoners/coming-out	24
4.5	Pokretanje centralizirane konfiguracije	24
4.6	Pokretanje Eureka servera	24
4.7	Prikaz Eureka servera	25
5.1	Dijagram jail → prisoner	26
5.2	Dijagram user → jail → notification	26
5.3	Dijagram user → message	27
5.4	Dijagram notification → user	27
6.1	Prikaz log-a za event-e	28
6.2	Prikaz event-ova u MySQL Workbench	28
7.1	POST auth/login (autorizacija, dobivanje tokena)	30
7.2	GET auth/hello (authorized)	30
7.3	GET auth/hello (unauthorized)	31
7.4	GET user by username	31
8.1	Uvezivanje komunikacije sa RabbitMQ	32
8.2	Prikaz uspješne komunikacije	32
8.3	Prikaz tabele message u bazi podataka (inverzna akcija)	33
8.4	Prikaz tabele notification transaction u bazi podataka (inverzna akcija)	33

1. Zadatak 1

1.1 Opis teme:

Upravljanje kazneno-popravnog zavoda je kompleksan sistem, čiju funkcionalnost kvalitetno može pratiti adekvatan softver. Cilj ovog softvera je da olakša protok informacija između upravnika i čuvara, kao i drugih lica vezanih za zatvorenike, što podrazumijeva one koji žele da upute zahtjev za transfer zatvorenika do onih koji žele da posjete zatvorenike u kazneno-popravnom zavodu. Također zadatak ovog softvera je da olakša generalno upravljanje ali i nadzor zatvorenika u smislu vođenja evidencije o prisutnim zatvorenicima, njihovoj lokaciji kao i događanjima u kazneno-popravnog zavodu koji se tiču njih, kao i čuvara i drugih zainteresovanih lica.

1.2 Funkcionalnosti:

Definisane su sljedeće funkcionalnosti, u svrhu kreiranja aplikacije "Menadžment sistem za kazneno-popravni zavod":

- **Kreacija korisničkog računa**
 - Čuvari koji započinju radni odnos kreiraju sebi račun, a upravnik ih odobrava
- **Ažuriranje korisničkog računa**
 - Upravnik je u stanju da svim korisnicima u određenoj mjeri ažurira korisnički račun, dok svi korisnici mogu ažurirati svoj korisnički račun
- **Brisanje korisničkog računa**
 - Upravnik je u stanju da svim korisnicima obriše korisnički račun, dok svi korisnici mogu sami svoj korisnički račun da obrišu
- **Mogućnost da se generiše kôd kako bi se novi korisnički račun otvorio**
 - Nakon potvrde zahtjeva za kreaciju korisničkog računa od strane upravnika, kôd se generiše i šalje neregistrovanom korisniku, omogućavajući mu kreaciju korisničkog računa
- **Kreacija zahtjeva**
 - Korisnici (upravnik i čuvar) imaju mogućnost kreacije zahtjeva
- **Slanje zahtjeva**
 - Upravnik može slati zahtjeve čuvarima. Čuvari također mogu slati zahtjeve između sebe, kao i upravniku

- **Procesiranje zahtjeva**

- Korisnik koji je primio zahtjev ima mogućnost da ga odobri ili odbije uz mogućnost da napiše komentar vezan za svoju odluku

- **Komunikacija između korisnika u zavisnosti od nivoa pristupa korisničkog računa**

- Korisnici su u stanju da jedan drugom šalju poruke u zavisnosti od toga koji je nivo pristupa njihovog korisničkog računa; tako da upravnik može komunicirati sa sudom i čuvarima. Sud može komunicirati između sebe kao i sa upravnikom, dok čuvari također mogu komunicirati između sebe kao i sa upravnikom

- **Mogućnost postavljanja obavijesti koje su vidljive samo određenim korisnicima**

- Upravnik je u mogućnosti da kreira obavijesti koje će se prikazati određenim korisnicima

- **Mogućnost pretrage zatvorenika, kao i drugih korisnika**

- Korisnici su u mogućnosti da pretražuju druge korisnike, kao i zatvorenike, i to po različitim kriterijima kao što su username za korisnike, i lokacija u kazneno-popravnog zavodu za zatvorenike

- **Mogućnost unosa profila zatvorenika**

- Upravnik je u stanju da unese nove zatvorenike u sistem

- **Mogućnost ažuriranja profila zatvorenika**

- Upravnik je u stanju da u potpunosti ažurira podatke vezane za zatvorenika, dok su čuvari u mogućnosti da ostavljaju komentare na njihove profile

- **Mogućnost brisanja profila zatvorenika**

- Upravnik je u stanju izbriše profile zatvorenika u sistemu

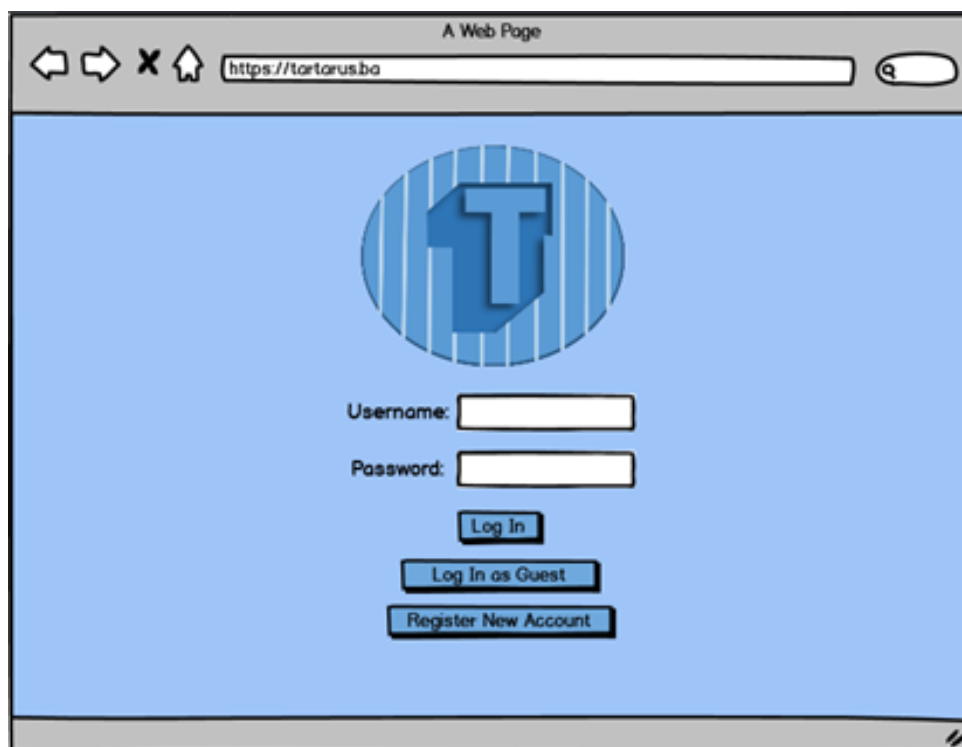
- **Dodavanje specifikacija kazneno-popravnog zavoda**

- Upravnik ima sposobnost da dodaje specifikacije kazneno-popravnog zavoda

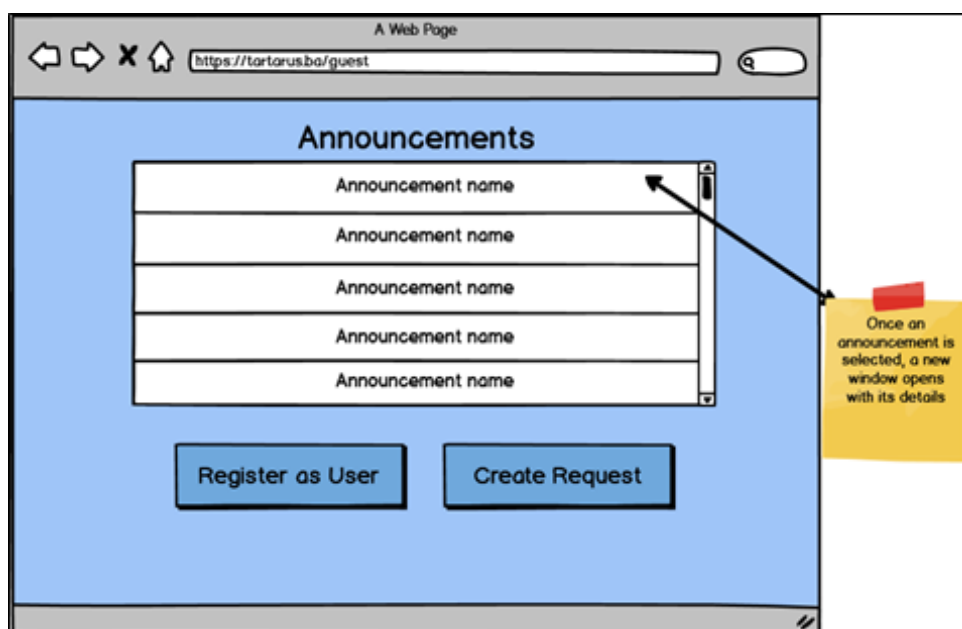
- **Ažuriranje specifikacija kazneno-popravnog zavoda**

- Upravnik ima sposobnost da ažurira specifikacije kazneno-popravnog zavoda kao što je kapacitet broja zatvorenika

1.3 Skice korisničkih interfejsa za funkcionalnosti:



Slika 1.1: Prikaz login-a na aplikaciju



Slika 1.2: Prikaz početne stranice sa obavijestima

A Web Page
https://tortarus.ba/guest/request

Name: Type of Request:

Surname: Occupation:

Email:

Request description:

Send

Account creation
Visitation
Other

Judge
Lawyer
Court Representative
Guard
Civilian (Guest)

Slika 1.3: Prikaz stranice za slanje zathjeva

A Web Page
https://tortarus.ba/registration

Welcome
Occupation - Name - Surname

Upload Picture

Username:

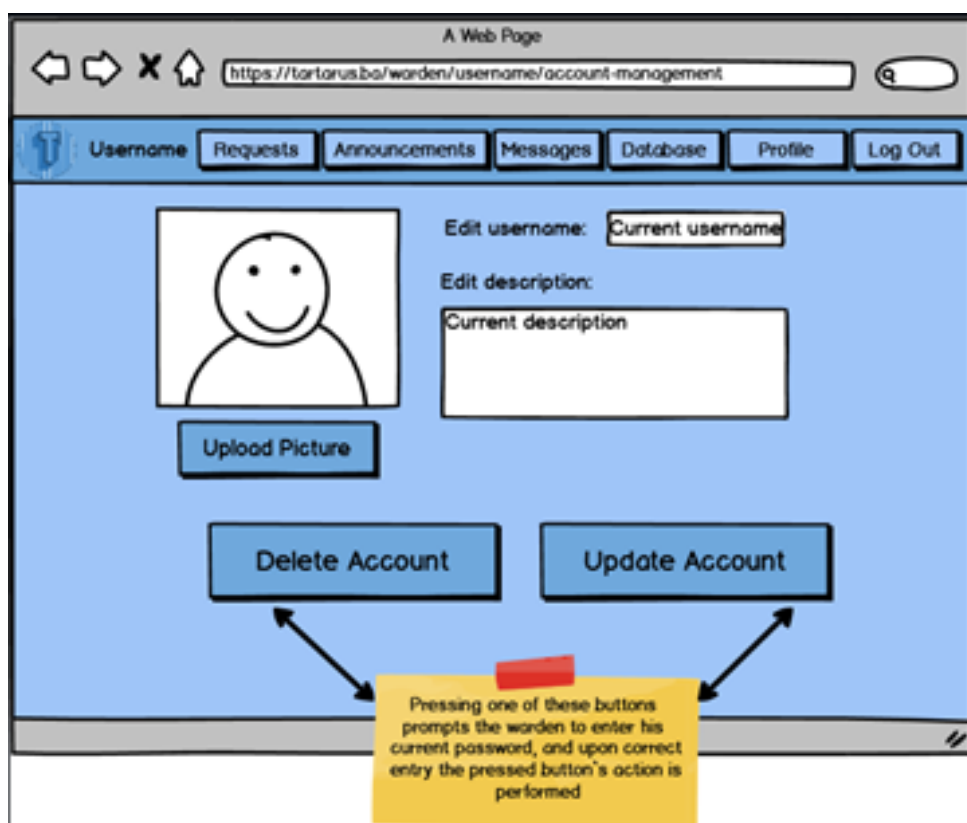
Password:

Repeat password:

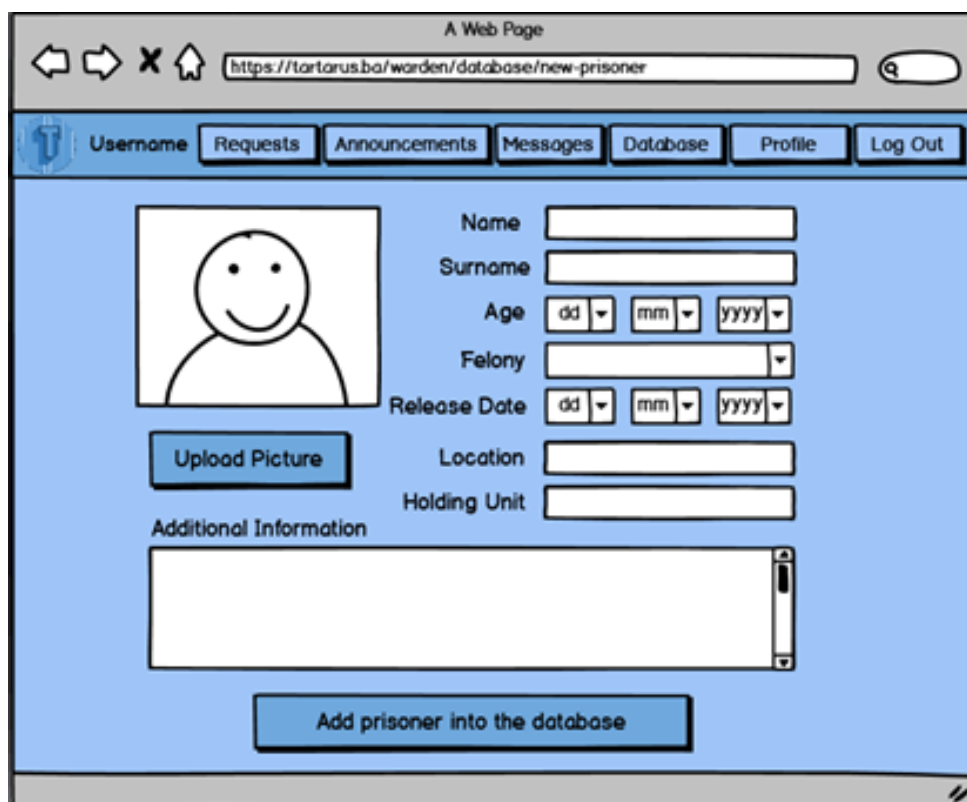
Description:

Complete Account Creation

Slika 1.4: Prikaz stranice za registraciju korisnika (popunjavanje podataka)



