

Informacioni sistemi

II deo

Valentina Jošanović 16648

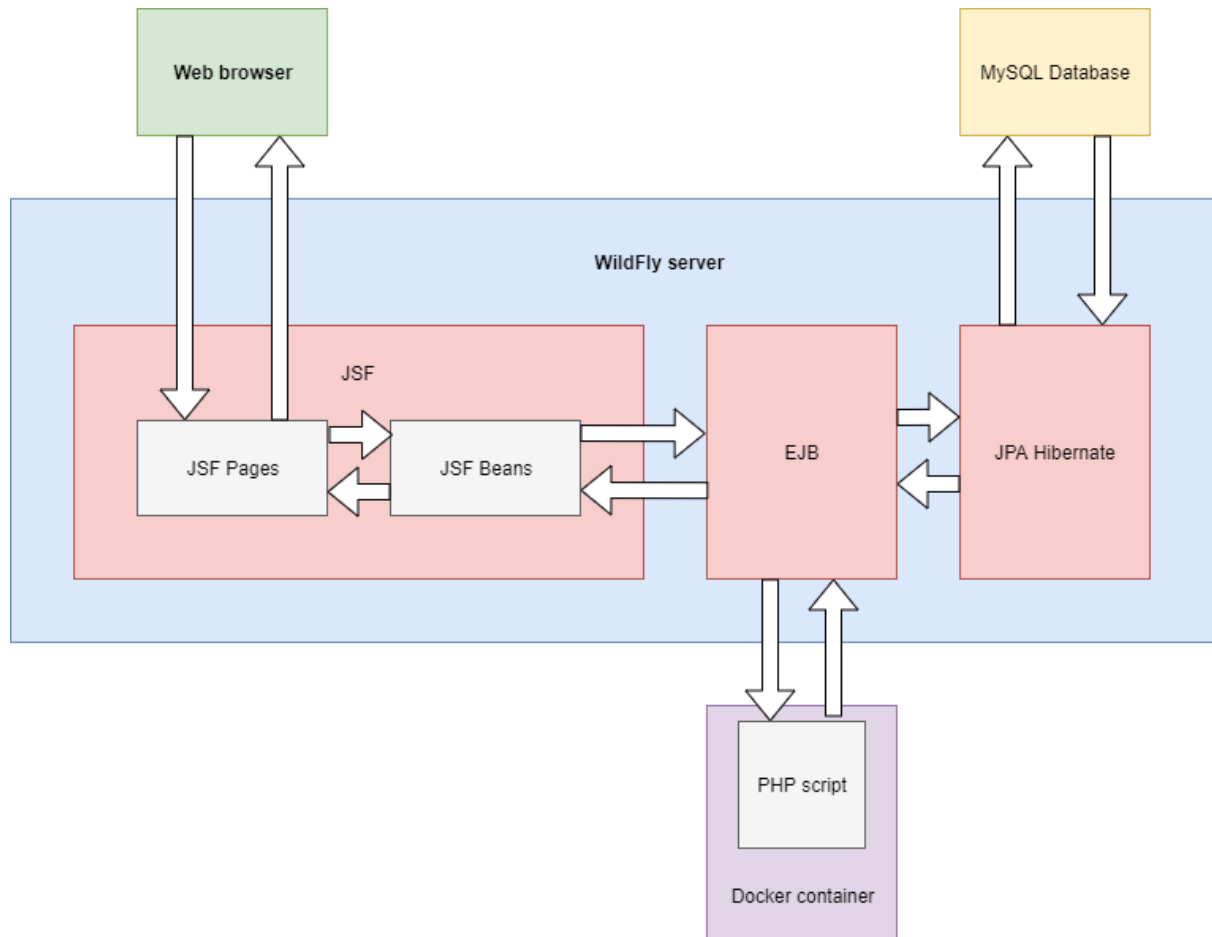
Email: valentina@elfak.rs

95. Informacioni sistem lanca prodavnica zdrave hrane

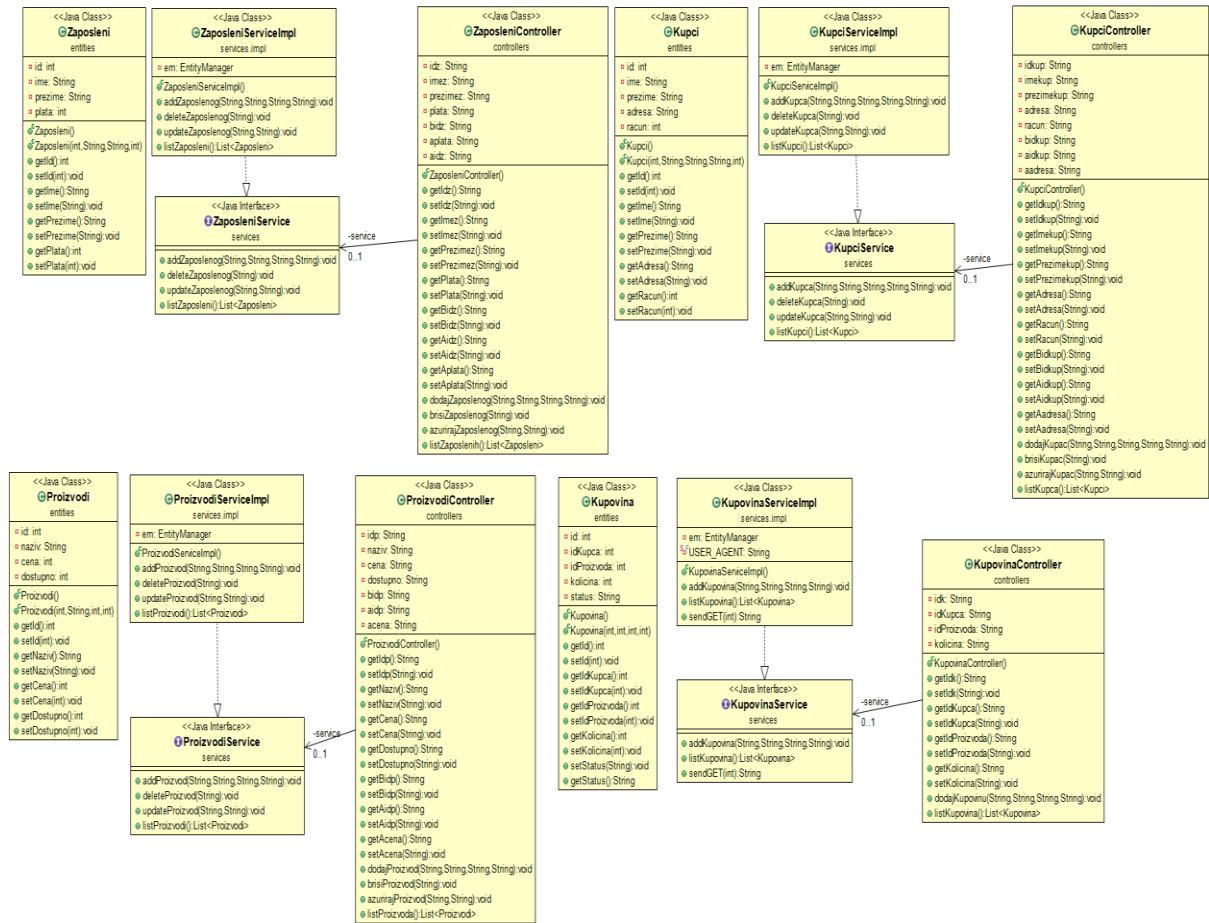
Lanac zdrave hrane pored prodajnih objekata poseduje magacine vezane za svaki od prodajnih objekata i centralni magacin iz koga se proizvodi dopremaju u magacine vezane za konkretno prodajno mesto. Predvideti elektronsku kupovinu, kupovinu licno u nekom od prodajnih objekata, komunikaciju izmedu magacina, evidentiranje kupaca, vodenje evidencije o zapošljenim radnicima, arhiviranje racuna. Svaka od prodavnica vrši isporuke kupcima na kucnu adresu. Organizacija poseduje sektor za upravljanje isporukama.

