Seminarski Rad

Fakultet: Fakultet Tehnickih Nauka u Novom Sadu
Smer: Softverske i informacione tehnologije

Tema: Dizajn i Implementacija Mikroservisa za Upravljanje Zdravstvenim Uslugama u Integrisanom Sistemu

Profesor:
Marko Markovic

Student:
Strahinja Sretenovic,

SR4/2022

Mesto, Godina
Novi Sad, 2025.

Sadržaj

1. Uvod

1.1. Problem digitalizacije u zdravstvu i obrazovanju

U modernom dobu, digitalizacija javnih usluga, kao što su zdravstvo i obrazovanje, predstavlja ključni korak ka efikasnijem i transparentnijem sistemu. Međutim, ovi sistemi su često razvijani izolovano, što dovodi do fragmentacije podataka i neefikasnih procesa koji zahtevaju manuelnu razmenu papirne dokumentacije. Primer takvog procesa je izdavanje lekarskih opravdanja za izostanke učenika, koje tradicionalno uključuje odlazak kod lekara, dobijanje papirne potvrde i njeno fizičko dostavljanje školi.

1.2. Cilj projekta

Cilj ovog projekta bio je razvoj integrisane platforme koja povezuje ključne aktere iz zdravstvenog i obrazovnog sistema (pacijente, lekare, učenike, nastavnike) kroz jedinstven digitalni ekosistem. Korišćenjem mikroservisne arhitekture, sistem je podeljen na logičke celine koje nezavisno funkcionišu, ali međusobno komuniciraju putem API-ja, omogućavajući automatizaciju složenih procesa.

1.3. Fokus rada

Ovaj seminarski rad se fokusira na detaljan opis dizajna i implementacije **Zdravstvenog Servisa (health_service)**. Rad će opisati njegove ključne funkcionalnosti, tehnička rešenja i način na koji health_service komunicira sa ostalim delovima sistema, pre svega sa Školskim servisom, ispunjavajući zahtev od minimalno tri razmene podataka.

2. Arhitektura Sistema

2.1. Mikroservisni pristup

Za potrebe projekta, izabrana je mikroservisna arhitektura zbog njene fleksibilnosti, skalabilnosti i mogućnosti nezavisnog razvoja i održavanja svakog dela sistema. Sistem je podeljen na tri glavna servisa:

- **SSO Servis:** Zadužen za jedinstvenu registraciju, prijavu (Single Sign-On) i upravljanje korisničkim ulogama.
- **Školski Servis:** Upravlja svim podacima i procesima vezanim za školu (ocene, izostanci, poruke).
- **Zdravstveni Servis:** Fokus ovog rada, zadužen za upravljanje terminima i lekarskim opravdanjima.

2.2. Tehnološki stek

- **Backend:** Python sa Flask radnim okvirom.
- **Baza Podataka:** MongoDB, sa GridFS modulom za skladištenje fajlova (PDF dokumenata).

•**Frontend:** Angular, sa angular-calendar bibliotekom za prikaz kalendara.

•**Kontejnerizacija:** Docker i Docker Compose.

3. Implementacija Zdravstvenog Servisa (health_service)

Zdravstveni servis (health_service) predstavlja centralnu komponentu za sve medicinske interakcije unutar sistema. Njegova arhitektura je dizajnirana da podrži složene tokove rada, od zakazivanja pregleda do izdavanja zvaničnih dokumenata. Implementacija je obuhvatila sledeće funkcionalnosti:

3.1. Upravljanje Terminima: Interaktivni Kalendarski Sistem

Da bi se proces zakazivanja pregleda modernizovao i učinio intuitivnim, implementiran je napredan sistem koji se oslanja na interaktivni kalendar. Ovaj sistem se sastoji od komplementarnih funkcionalnosti za lekara i pacijenta.

Lekaru je na njegovom panelu data mogućnost da efikasno upravlja svojim radnim vremenom. Odabirom datuma, početnog i krajnjeg sata radnog dana, sistem automatski izvršava POST zahtev na rutu /api/timeslots/create-day. U pozadini, servis generiše niz slobodnih termina (slotova) u predefinisanim intervalima od 30 minuta. Ovi termini se čuvaju u timeslots kolekciji u bazi podataka, svaki sa statusom "SLOBODAN".

Nakon kreiranja, termini se vizuelno prikazuju lekaru na kalendaru (angular-calendar biblioteka). Lekar u svakom trenutku ima jasan pregled svog rasporeda, gde su slobodni termini obojeni plavom, a rezervisani crvenom bojom, pružajući mu trenutnu informaciju o popunjenosti kapaciteta.

Pacijent na svom panelu prvo bira željenog lekara. Nakon toga, sistem poziva GET rutu /api/timeslots/doctor/<doctor_id> kako bi preuzeo sve termine za izabranog lekara. Klikom na željeni dan u kalendaru, frontend filtrira i izlistava samo dostupne termine. Jednim klikom na dugme "Rezerviši", pacijent pokreće PUT zahtev na rutu /api/timeslots/<slot_id>/book. U pozadini, servis atomično proverava da li je termin i dalje slobodan i, ukoliko jeste, menja njegov status u "REZERVISAN" i upisuje ID pacijenta u njega.

Svaki pacijent ima pristup listi svojih predstojećih i prošlih termina pozivom na GET rutu /api/timeslots/patient, što mu omogućava da lako prati svoje medicinske obaveze.

3.2. Proces Izdavanja Lekarskih Opravdanja

Ova funkcionalnost automatizuje kompletan tok izdavanja opravdanja, eliminišući potrebu za papirnom dokumentacijom i direktno povezujući zdravstveni i školski sistem.