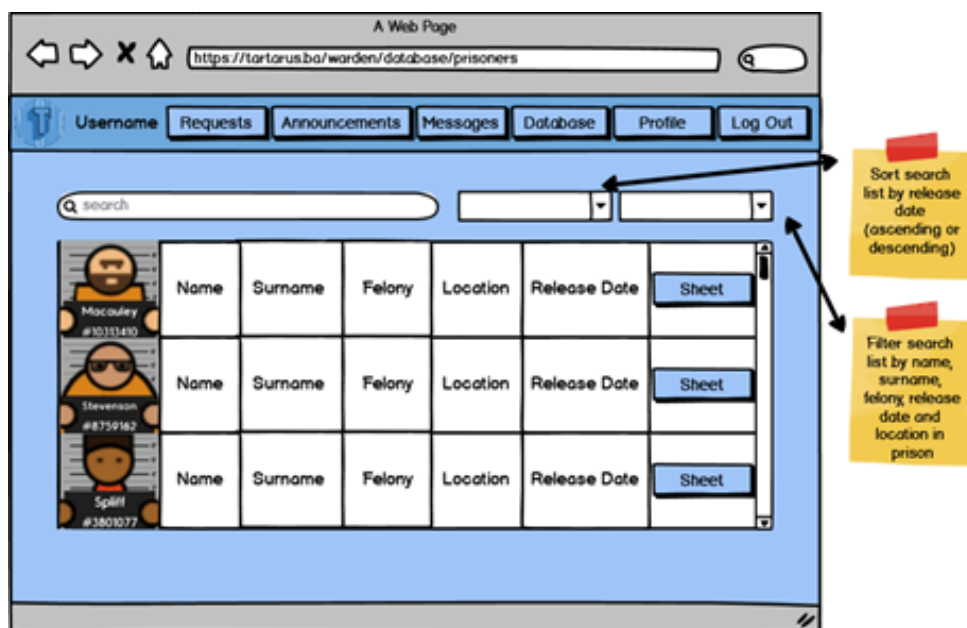
Slika 1.5: Prikaz stranice za brisanje/ažuriranje profila od strane upravnika



Slika 1.6: Prikaz stranice za dodavanje profila zatvorenika od strane upravnika

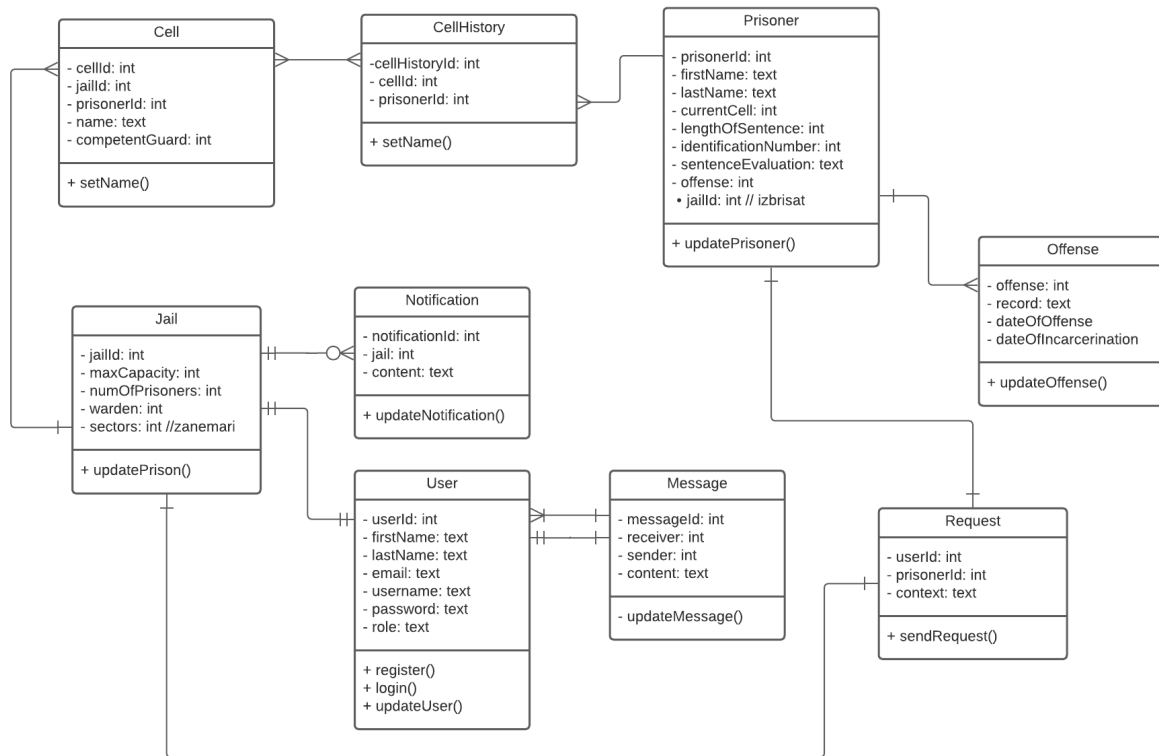


Slika 1.7: Prikaz stranice za pregled profila zatvorenika od strane upravnika



Slika 1.8: Prikaz stranice za pregled svih profila zatvorenika od strane upravnika

1.4 Entity Relationship (ER) diagram:



Slika 1.9: Prikaz Entity Relationship Diagram (ERD) za aplikaciju

1.5 Primjeri aplikacija koje već postoje sa istom/sličnom temom:

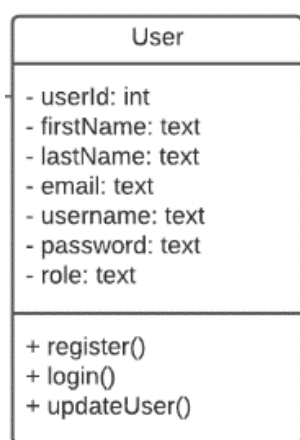
- **eFORCE** - ovaj softver omogućava firmama da koriste kompjuterski potpomognutu dispečersku funkciju za unos i pristup informacijama, statistikama i drugim bitnim podacima u realnom vremenu. Platforma namjerava smanjiti vrijeme odgovora koristeći kontrolu komandne linije i može upravljati otpremom za više agencija. Uključuje opsežne planove odgovora, ESRI mapiranje, dijeljenje podataka, upozorenja na tajmeru statusa i dinamičku traku upozorenja. eFORCE softver pruža rješenje koje omogućava administratorima da planiraju optimalne rute.
- **JailTracker** - je rješenje za upravljanje zatvorom zasnovano na oblaku koje je posebno dizajnirano da zadovolji jedinstvene potrebe upravljanja velikim i malim popravnim ustanovama. Platforma pruža mogućnost interne i eksterne komunikacije i nudi izvještaje, portale i integraciju internih i eksternih sistema. Alati koje pruža JailTracker imaju za cilj automatizaciju ručnih zadataka koji se ponavljaju i eliminaciju suvišnih informacija.
- **Guard1** - je softver za fizičku sigurnost dizajniran da pomogne preduzećima u popravnim, zdravstvenim i ugostiteljskim djelatnostima da automatiziraju sigurnosne operacije objekata i nadgledaju obilaske stražara. Omogućava administratorima da prate napredak patroliranja radnika na terenu i statuse u realnom vremenu kao što su završene, u toku ili propuštene kontrolne tačke na objedinjene platforme.

2. Zadatak 2

Funkcionalnosti smo podijelili na 5 mikroservisa.

2.1 Upravljanje nad korisničkim računima

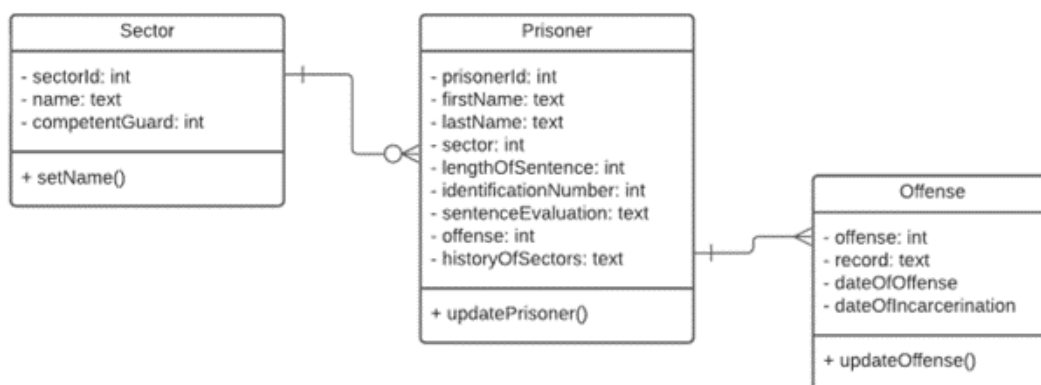
- CRUD operacije nad korisničkog računa
- Registracija i prijava korisničkog računa



Slika 2.1: ERD mikroservisa: upravljanje nad korisničkim računima

2.2 Upravljanje nad zatvorenicima

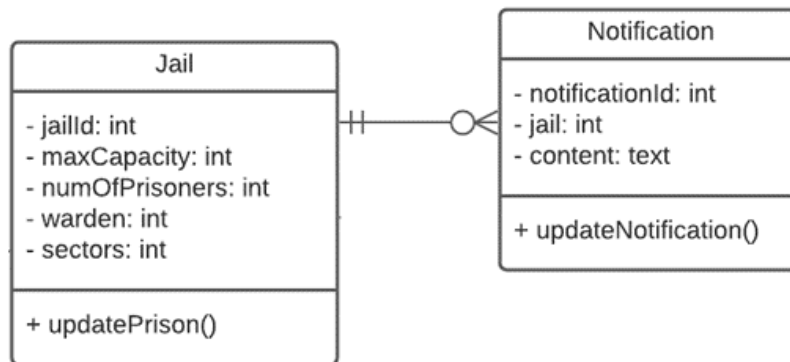
- CRUD operacije nad profilom zatvorenika
- Pretraga i filtriranje zatvorenika



Slika 2.2: ERD mikroservisa: upravljanje nad zatvorenicima

2.3 Servis za obavijesti

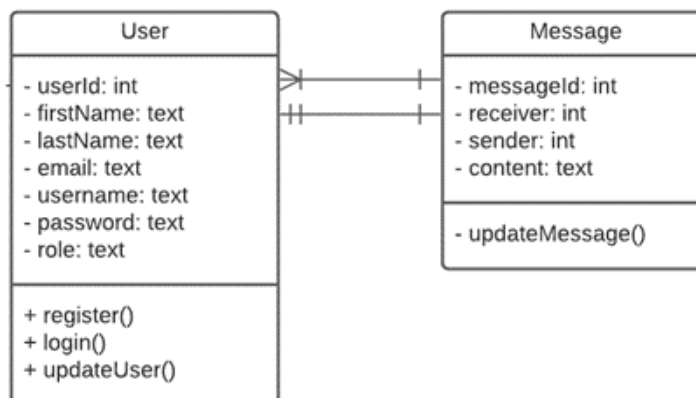
- Mogućnost postavljanja obavijesti koje su vidljive samo određenim korisnicima
- Mogućnost pregleda i brisanja postavljениh obavijesti



Slika 2.3: ERD mikroservisa: servis za obavijesti

2.4 Servis za poruke

- CRUD operacije nad zahtjevima
- Slanje i procesiranje zahtjeva (u zavisnosti od nivoa pristupa korisničkog računa)



Slika 2.4: ERD mikroservisa: servis za poruke

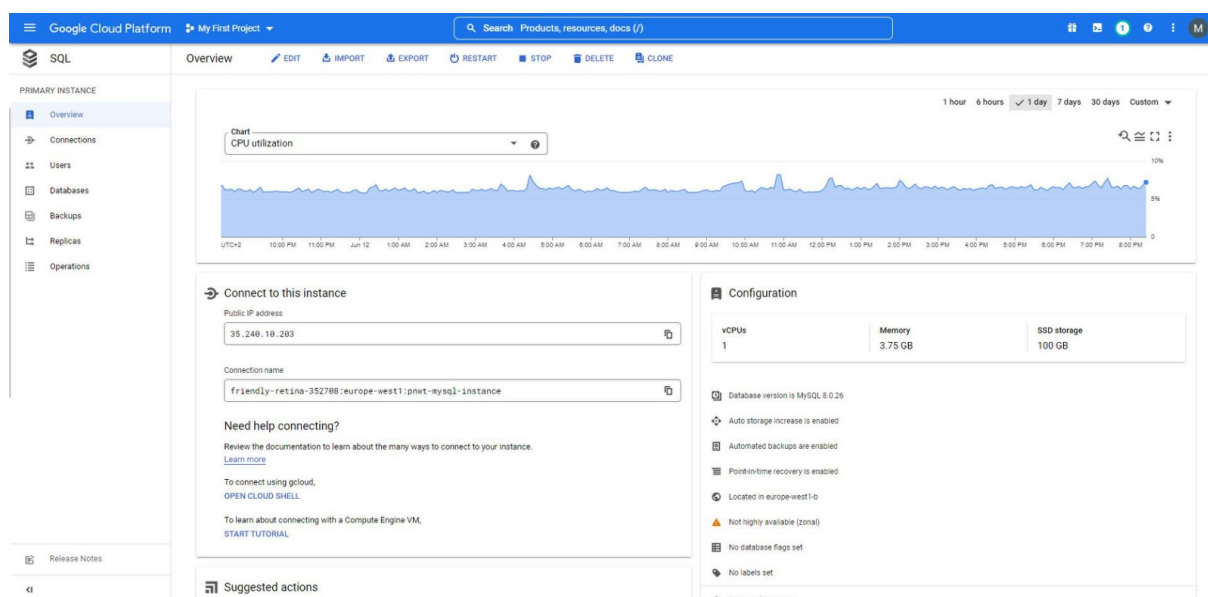
2.5 Administracija nad kazneno-popravnim zavodu

- Dodavanje sigurnosnih specifikacija kazneno-popravnog zavoda
- Ažuriranje sigurnosnih specifikacija kazneno-popravnog zavoda

3. Zadatak 3

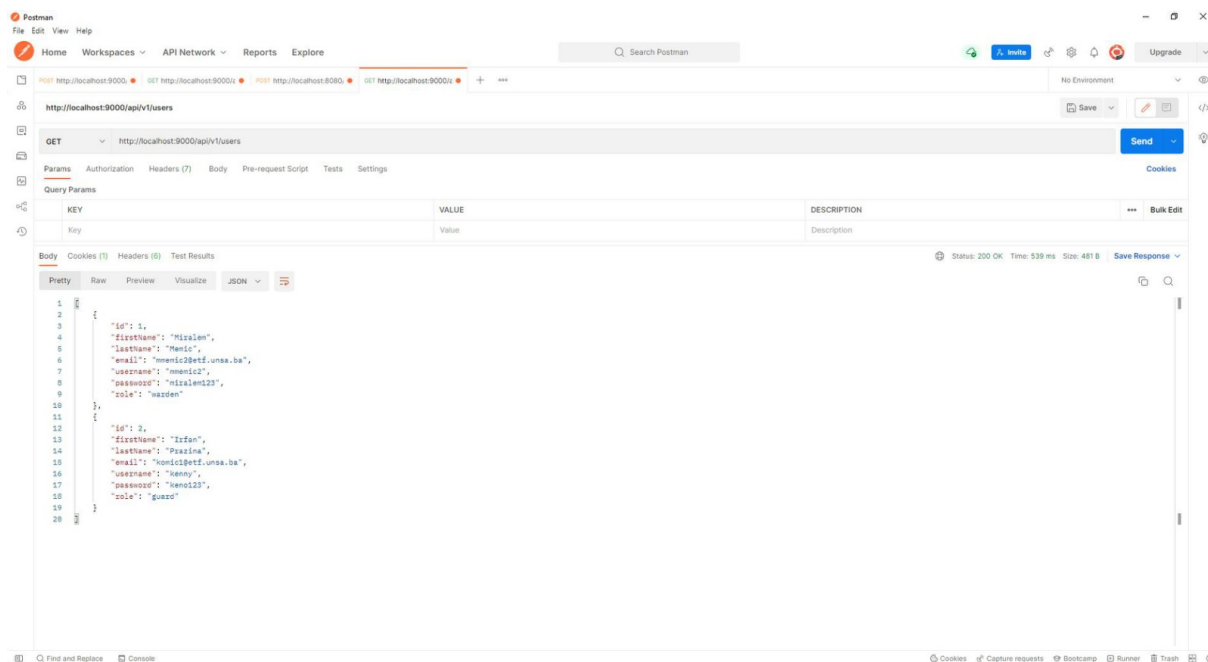
U naredom zadatku dokumentovan je svaki servis koji je implementiran na način da je priložen screenshot testa kroz Postman. Test sadrži prikaz osnovnih CRUD operacija, a pored toga za user-service je prikazan jedan slučaj uspješnog zahtjeva-odgovora i jedan slučaj neuspješnog zahtjeva-odgovora (podaci u pogrešnom formatu, nepotpuni podaci i sl.).