Obrazac: obrazac za zaposlenje

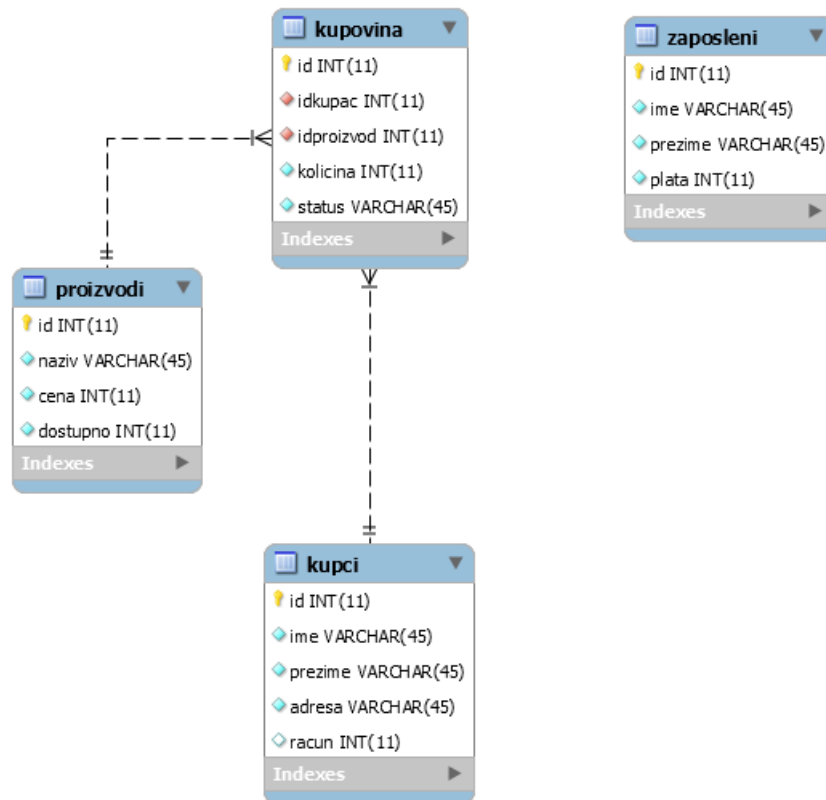
Arhitektura aplikacije



UML diagram

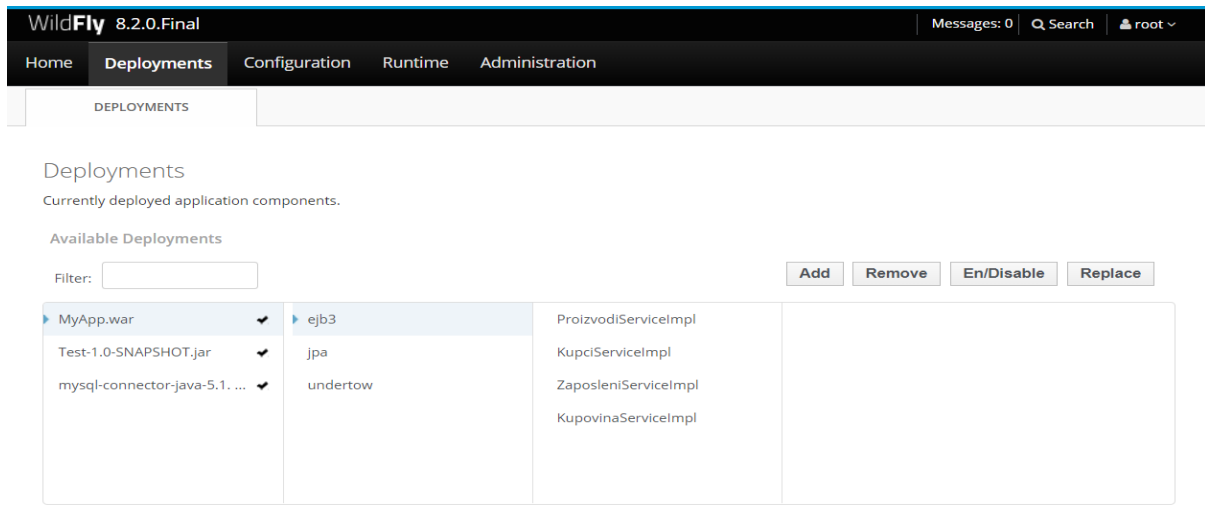


Šema baze podataka



Deployment aplikacije + docker

Startuje se WildFly server. Desnim klikom na projekat bira se opcija Run As > Run on Server. Zatim se pokreće JSF aplikacija koja se može videti na WildFly serveru na adresi „localhost:9991“ u delu Deployments.



Takođe treba pokrenuti i docker. Prvo treba da se pozicioniramo u terminalu gde se nalazi naša index.php skripta. Zatim se komandom `docker build -t my-app-lzh .` izgradi slika. Nakon toga se naredbom `docker run -d -p 80:80 --name lzh-app my-app-lzh` pokreće kontejnar.

```
MINGW64~/d/DockerIS
docker is configured to use the default machine with IP 192.168.99.100
For help getting started, check out the docs at https://docs.docker.com

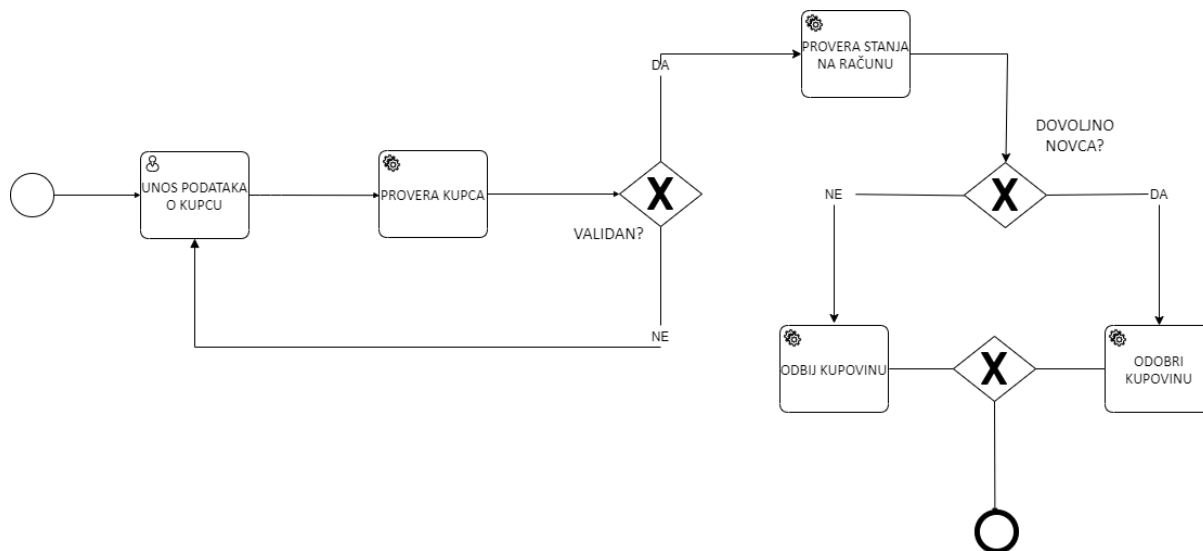
Start interactive shell
Vax@DESKTOP-T98PULQ MINGW64 /d/Docker Toolbox
$ cd "D:\DockerIS"

Vax@DESKTOP-T98PULQ MINGW64 /d/DockerIS
$ docker build -t my-app-lzh .
Sending build context to Docker daemon 3.072kB
Step 1/2 : FROM php:7.2-apache
--> 25a05bf23410
Step 2/2 : COPY . /var/www/html/
--> Using cache
--> 7f5c4c066193
Successfully built 7f5c4c066193
SECURITY WARNING: You are building a Docker image from Windows against a non-Windows Docker host. All files and directories added to build will have Windows permissions. Verify that the results are correct, and check and reset permissions for sensitive files and directories.

Vax@DESKTOP-T98PULQ MINGW64 /d/DockerIS
$ docker run -d -p 80:80 --name lzh-app my-app-lzh
a747641f99754104cec9bf22f3a670f1611c1888ad34e91510fe8cfe983932ce

Vax@DESKTOP-T98PULQ MINGW64 /d/DockerIS
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
a747641f9975        my-app-lzh         "docker-php-entryp..." 12 seconds ago     Up 13 seconds      0.0.0.0:80->80/tcp   lzh-app
```