Proces započinje u školskom sistemu. Kada učenik zatraži opravdanje, school_service šalje POST zahtev na internu API rutu /api/justifications/request. health_service prima ovaj zahtev i u svojoj bazi kreira novi JustificationRequest dokument, sa statusom "ZAPRIMLJEN", čime se formalno otvara slučaj.

Lekar na svom panelu vidi listu svih pristiglih zahteva za opravdanje. Za svaki zahtev ima opciju da ga odobri ili odbije jednim klikom, što pokreće PUT zahtev na rute /api/justifications/<id>/approve ili /api/justifications/<id>/reject.

Kada lekar odobri zahtev, health_service pokreće najsloženiji deo procesa:

1.**Komunikacija sa SSO Servisom:** Da bi se u PDF-u prikazala puna imena umesto ID-jeva, servis šalje GET zahteve na /api/users/<id> rutu sso_service-a kako bi dobio ime učenika i lekara.

2.**Generisanje PDF-a:** Koristeći fpdf2 Python biblioteku, dinamički se generiše PDF dokument. Podrška za Unicode karaktere (č,ć,š,đ,ž) je obezbeđena korišćenjem DejaVuSans fonta.

3.**Slanje PDF-a:** Odmah nakon generisanja, servis šalje PUT zahtev ka school_service-u. Ovaj zahtev je formatiran kao multipart/form-data i sadrži ažurirani status izostanka ("OPRAVDANO") i generisani **PDF fajl kao prilog**.

3.3. Prijem Zahteva za Konsultacije od Nastavnika

Kao treća ključna funkcionalnost koja omogućava direktnu saradnju između prosvete i zdravstva, health_service poseduje API rutu /api/consultations/request. Ova ruta mu omogućava da primi zahtev za konsultacije direktno od nastavnika, poslat preko školskog servisa. Ovi zahtevi se čuvaju u posebnoj kolekciji i prikazuju se lekaru na njegovom panelu, omogućavajući mu uvid u zabrinutost nastavnika povodom zdravstvenog stanja učenika.

4. Povezanost i Razmena Podataka sa Školskim Servisom

health_service nije izolovan, već je duboko integrisan sa school_service-om kroz **tri jedinstvene razmene podataka**:

4.1. Razmena 1: Prijem zahteva za opravdanje (Školski → Zdravstveni)

Kada učenik zatraži opravdanje, school_service inicira komunikaciju slanjem POST zahteva na /api/justifications/request rutu health_service-a. Ovim se formalno kreira "slučaj" u zdravstvenom sistemu.

4.2. Razmena 2: Slanje odobrenog opravdanja sa PDF prilogom (Zdravstveni → Školski)

Nakon što lekar obradi zahtev i sistem generiše PDF, health_service inicira povratnu komunikaciju. Šalje PUT zahtev na /api/absences/update-status rutu school_service-a. Ovaj zahtev sadrži ažurirani status izostanka i generisani **PDF fajl kao prilog**, koji school_service zatim čuva u svojoj bazi.

4.3. Razmena 3: Prijem zahteva za konsultacije (Školski → Zdravstveni)

Kada nastavnik ima potrebu da se konsultuje sa lekarom povodom nekog učenika, school_service inicira komunikaciju slanjem POST zahteva na /api/consultations/request rutu health_service-a. Ovaj zahtev sadrži ID-jeve nastavnika, učenika, lekara i tekst poruke.

4.4. Komunikacija sa Single Sign-On (SSO) servisom

Pored komunikacije sa školskim sistemom, health_service komunicira i sa sso_service-om radi dobijanja metapodataka o korisnicima, kao što su imena za popunjavanje PDF dokumenata, čime se izbegava redundancija podataka.

5. Kontejnerizacija pomoću Docker-a

5.1. Uloga Docker-a u projektu

Docker je korišćen da se svaki mikroservis i baza podataka zapakuju u sopstveni, izolovani kontejner. Ovo donosi ključne prednosti: konzistentnost okruženja za sve članove tima, jednostavnost pokretanja celog sistema jednom komandom i laku prenosivost na bilo koji server.

5.2. Konfiguracija docker-compose.yml

Centralni docker-compose.yml fajl orkestrira rad svih kontejnera. Definisana je zajednička virtuelna mreža (zss-network) koja omogućava kontejnerima da komuniciraju međusobno koristeći imena servisa kao adrese (npr. http://sso_service:5001), što je ključno za API komunikaciju.

6. Zaključak

6.1. Ostvareni rezultati

Kroz ovaj projekat uspešno je implementiran funkcionalan i robustan zdravstveni mikroservis. Realizovane su napredne funkcionalnosti poput interaktivnog kalendara za zakazivanje i automatskog generisanja i slanja PDF dokumenata. Najvažniji rezultat je uspešna, dvosmerna integracija sa školskim sistemom kroz tri jasno definisane razmene podataka, čime je u potpunosti digitalizovan i unapređen proces komunikacije između ova dva ključna javna sektora.

6.2. Mogućnosti za dalji razvoj

Postojeći sistem predstavlja odličnu osnovu za dalja unapređenja, kao što su: notifikacioni sistem za obaveštavanje korisnika o bitnim događajima, mogućnost video konsultacija, i integracija sa apotekama za slanje digitalnih recepata.

7. Literatura

1.Grinberg, M. (2018). Flask Web Development, 2nd Edition. O'Reilly Media.

2.Richardson, C. (2018). Microservices Patterns: With examples in Java. Manning Publications.

3.Freeman, A. (2020). Pro Angular 9, 4th Edition. Apress.

4.Hows, D., Membrey, P., & Plugge, E. (2014). MongoDB Basics. Apress.

5.Turnbull, J. (2018). The Docker Book: Containerization is the new virtualization. James Turnbull.