Prije prikaza ruta za dohvaćanje podataka, neophodno je naglasiti da je za bazu podataka korištena MySQL baza podataka podignuta od strane Google Cloud SQL instance. U svrhu postizanja boljeg pristupa svih članova tima, izvršene se manje izmjene u lokalnim config file-ovima. Općenito, MySQL funkcionalnost koju pruža Cloud SQL instance je ista kao i funkcionalnost koju pruža lokalno hostirana MySQL instance. Pored toga, Google Cloud Platform pruža baze podataka koje brzo rade, ne ponestaju bez prostora i daje aplikaciji redundantnu, pouzdanu pohranu koja joj je potrebna.

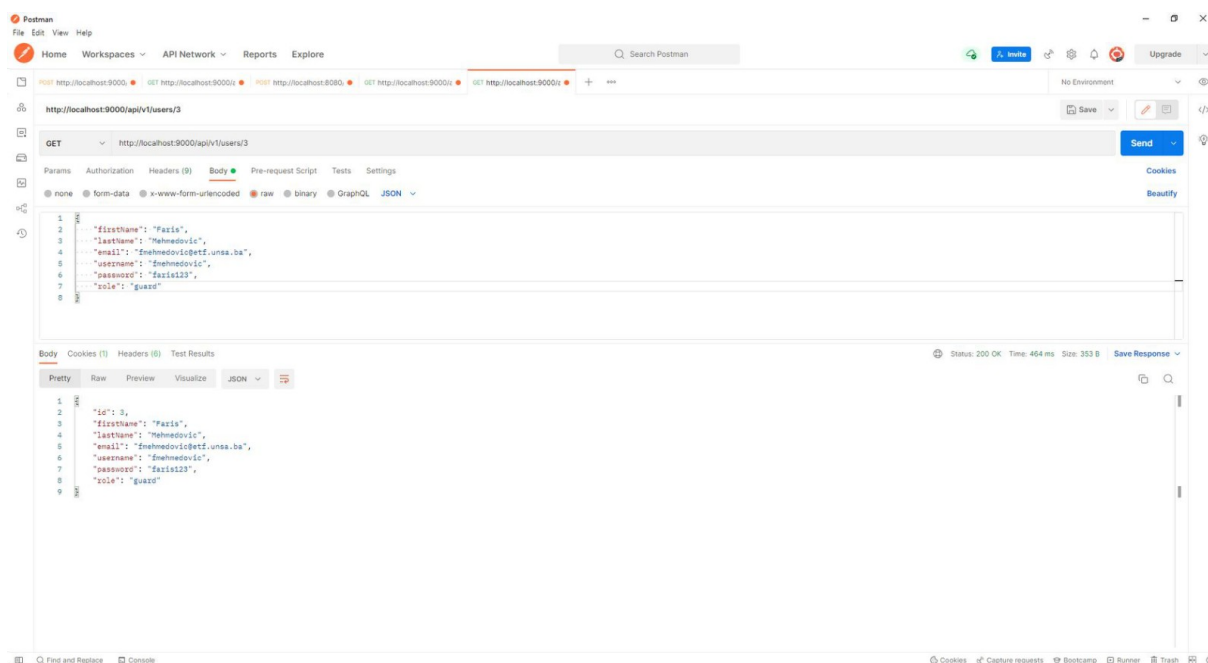


Slika 3.1: Prikaz MySQL baze podataka podignute od strane Google Cloud SQL instance

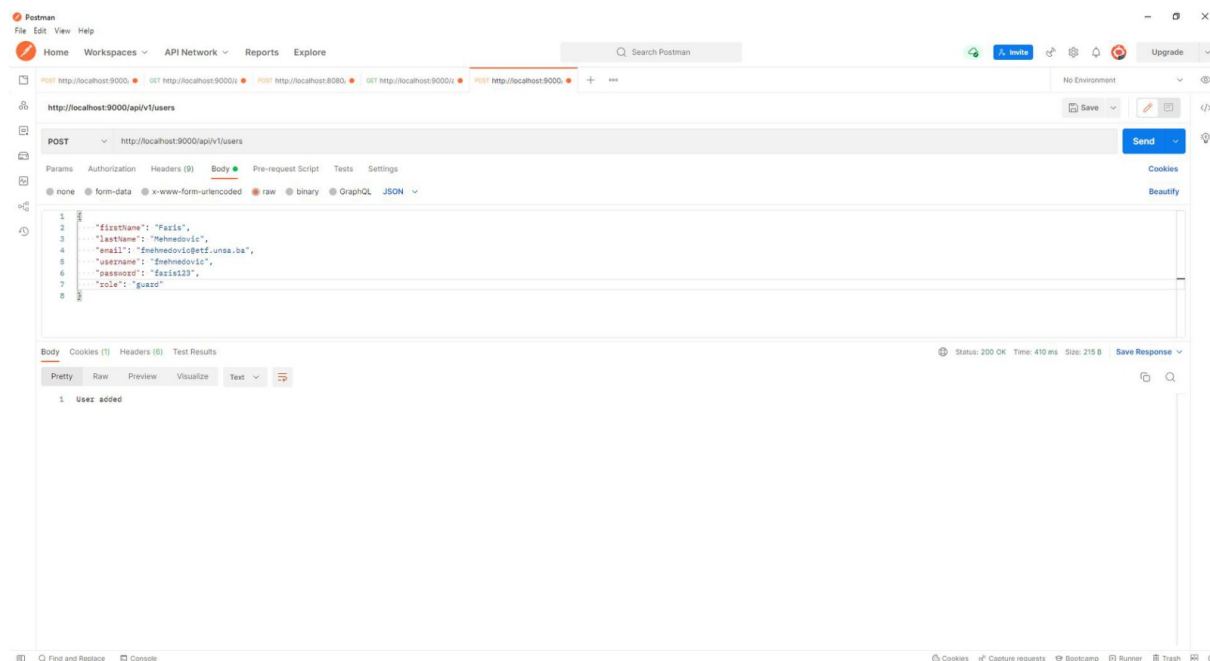
3.1 Testiranje putem Postman-a za user-service