BPMN dijagram



Docker servis proverava da li kupac ima dovoljno novca na računu.

Biznis logika na dijagramu iznad je predstavljena tako što se unesu podaci o kupcu pa se proverí validnost tih podataka. Nakon toga se proverava stanje na računu kupca. Shodno tome se odlučuje da li će kupovina biti odobrena ili odbijena.

Testiranje

1. testListZaposleni()

Naziv testa:

Funkcija koja vraća sve zaposlene.

Opis testa:

U bazi se nalaze svi zaposleni. Proverava se da li funkcija vraća sve zaposlene.

Preduslovi:

Treba da postoje neki zaposleni.

Koraci testa:

1. Pozvati funkciju koja vraća sve korisnike.
2. Izvršiti query upit
3. Proveriti da li je nešto vraćeno pomoću `assertNotNull`.

Test-podaci:

Podaci koji se nalaze u bazi.

Očekivani rezultati:

Funkcija ne vraća null.

Post-uslov:

Potrebno je da lista koju vraća funkcija ima neke vrednosti.

Status:

Uspešan.

2. testAddZaposlenog()

Naziv testa:

Funkcija koja dodaje novog zaposlenog.

Opis testa:

Poziva se funkcija i kao parametri joj se prosleđuju informacije o zaposlenom. Funkcija treba da napravi objekat tipa `Zaposleni` i da ga preslika u bazu podataka.

Preduslovi:

Treba da postoje klasa `Zaposleni`, baza i mapiranja na bazu.

Koraci testa:

1. Pozvati funkciju za dodavanje zaposlenog.
2. Napraviti objekat tipa Zaposleni.
3. Izvršiti query.

Test podaci:

Random izgenerisani podaci.

Očekivani rezultati:

Funkcija dodaje novog zaposlenog u bazu.

Post-uslov:

Da zaposleni bude kreiran.

Status:

Uspešan.

3. testUpdateZaposlenog()

Naziv testa:

Funkcija za ažuriranje postojećeg zaposlenog.

Opis testa:

Poziva se funkcija i kao parametri joj se prosleđuju id zaposlenog i nova vrednost plate. Funkcija će pronaći zaposlenog sa datim id-jem i promeniti mu platu, ako takav zaposleni postoji.

Preduslovi:

Zaposleni sa zadatim id-jem postoji.

Koraci testa:

1. Pozvati funkciju za ažuriranje zaposlenog.
2. Proveriti da li je ažuriranje izvršeno upoređivanjem sa ranije sačuvanom platom uz pomoć assertEquals.

Test-podaci:

Podaci iz baze podataka.

Očekivani rezultati:

Funkcija ažurira vrednost plate zaposlenom sa zadatim id-jem.

Post-uslov:

Da plata bude ažurirana.

Status:

Uspešan.

4. **testDeleteZaposlenog()**

Naziv testa:

Funkcija za brisanje postojećeg zaposlenog.

Opis testa:

Funkcija briše zaposlenog čiji id joj se prosleđuje kao parametar.

Preduslovi:

Zaposleni sa zadatim id-jem postoji.

Koraci testa:

1. Pozvati funkciju za brisanje zaposlenog i proslediti joj odgovarajući id.
2. Proveriti da li je zaposleni sa zadatim id-jem izbrisan iz baze.

Test-podaci:

Podaci iz baze podataka.

Očekivani rezultati:

Zaposleni je izbrisan iz baze.

Post-uslov:

Zaposleni sa prethodno zadatim id-jem mora da ne postoji više u bazi.

Status:

Uspešan.