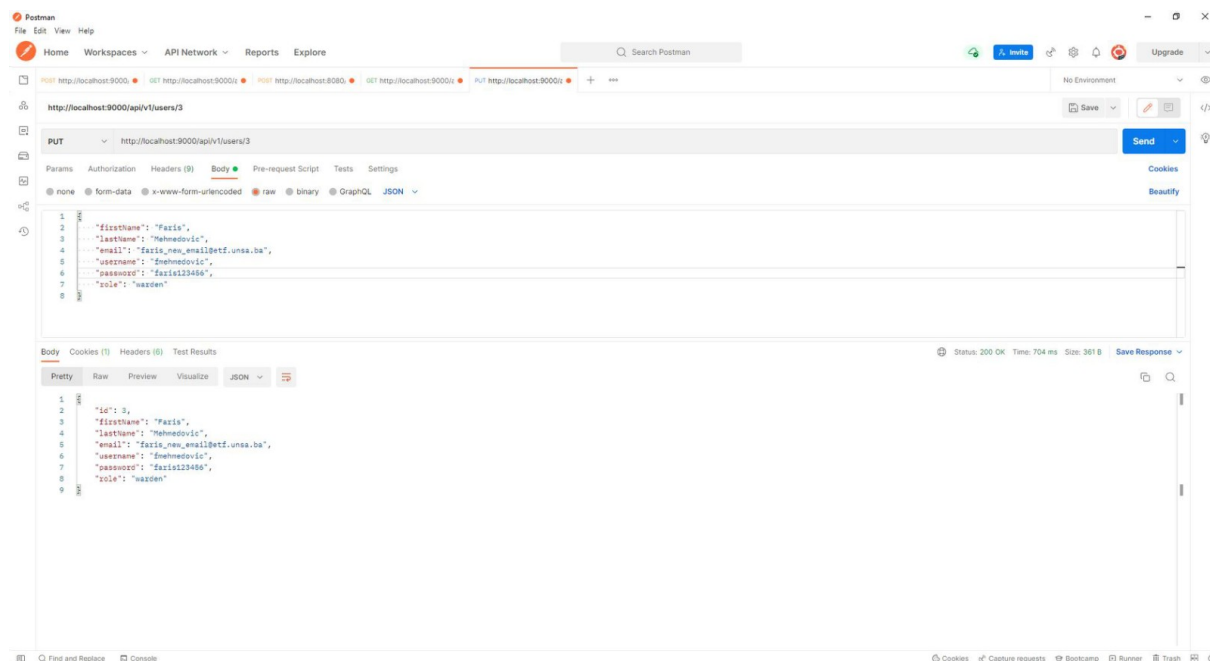
Slika 3.2: GET all users



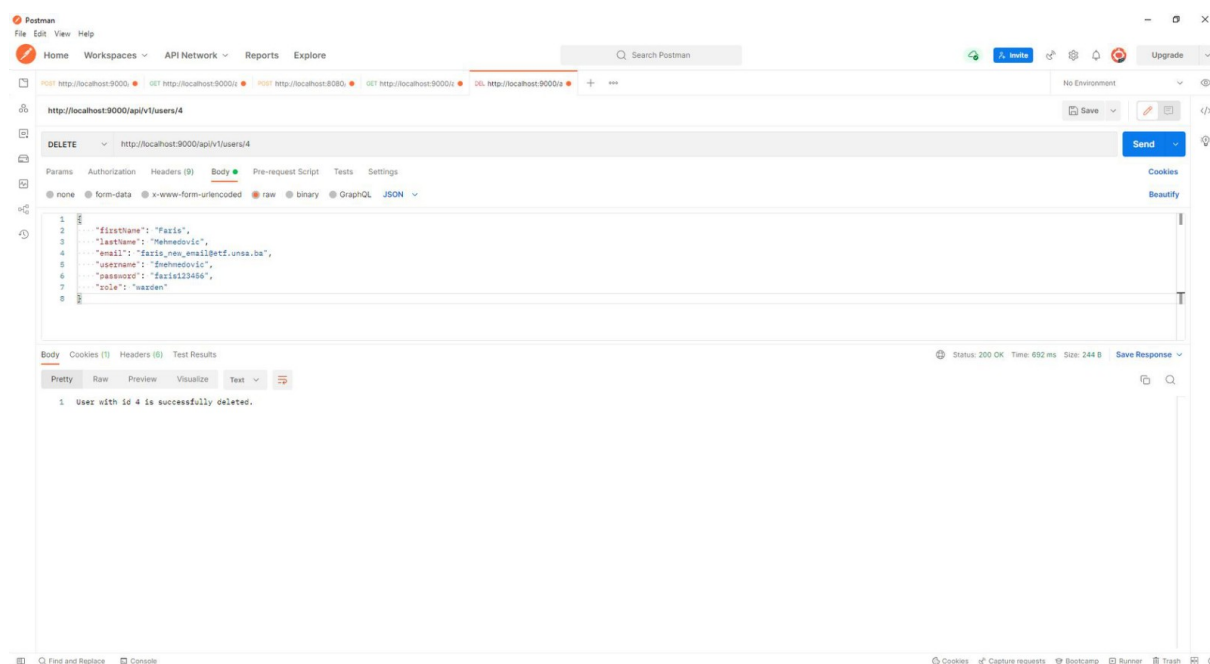
Slika 3.3: GET user by id



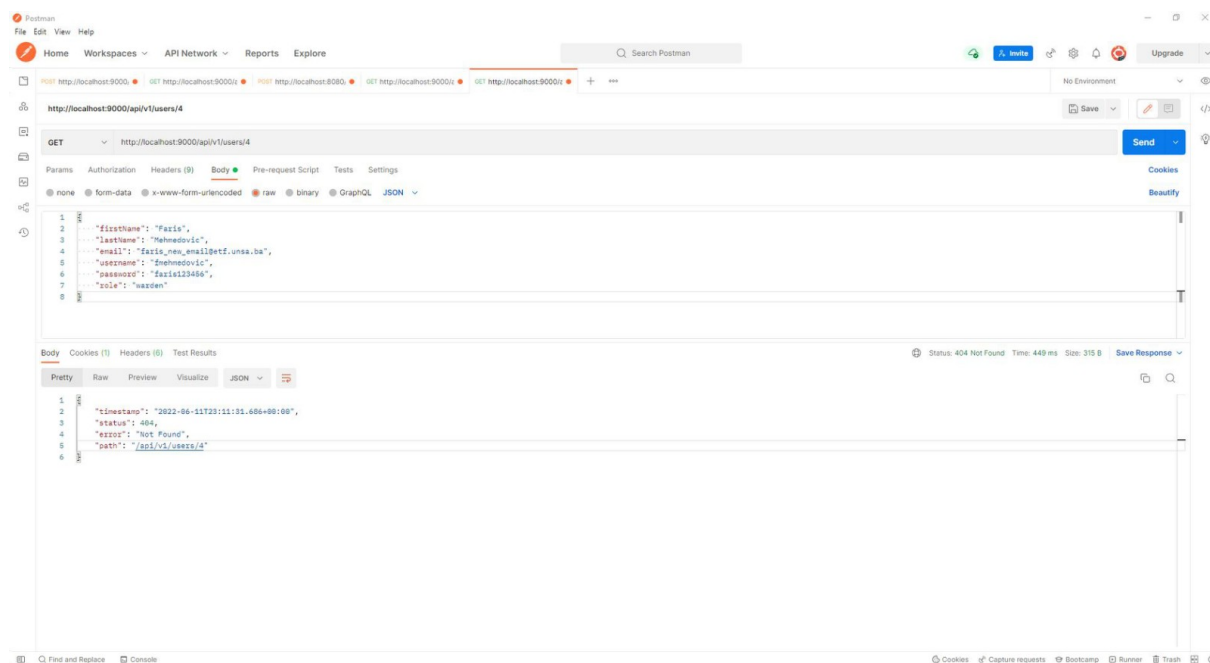
Slika 3.4: POST user



Slika 3.5: PUT user

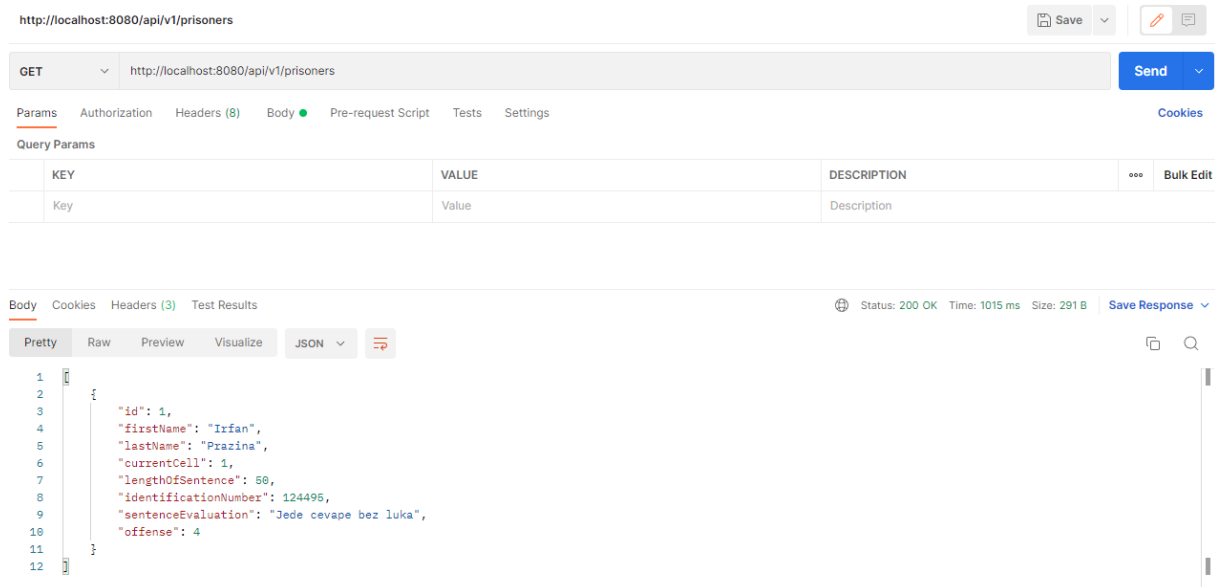


Slika 3.6: DELETE user

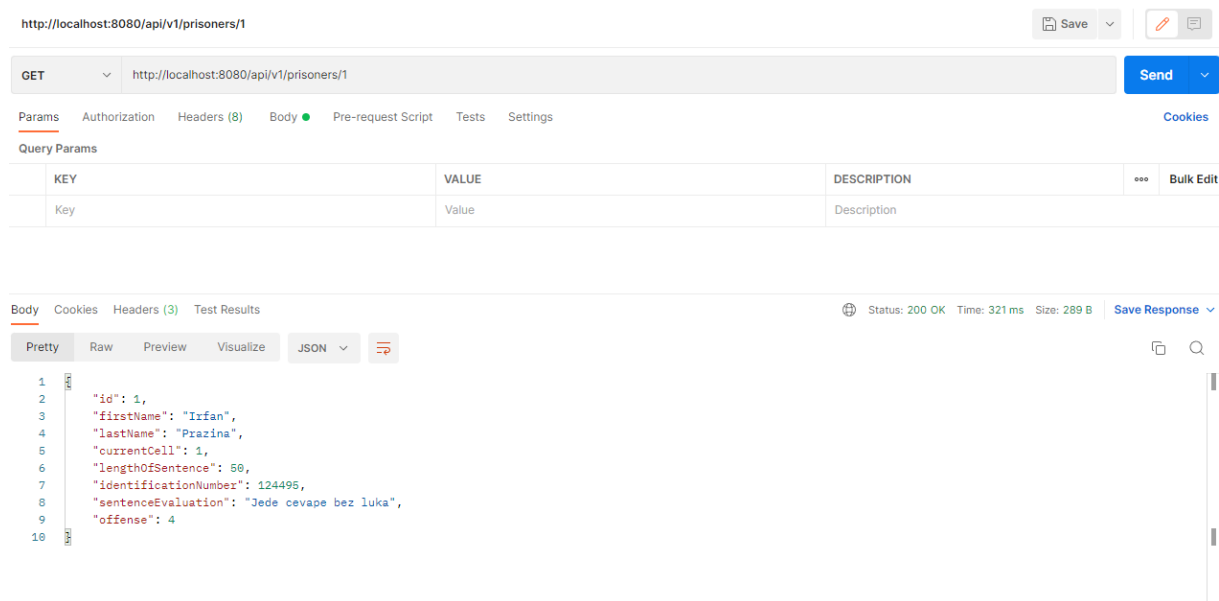


Slika 3.7: GET user by id (slučaj neuspješnog zahtjeva-odgovora)

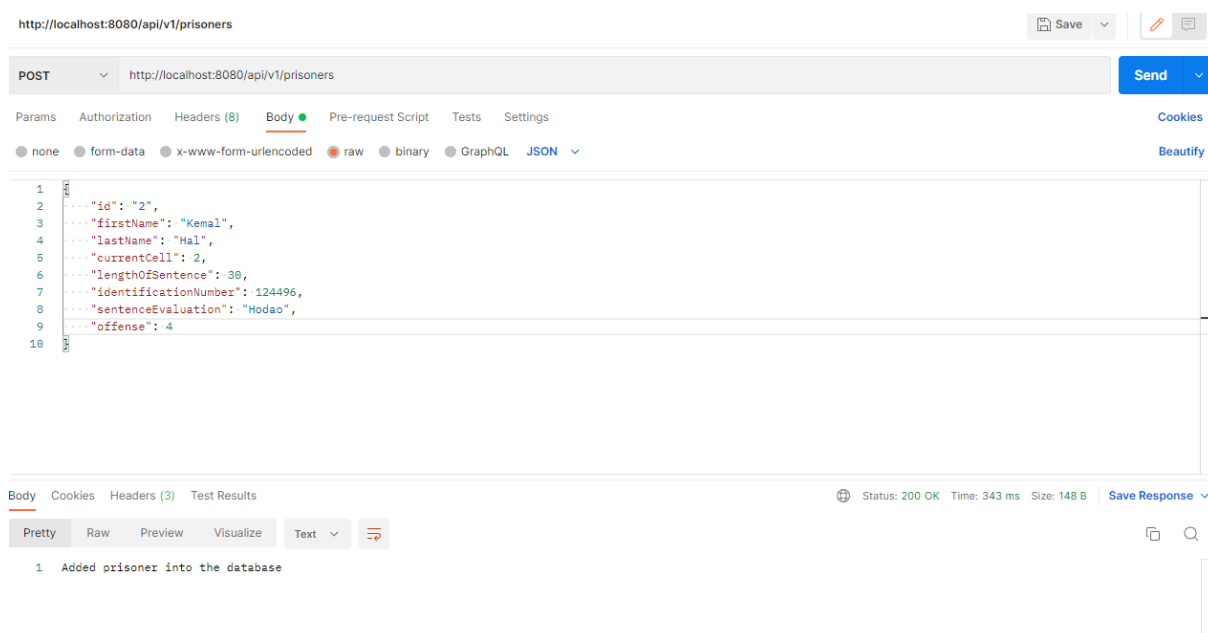
3.2 Testiranje putem Postman-a za prisoner-service



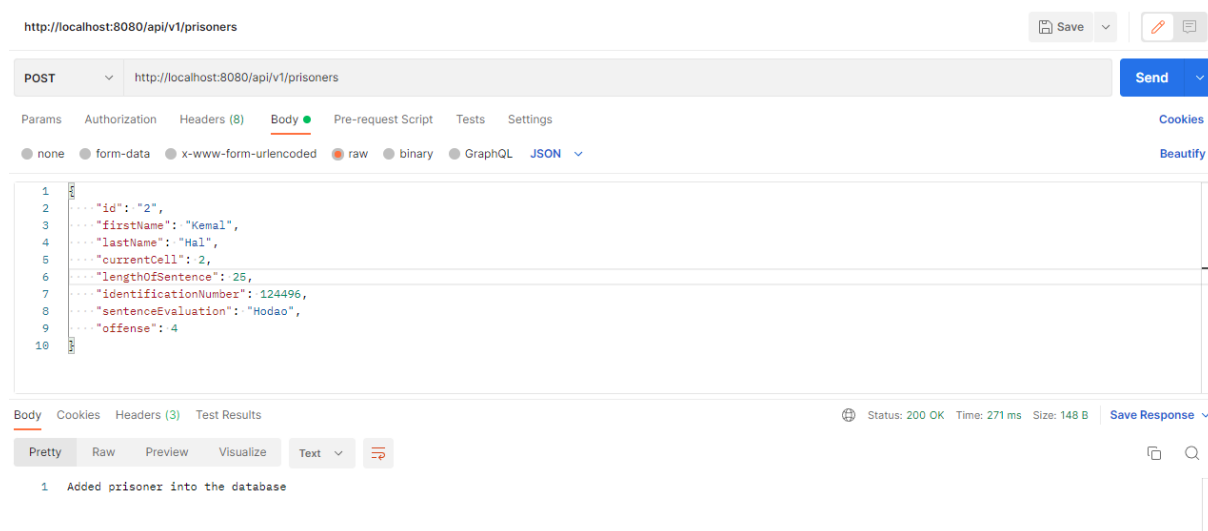
Slika 3.8: GET all prisoners



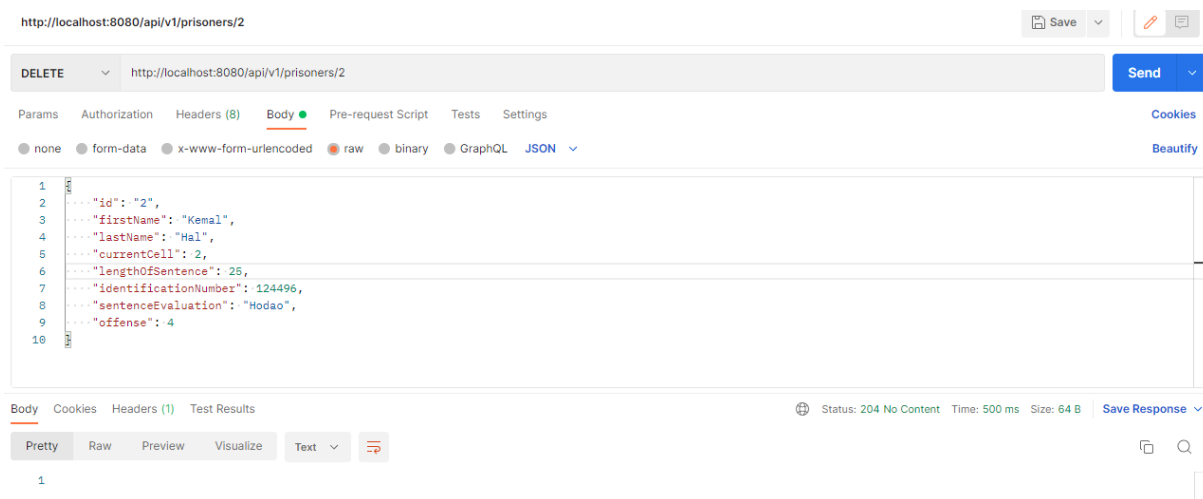
Slika 3.9: GET prisoner by id



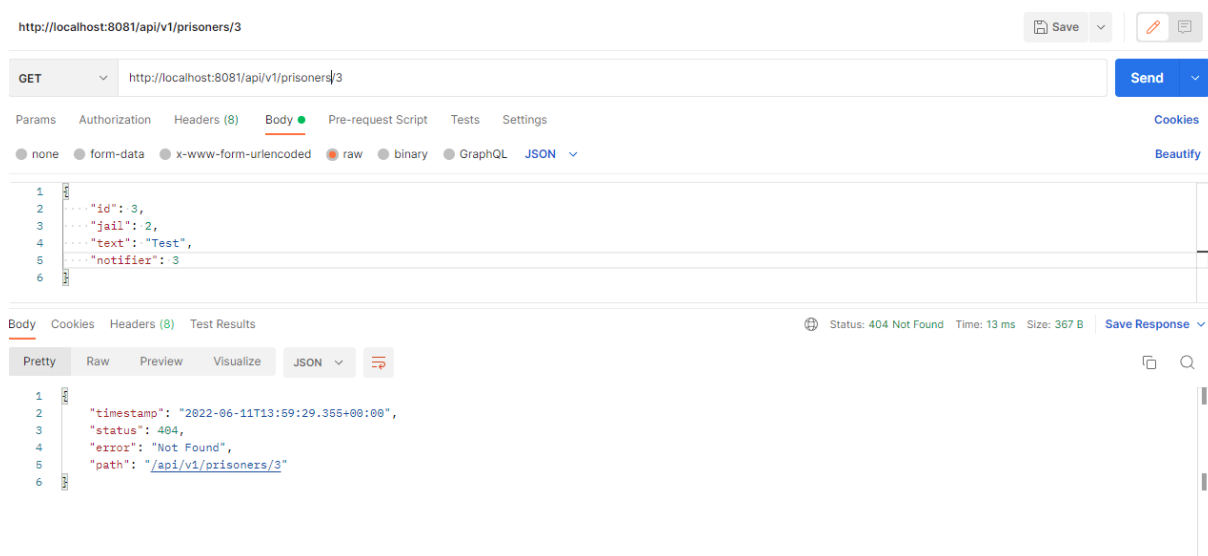
Slika 3.10: POST prisoner



Slika 3.11: PUT prisoner

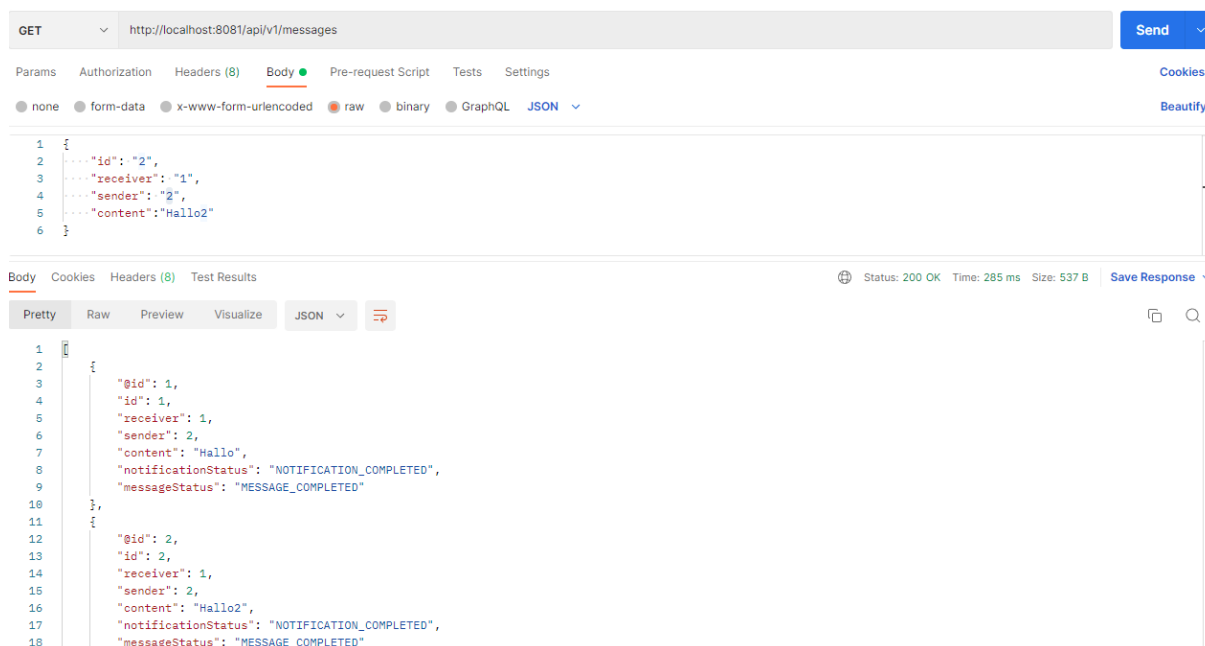


Slika 3.12: DELETE prisoner

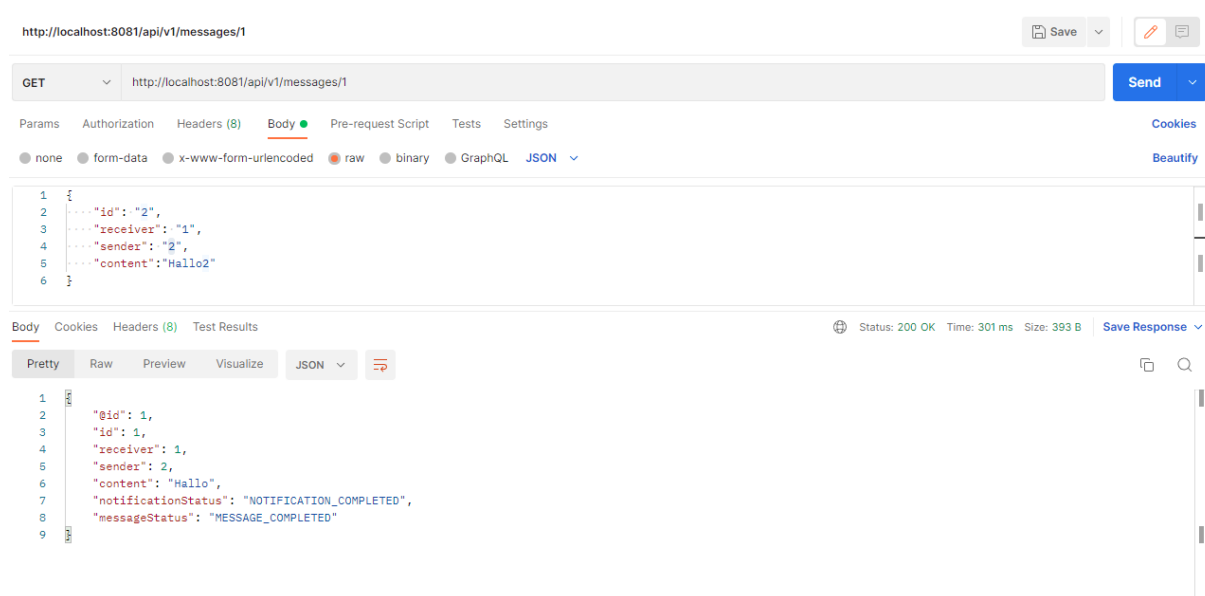


Slika 3.13: GET prisoner by id (slučaj neuspješnog zahtjeva-odgovora)

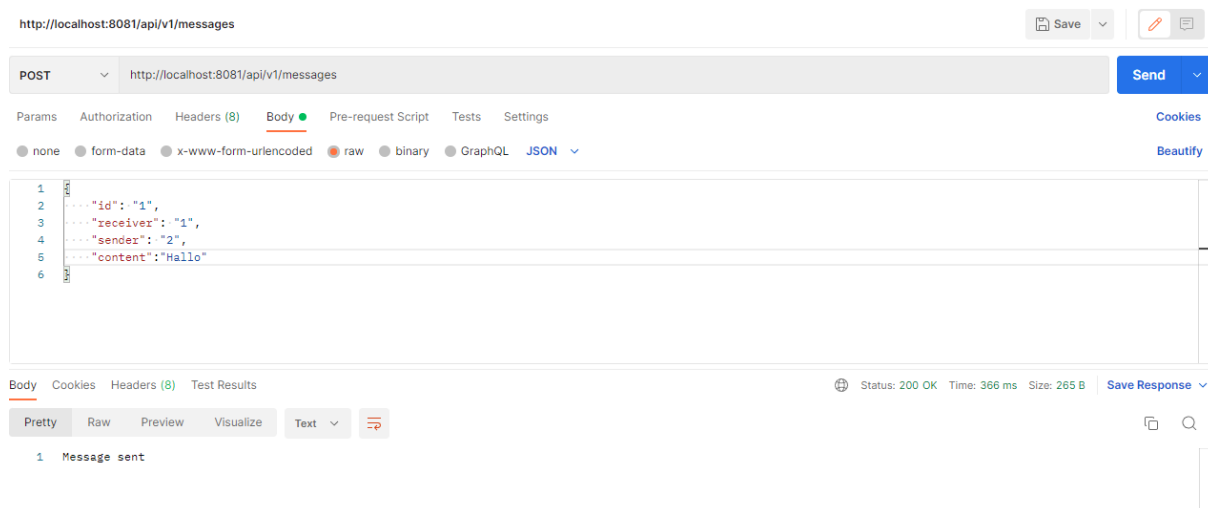
3.3 Testiranje putem Postman-a za message-service



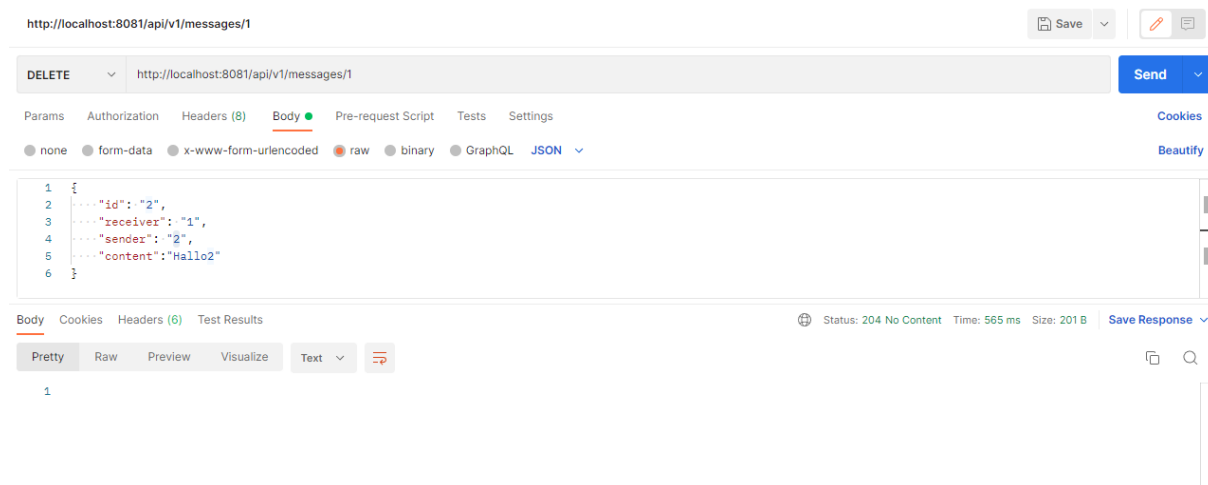
Slika 3.14: GET all messages



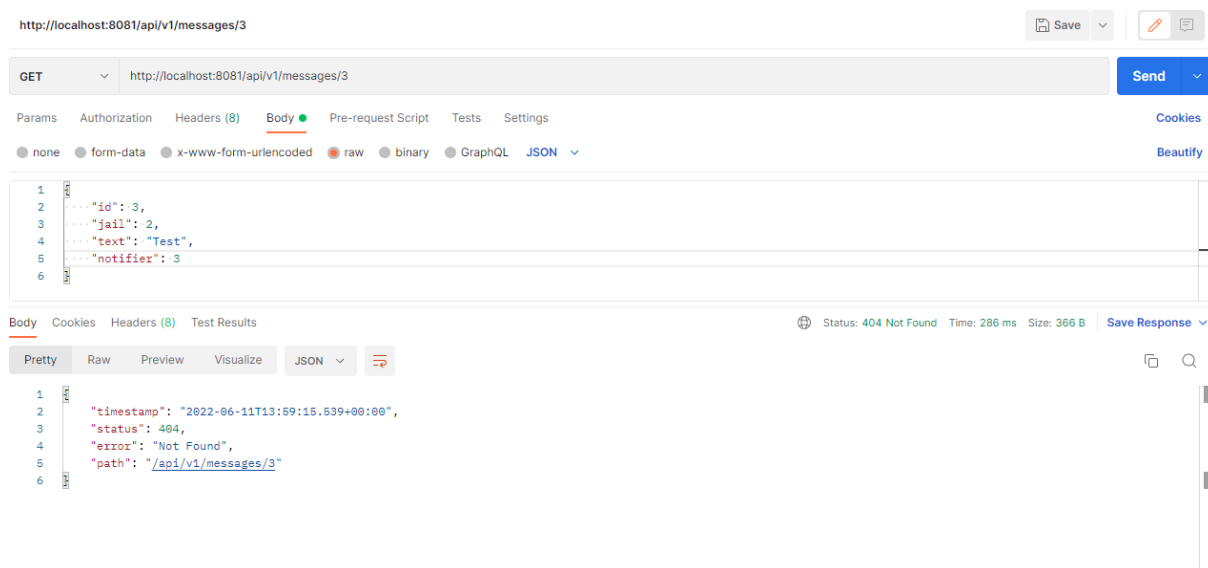
Slika 3.15: GET message by id



Slika 3.16: POST message

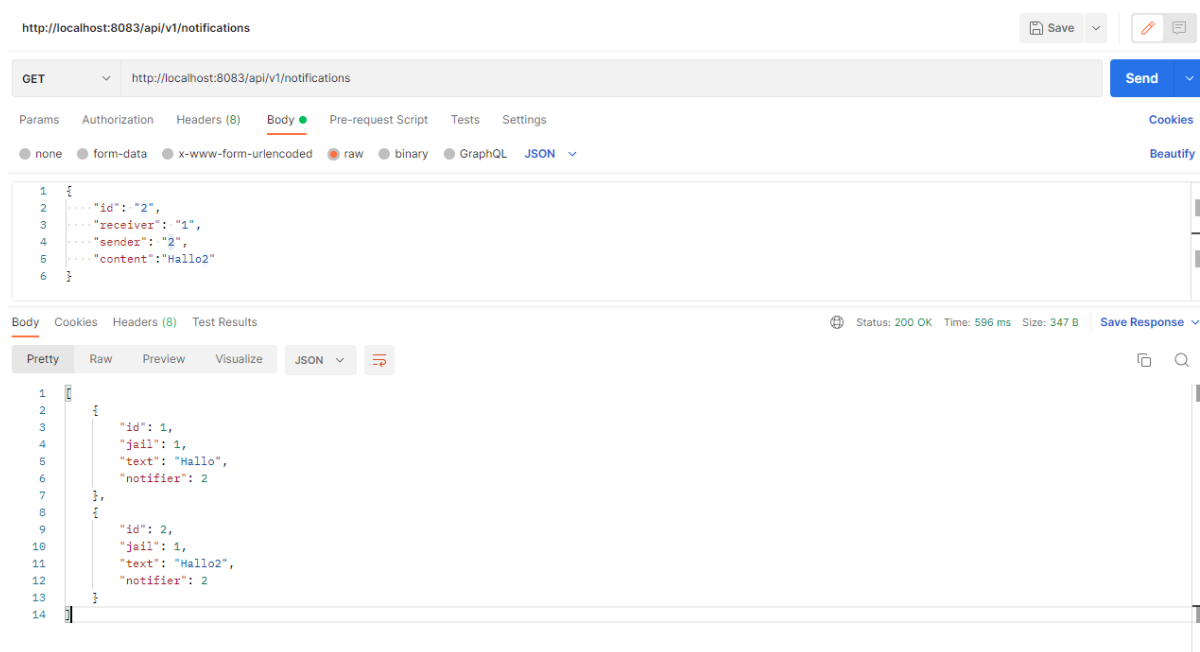


Slika 3.17: DELETE message

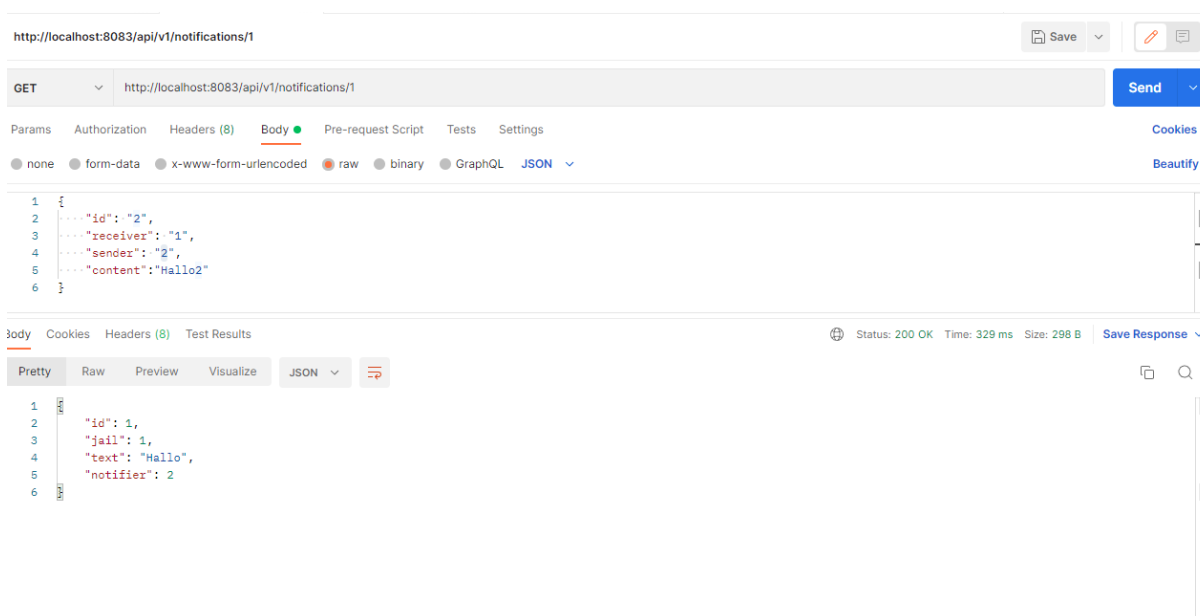


Slika 3.18: GET message by id (slučaj neuspješnog zahtjeva-odgovora)

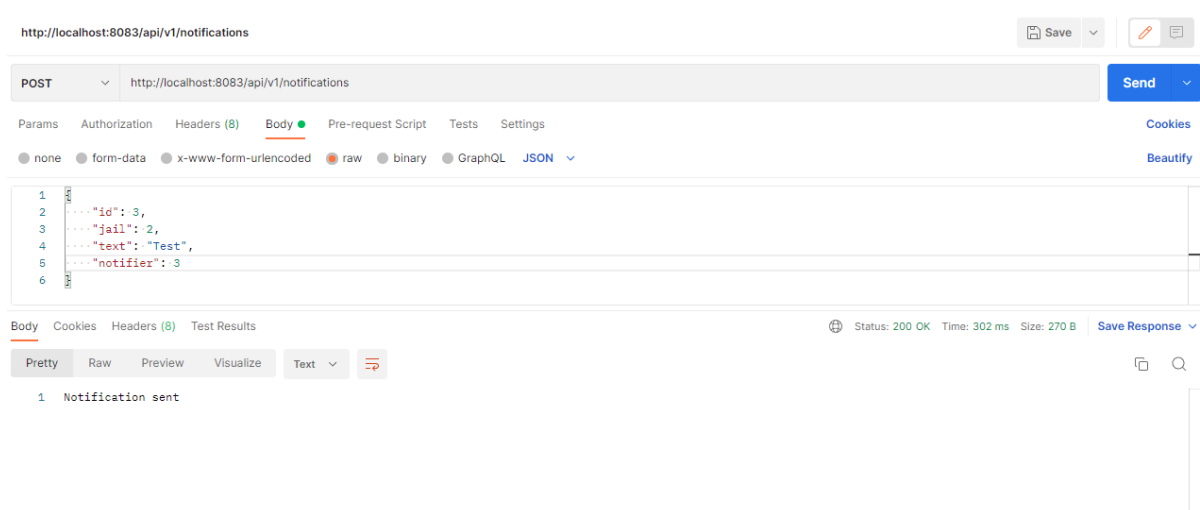
3.4 Testiranje putem Postman-a za notification-service



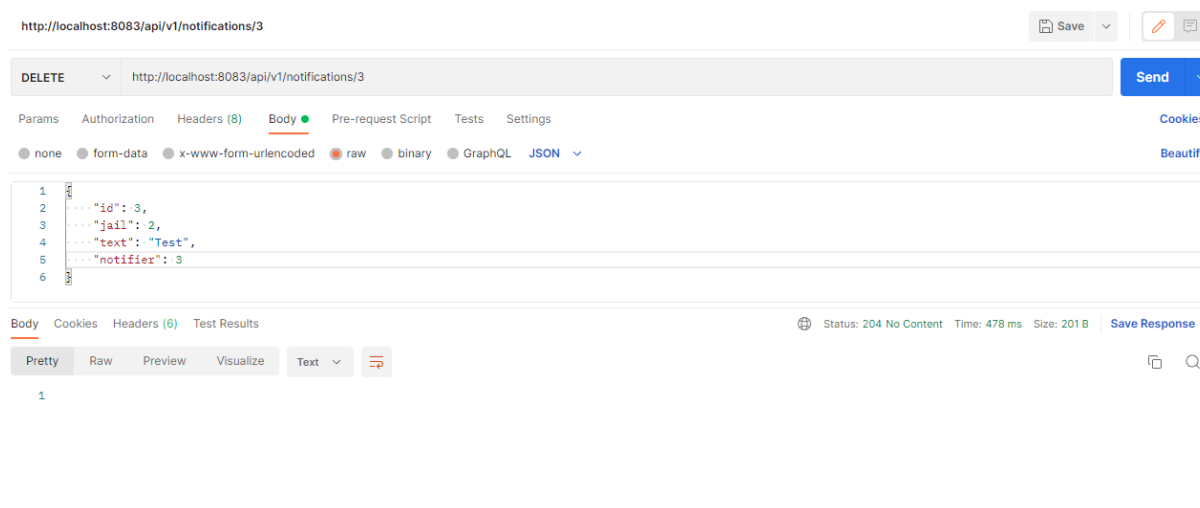
Slika 3.19: GET all notifications



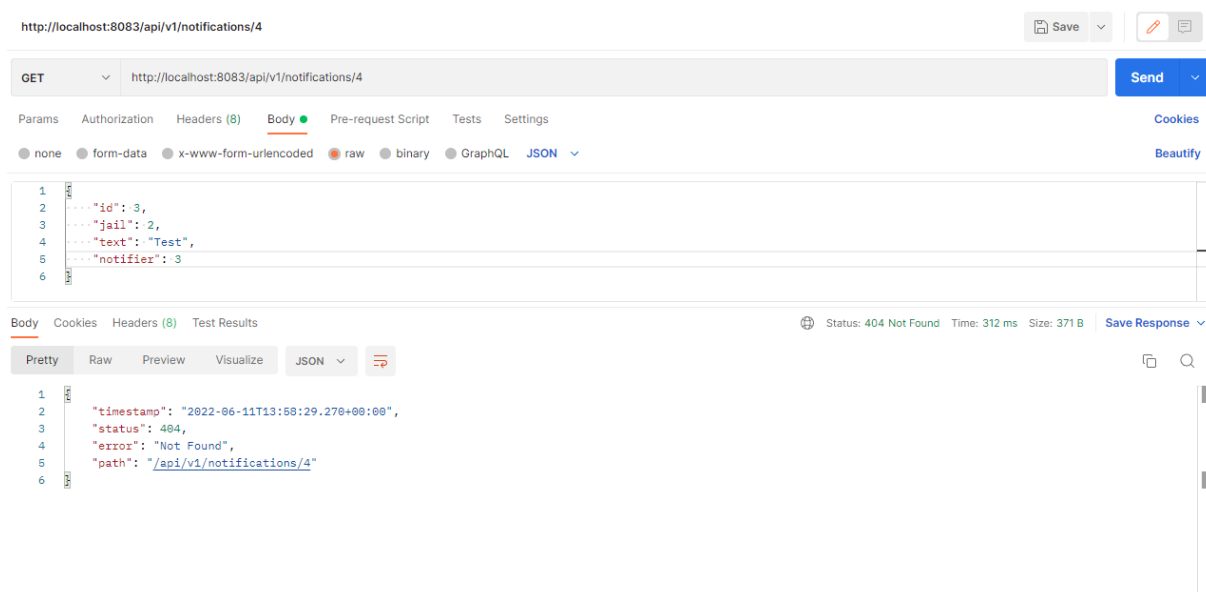
Slika 3.20: GET notification by id



Slika 3.21: POST notification



Slika 3.22: DELETE notification



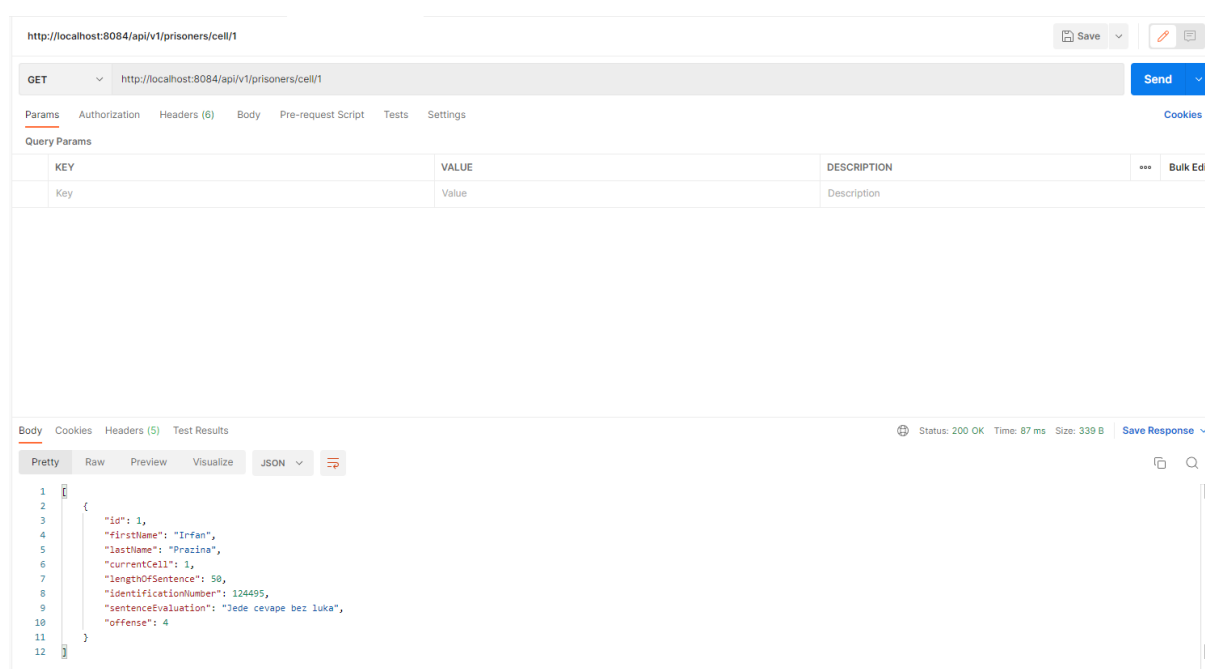
Slika 3.23: GET notification by id (slučaj neuspješnog zahtjeva-odgovora)

4. Zadatak 4

4.1 Netrivijalni webservisi

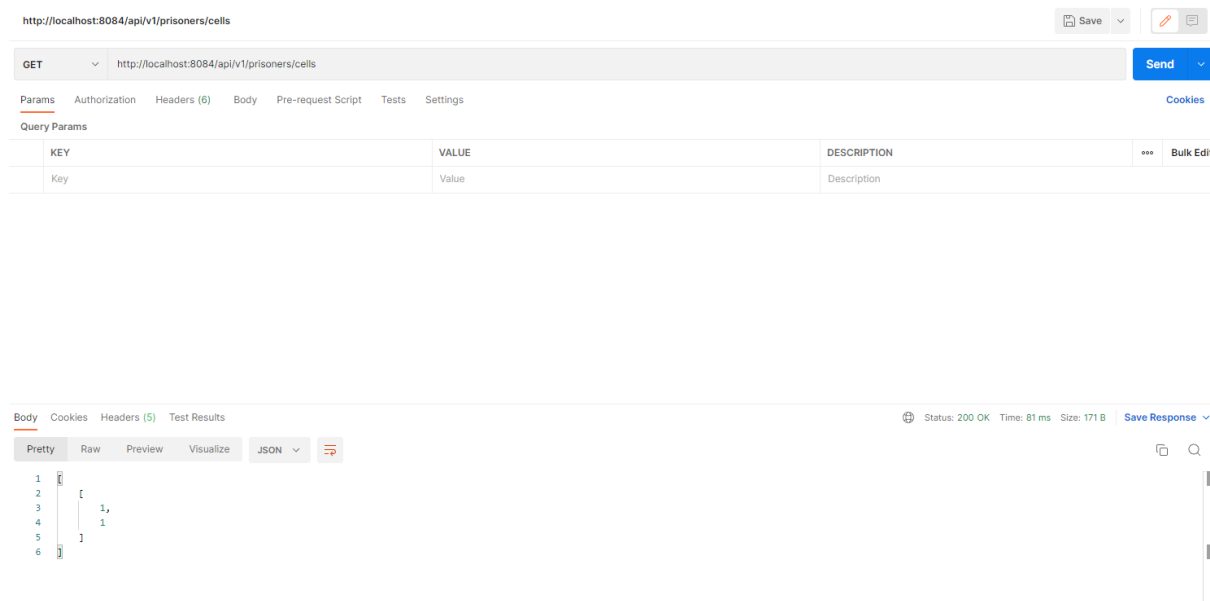
U zadatku 4 implementirani su netrivijalni webservisi - koji nisu automatski generisani iz metoda repozitorija (napravljeni su složeni upiti nad bazom bez komunikacije sa drugim mikroservisima). U ovoj sekciji navedene stavke će biti prikazane kroz Postman testiranje.

Na sljedećoj slici je prikazan pristup zatvorenicima preko id-a ćelije (u kojoj borave).



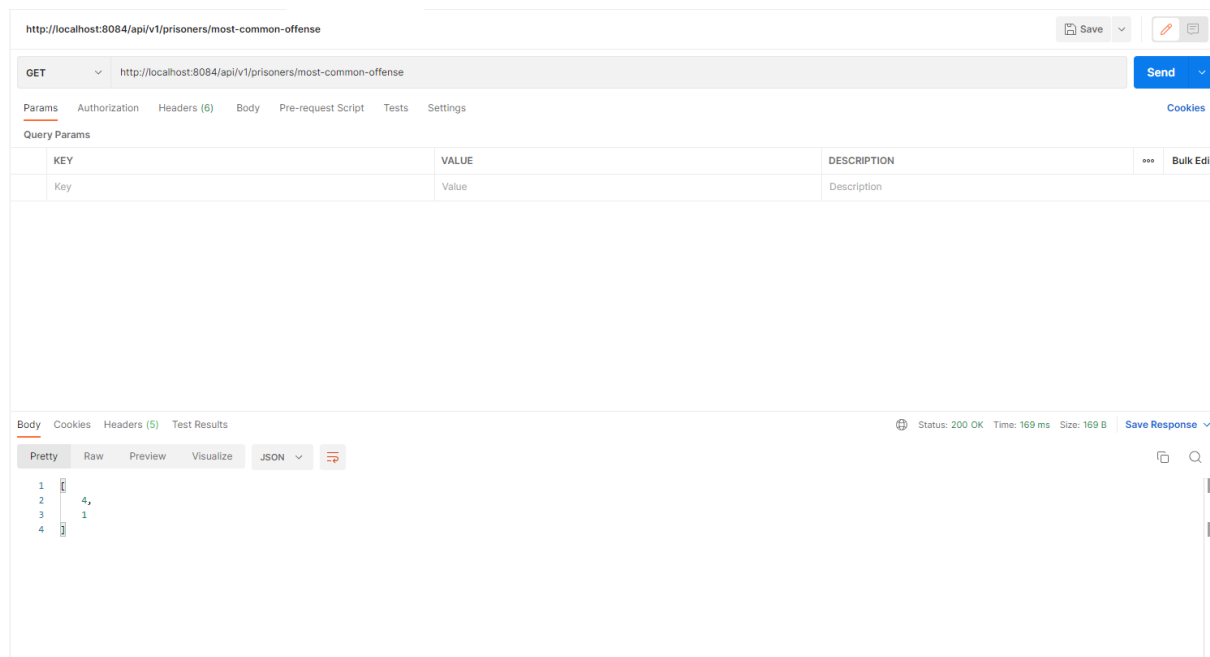
Slika 4.1: GET prisoners by cell id

Na sljedećoj slici je prikazane su ćelije sa brojem zatvorenika koji borave u zauzetim ćelijama.



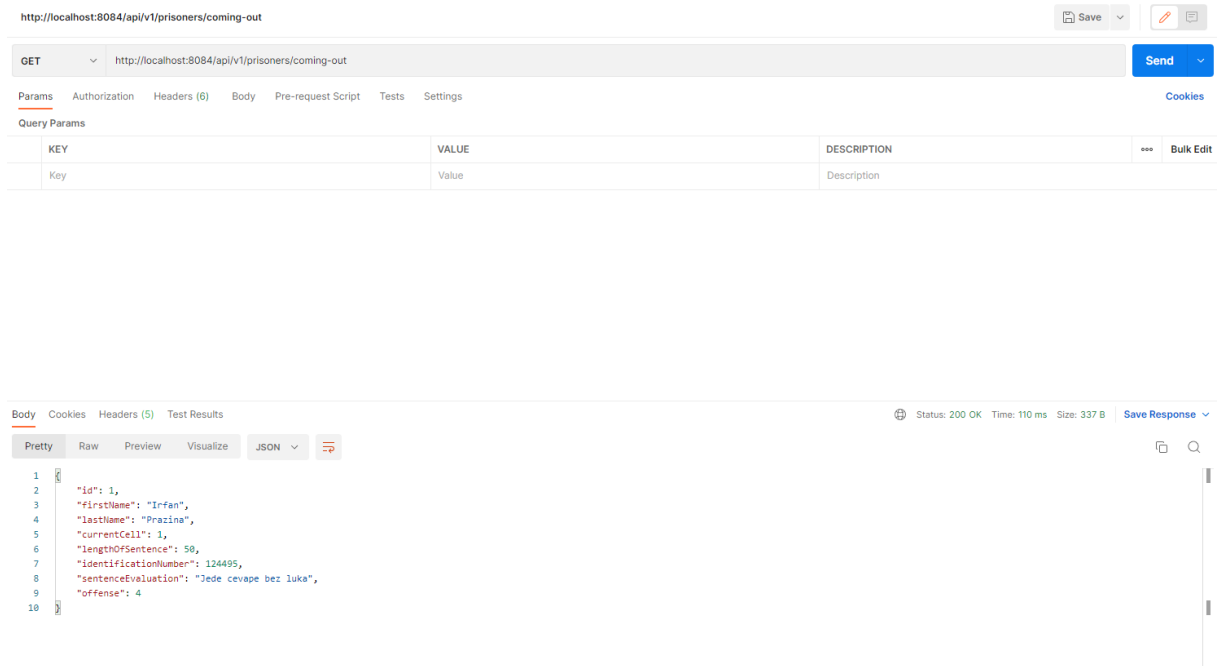
Slika 4.2: GET prisoners/cells

Na sljedećoj slici je prikazani su najčešći prekršaji, radi kojih zatvorenici borave u zatvoru.



Slika 4.3: GET prisoners/most-common-offense

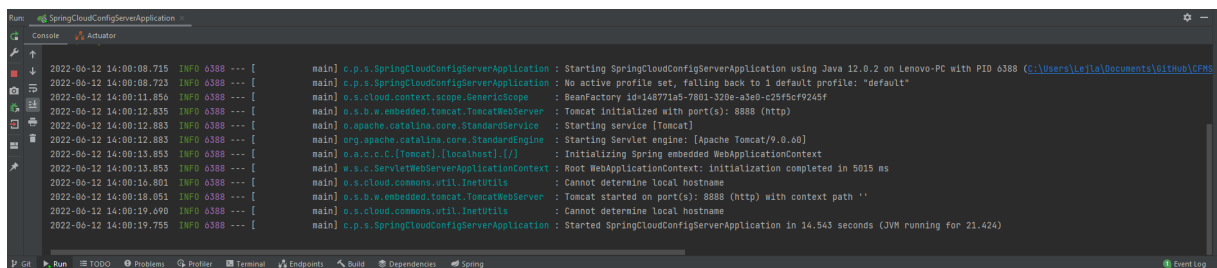
Na sljedećoj slici je prikazan je zatvorenik, koji ima najkraću kaznu služenja.



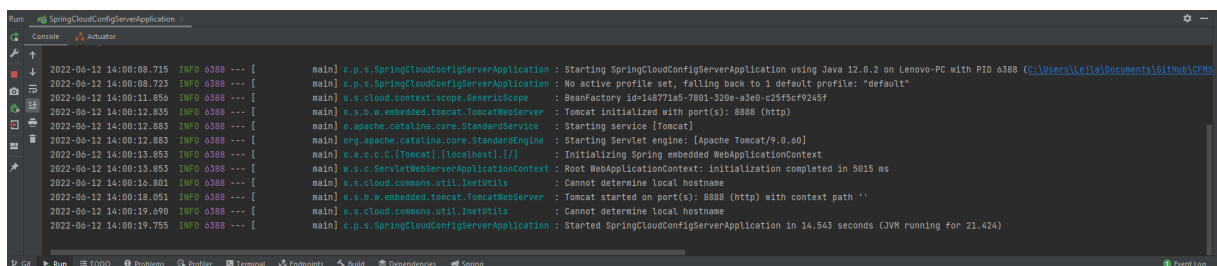
Slika 4.4: GET prisoners/coming-out

4.2 Service discovery (Eureka) i centralizirana konfiguracija

Unutar ove sekcije prikazano je pokretanje centralizirane konfiguracije (spring-cloud-config-server) i Eureka servera (discovery-service).



Slika 4.5: Pokretanje centralizirane konfiguracije



Slika 4.6: Pokretanje Eureka servera

Na sljedećoj slici prikazan je Eureka server na adresi <http://localhost:8761> sa pokrenutom instancom prisoner-service.

The screenshot displays the Spring Eureka server interface. At the top, the 'spring Eureka' logo is visible alongside navigation links for 'HOME' and 'LAST 1000 SINCE STARTUP'. The 'System Status' section contains two tables. The left table lists 'Environment' as 'N/A' and 'Data center' as 'N/A'. The right table shows 'Current time' as '2022-06-12T13:34:08 +0200', 'Uptime' as '00:14', 'Lease expiration enabled' as 'false', 'Renews threshold' as '3', and 'Renews (last min)' as '2'. Below this, a red emergency message states: 'EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.' The 'DS Replicas' section indicates 'Instances currently registered with Eureka'. A table below shows one instance: 'PRISONER-SERVICE' with 'n/a (1)' AMIs, '(1)' Availability Zones, and a status of 'UP (1) - prisoner-service:39d56c0e-1711-49e8-8a58-4d19119b4a39'. The 'General Info' section features a table with system metrics: 'total-avail-memory' (65mb), 'num-of-cpus' (4), 'current-memory-usage' (27mb (41%)), 'server-uptime' (00:14), and counts for 'registered-replicas', 'unavailable-replicas', and 'available-replicas'.

spring Eureka		HOME	LAST 1000 SINCE STARTUP
System Status			
Environment	N/A	Current time	2022-06-12T13:34:08 +0200
Data center	N/A	Uptime	00:14
		Lease expiration enabled	false
		Renews threshold	3
		Renews (last min)	2
EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.			
DS Replicas			
Instances currently registered with Eureka			
Application	AMIs	Availability Zones	Status
PRISONER-SERVICE	n/a (1)	(1)	UP (1) - prisoner-service:39d56c0e-1711-49e8-8a58-4d19119b4a39
General Info			
Name	Value		
total-avail-memory	65mb		
num-of-cpus	4		
current-memory-usage	27mb (41%)		
server-uptime	00:14		
registered-replicas			
unavailable-replicas			
available-replicas			

Slika 4.7: Prikaz Eureka servera

5. Zadatak 5

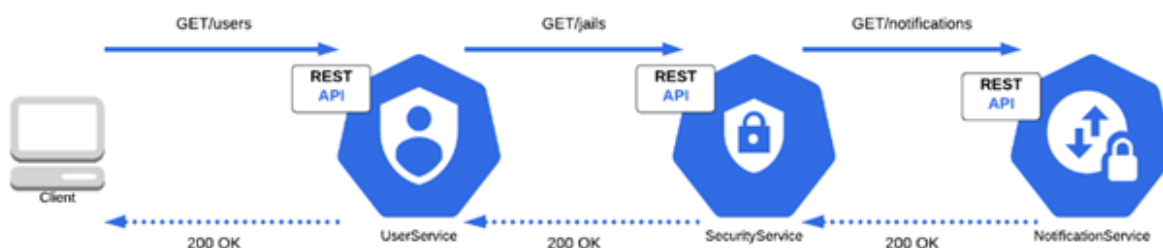
U sljedećem zadatku prikazani su dijagrami toka komunikacije za bar jednu netrivialnu komunikaciju. Netrivialna komunikacija podrazumijeva onu čiji rezultat zavisi od stanja baza i/ili memorije dva ili više mikroservisa

5.1 Dijagram jail → prisoner



Slika 5.1: Dijagram jail → prisoner

5.2 Dijagram user → jail → notification



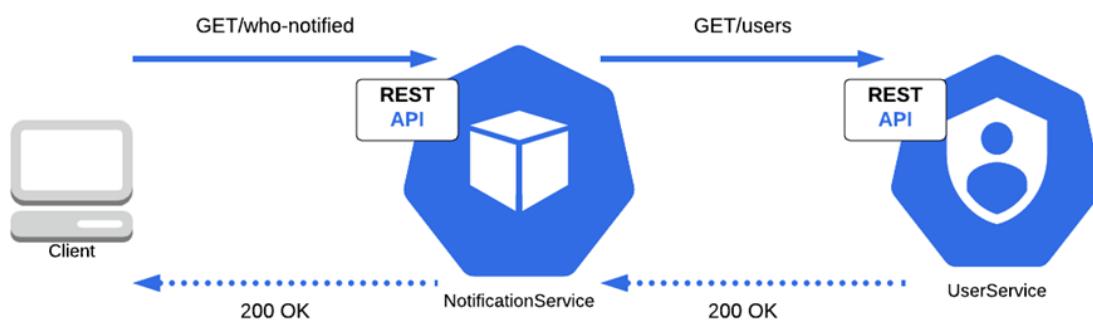
Slika 5.2: Dijagram user → jail → notification

5.3 Dijagram user → message



Slika 5.3: Dijagram user → message

5.4 Dijagram notification → user



Slika 5.4: Dijagram notification → user

6. Zadatak 6

U ovom zadatku potrebno je bilo napraviti system-events service koji će pratiti akcije koje dešavaju u sistemu. Akcija predstavlja bilo koji oblik promjene ili dohvaćanja podataka. U ovom servisu se čuvaju sljedeći podaci: timestamp akcije, naziv mikroservisa u kojem se akcija desila, user koji je pokrenuo akciju. Komunikacija mikroservisa sa system-events service je urađena putem gRPC-a.

```
Saved log: Prisoner, Sun Jun 12 20:18:51 CEST 2022, prisoner, GET, 200
Saved log: Prisoner, Sun Jun 12 20:19:01 CEST 2022, prisoner, POST, 200
Saved log: Prisoner, Sun Jun 12 20:19:18 CEST 2022, prisoner, GET, 200
Saved log: Prisoner, Sun Jun 12 20:19:21 CEST 2022, prisoner, GET, 404
Saved log: Prisoner, Sun Jun 12 20:19:21 CEST 2022, prisoner, GET, 404
Saved log: Notification, Sun Jun 12 20:20:12 CEST 2022, notification, GET, 200
Saved log: Notification, Sun Jun 12 20:20:14 CEST 2022, notification, GET, 200
Saved log: Notification, Sun Jun 12 20:20:34 CEST 2022, notification, POST, 405
Saved log: Notification, Sun Jun 12 20:20:38 CEST 2022, notification, POST, 200
```

Slika 6.1: Prikaz log-a za event-e

id	service_name	resource	response	timestamp	type
1	Prisoner	prisoner	200	Sun Jun 12 20:15:17 CEST 2022	GET
2	Messages	message	200	Sun Jun 12 20:17:19 CEST 2022	GET
3	Prisoner	prisoner	200	Sun Jun 12 20:18:51 CEST 2022	GET
4	Prisoner	prisoner	200	Sun Jun 12 20:19:01 CEST 2022	POST
5	Prisoner	prisoner	200	Sun Jun 12 20:19:18 CEST 2022	GET
6	Prisoner	prisoner	404	Sun Jun 12 20:19:21 CEST 2022	GET
7	Prisoner	prisoner	404	Sun Jun 12 20:19:21 CEST 2022	GET
8	Notification	notification	200	Sun Jun 12 20:20:12 CEST 2022	GET
9	Notification	notification	200	Sun Jun 12 20:20:14 CEST 2022	GET
10	Notification	notification	405	Sun Jun 12 20:20:34 CEST 2022	POST
11	Notification	notification	200	Sun Jun 12 20:20:38 CEST 2022	POST
12	User	user	200	Sun Jun 12 20:31:33 CEST 2022	GET
13	User	user	405	Sun Jun 12 20:31:47 CEST 2022	POST

Slika 6.2: Prikaz event-ova u MySQL Workbench

7. Zadatak 7

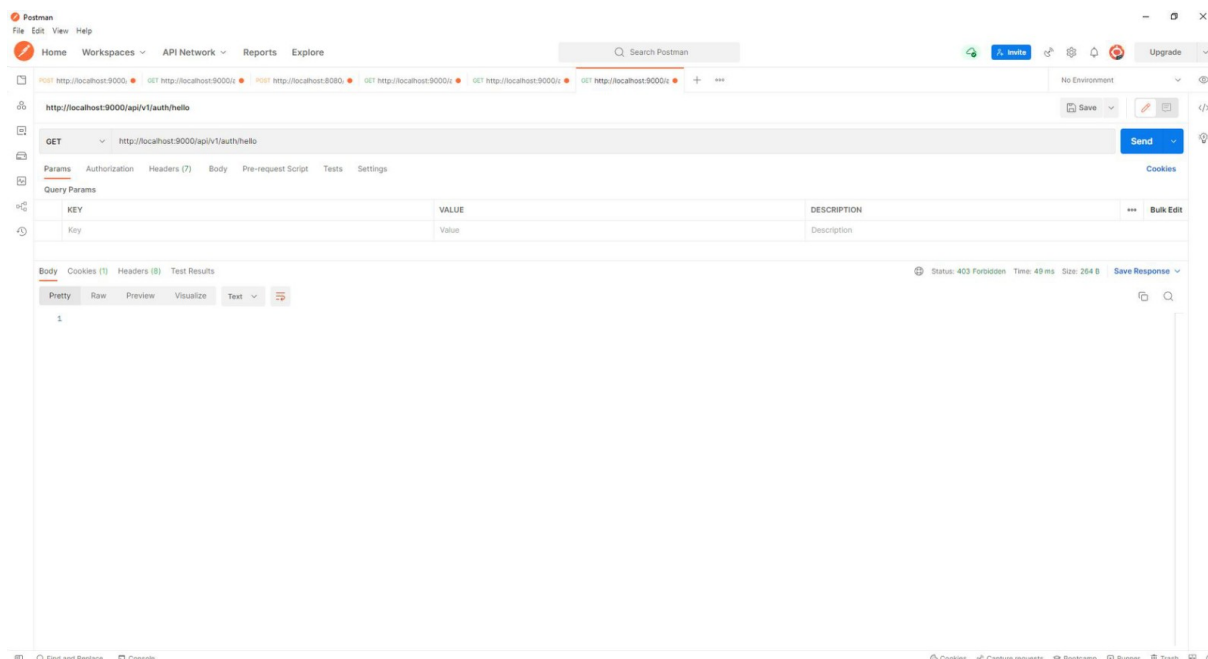
7.1 Analiza rješenja za sigurnost sistema

U ovom zadatku istražena su razna rješenja za sigurnost sistema baziranih na mikroservisnoj arhitekturi, te napisan je kratka pregled o predloženom rješenju sigurnosti aplikacije "Menadžment sistem za kazneno-popravni zavod".

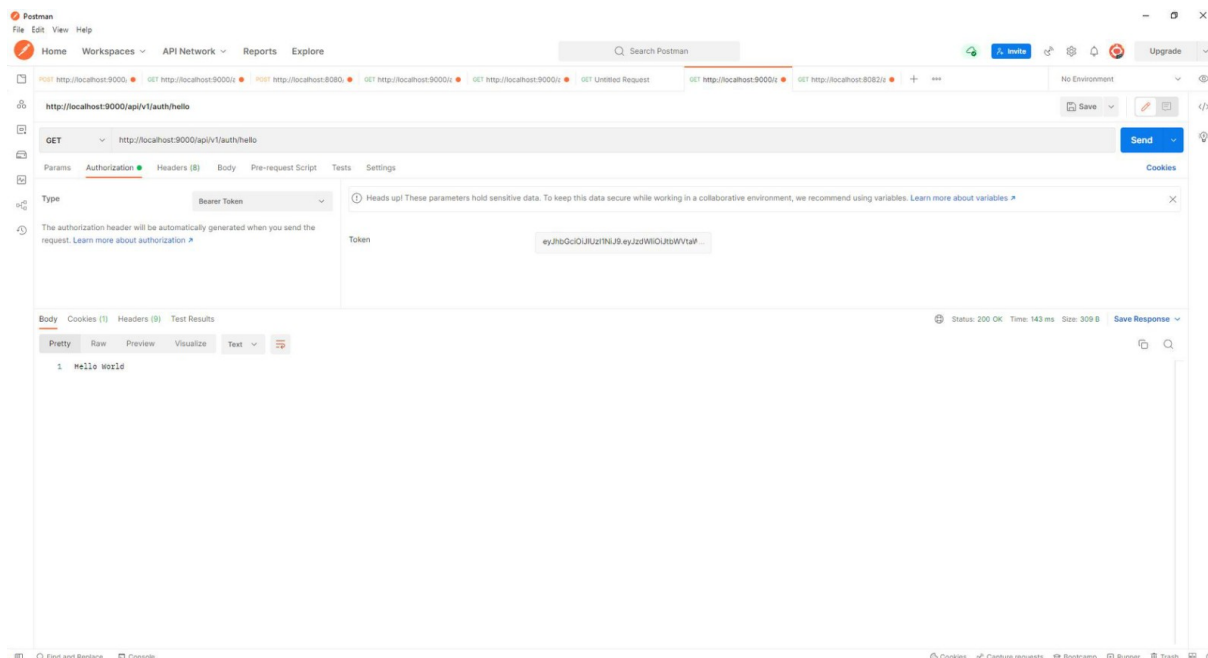
- O autentifikaciji će voditi računa API Gateway, jer će svaki call na backend ići preko gateway-a
- Pri autentifikaciji koristit će se tokeni, konkretno JWT token. Glavni razlog njegove upotrebe je zbog podrške koju je dobio na svim platformama i provjerenoj sigurnosti istog
- Što se tiče uloga (eng. *roles*) i dopuštenja (eng. *permission*), koristi se zaseban mikroservis auth-service i na njemu je kreiran endpoint u kontroleru koji se authenticate i izvršava se na port-u 1234 (<http://localhost:1234/authenticate>). Na ovom endpoint-u, šalje se post zahtjev, zajedno sa username i password. Ukoliko postoji user, vraća se token, onda je na korisniku Postmana, da prilikom svakom zahtjeba pošalje token u header-u.
- (De)centralizirana autorizacija
- Nije potrebna autorizacija između mikroservisa jer je jedan mikroservis uvijek "jači" od drugog. Odnosno, ukoliko korisnik je privilegovao da koristi mikroservis A, a taj mikroservis poziva mikroservis B, sigurno je da bi korisnik mogao i direktno da koristi mikroservis B. Pristup izvana pojedinačnim mikroservisima je ograničen, odnosno tema aplikacije predstavlja nešto što je confidential, i ne bi bilo poželjno previše pokazati široj javnosti.
- Prilikom logout-a token će biti uništen na klijentskoj strani, odnosno u browseru. Pobrinut ćemo se da trajanje tokena ne bude predugo, da se tokeni dodaju na blocklistu u slučaju neprirodnih radnji na profilu. Naravno, vršit će se i refreshovanje tokena da ne bi došlo do isteka sesije, odnosno da user se mora ponovo ulogovat u sistem.
- S obzirom da nije u planu razvoj mobilne varijante aplikacije, neće biti user friendly UI-a za mobilne aplikacije.

7.2 Prikaz implementirane sigurnosti

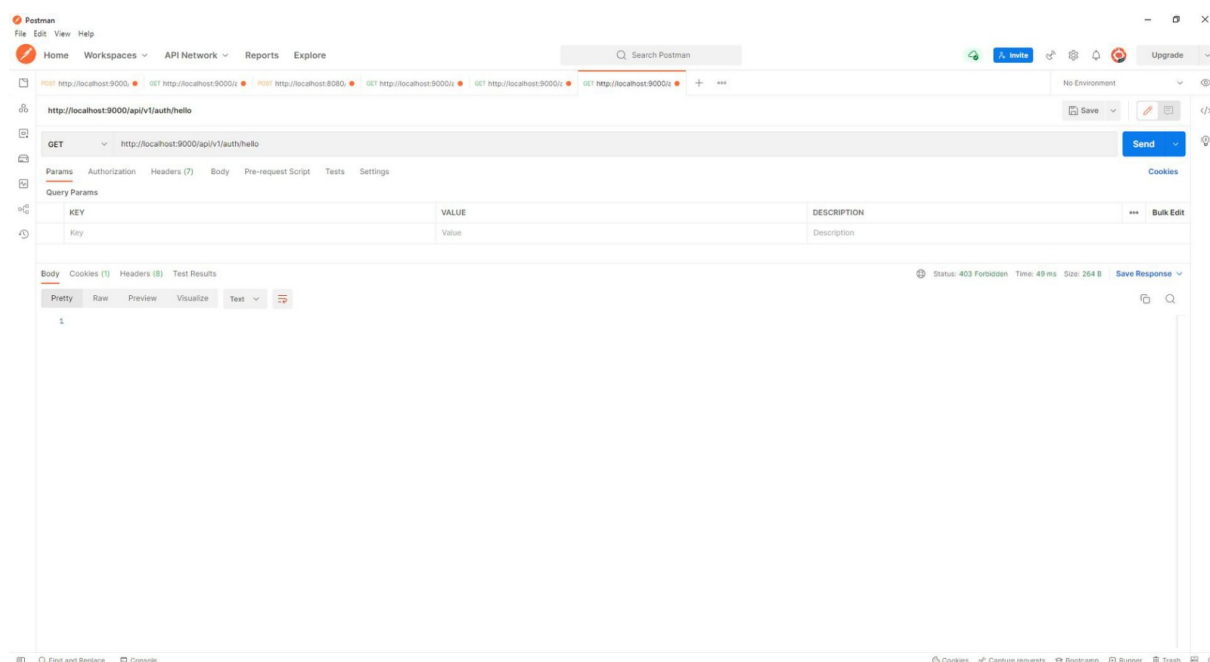
U narednoj sekciji prikazana je implementirana sigurnost koristeći Postman testiranje. Navedeni testovi se referiraju na drugu stavku u prethodnoj sekciji.



Slika 7.1: POST auth/login (autorizacija, dobivanje tokena)

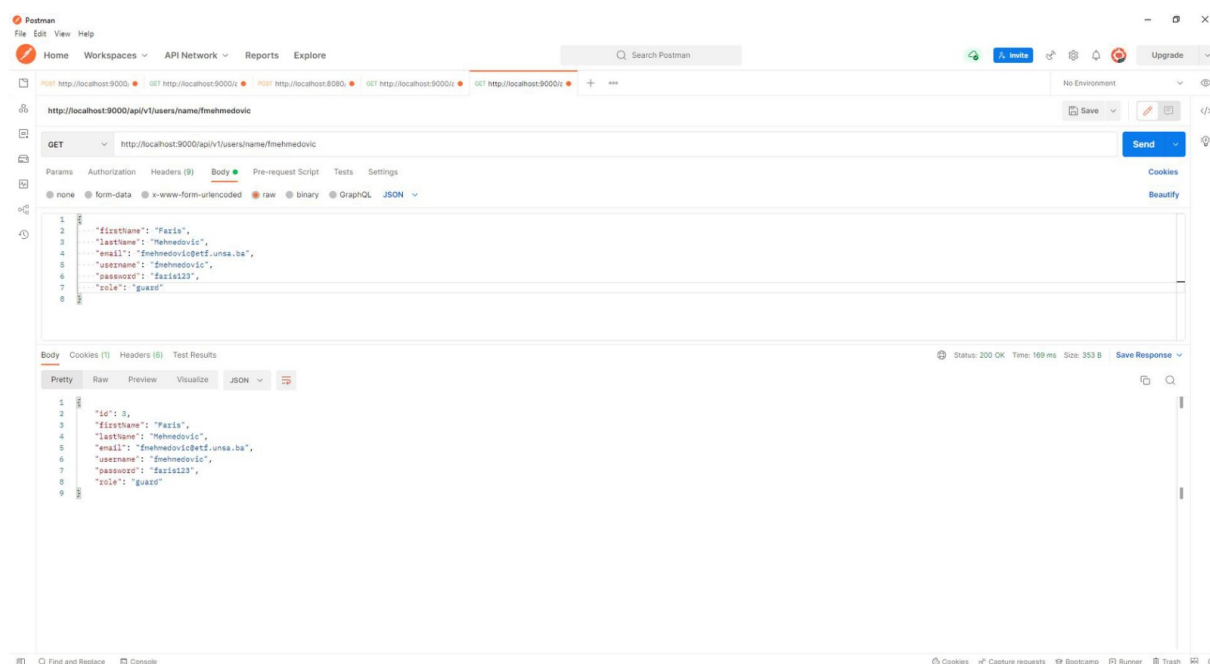


Slika 7.2: GET auth/hello (authorized)



Slika 7.3: GET auth/hello (unauthorized)

Kada korisnik unese username i pwd, aplikacija interno pošalje zahtjev na endpoint, ukoliko se poklapaju lozinke (eng. *password*) onda se dobija token, u suprotnom dolazi do greške.



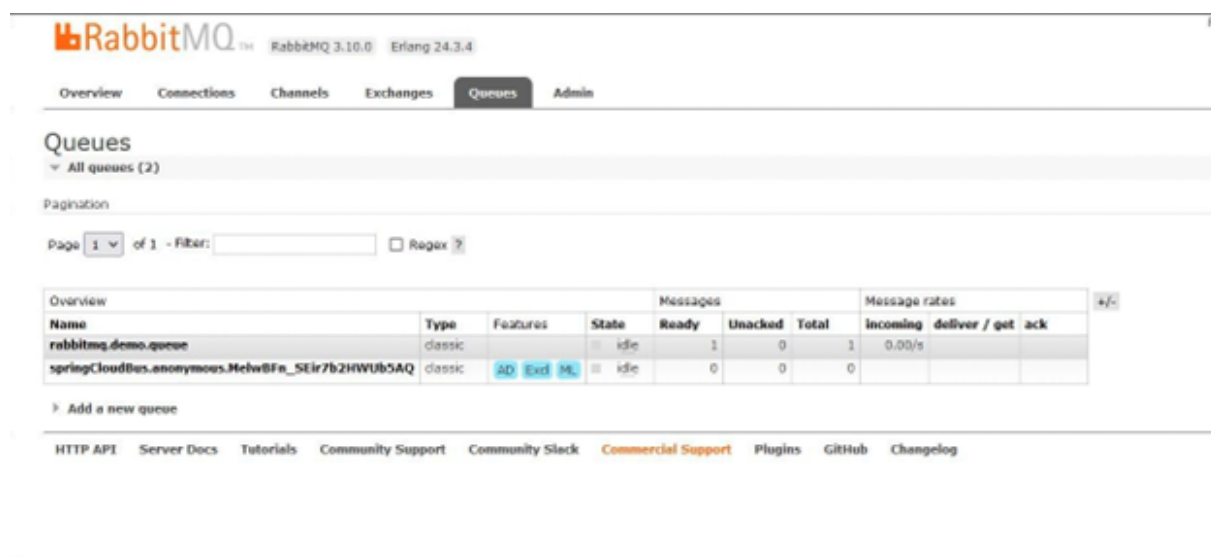
Slika 7.4: GET user by username

8. Zadatak 8

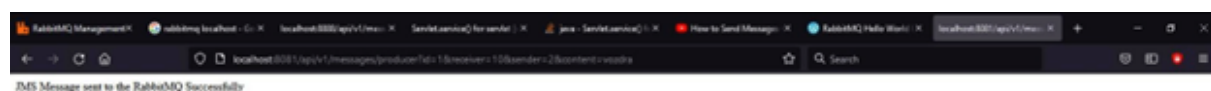
U zadatku 8 neophodno je bilo za bar jednu funkcionalnost implementirati asinhronu komunikaciju koristeći RabbitMQ. Pri implementaciji potrebno je bilo obratiti pažnju na to da postoje i inverzne akcije koje će vratiti bazu u početno stanje ako dođe do greške i zahtjev ne bude zadovoljen. Korišten je event-based pristup/saga choreography. U ovoj komunikaciji trebaju biti bar dva upisa u bazu u različitim mikroservisima (bar dvije lokalne transakcije) i obje trebaju zavisiti jedna od druge tako da ako jedna padne i druga poziva inverznu akciju i obrnuto.

8.1 Implementacija asinhronne komunikacije

Mikroservis je poslao poruku RabbitMQ, koja je otišla asihrono u message broker, te kreiran je drugi thread koji osluškuje navedeni proces.



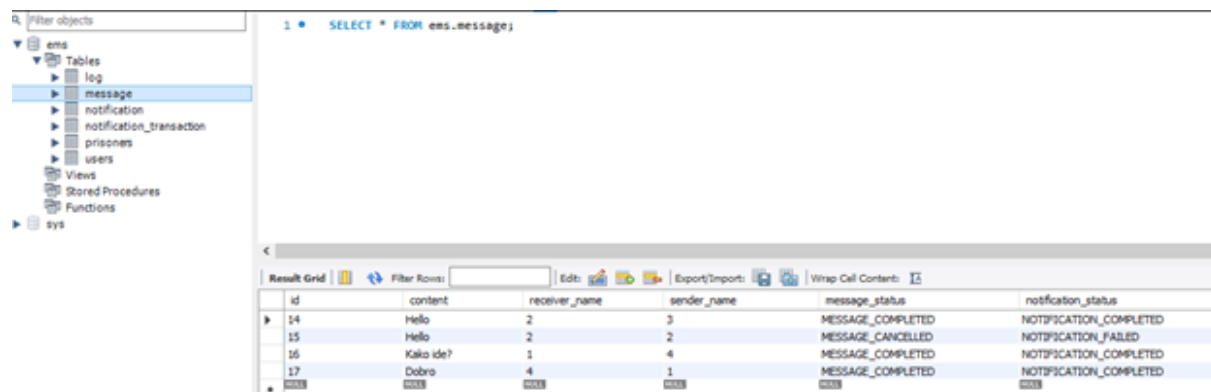
Slika 8.1: Uvezivanje komunikacije sa RabbitMQ



Slika 8.2: Prikaz uspješne komunikacije

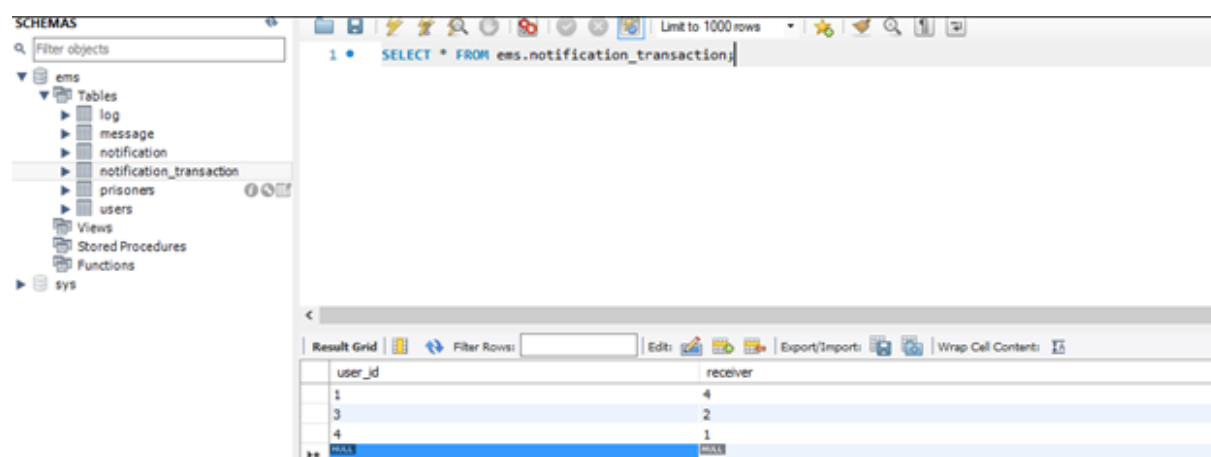
8.2 Implementacija saga pattern-a

Na sljedećim slikama prikazana je implementacija event-based pristup/saga choreography, obje zavise jedna od druge tako da ako jedna padne i druga poziva inverznu akciju i obrnuto.



id	content	receiver_name	sender_name	message_status	notification_status
14	Hello	2	3	MESSAGE_COMPLETED	NOTIFICATION_COMPLETED
15	Hello	2	2	MESSAGE_CANCELLED	NOTIFICATION_FAILED
16	Kako ide?	1	4	MESSAGE_COMPLETED	NOTIFICATION_COMPLETED
17	Dobro	4	1	MESSAGE_COMPLETED	NOTIFICATION_COMPLETED

Slika 8.3: Prikaz tabele message u bazi podataka (inverzna akcija)



user_id	receiver
1	4
3	2
4	1

Slika 8.4: Prikaz tabele notification transaction u bazi podataka (inverzna akcija)

9. Zadatak 9, 10 i 11

Navedeni zadaci su vezani za front-end, shodno tome, dokumentovanje zadataka će biti prikazano na video prezentaciji aplikacije, koja se nalazi na link-u: **video demonstracija aplikacije**.