

Семинарски рад

Апликација за продају карата у биоскопу

|  |  |
| --- | --- |
| Професор: | Студент: |
| Др Синиша Влајић | Милош Бркић 120/16 Д1 |

Садржај

[Опис 1](#_Toc50055014)

[Модел случајева коришћења 2](#_Toc50055015)

[Структура софтверског система – Концептуални модел 3](#_Toc50055016)

[Структура софтверског система – Релациони модел 4](#_Toc50055017)

[Пројектовање пословне логике 7](#_Toc50055018)

[Контролер апликационе логике 7](#_Toc50055019)

[Пословна логика 8](#_Toc50055020)

[Структура система 9](#_Toc50055021)

[Перзистенција података 10](#_Toc50055022)

[Пројектованје складишта података 12](#_Toc50055023)

[Архитектура софтверског система 15](#_Toc50055024)

[Имплементација 16](#_Toc50055025)

[Технологије 16](#_Toc50055026)

# Опис

Систем за продају карата треба да омогући запосленима биоскопа да воде евиденцију о продаји карата за различите пројекције. Систем ће обезбедити евиденцију о активним филмовима који се могу приказивати, филмским дистрибутерима, салама којима биоскоп располаже као и планираним пројекцијама.

Корисник система биоскипа биће **запослени и администратор**.

Да би се приступило систему запослени мора да се пријави. Након тога има могућности да уносни, брише и мења податке о филмовима, салама, дистрибутерима, пројекцијама као и продају карата купцима. Такође је потребна могућност за прертрагу наведених ентитета.

Филм и сала се могу избрисати из система под условом да се не користе ни у једној пројекцију. Да би се креирала нова пројекција потребно је да постоји бар један филм и сала. Брисање пројекције је могуће само уколико ниједна карта није продата за ту пројекцију. Да би се продавале карте мора псотојати бар једна пројекција. Такође систем треба да води рачуна о попуњеним седиштима за сваку пројекцију и према томе водити рачуна о продаји карата.

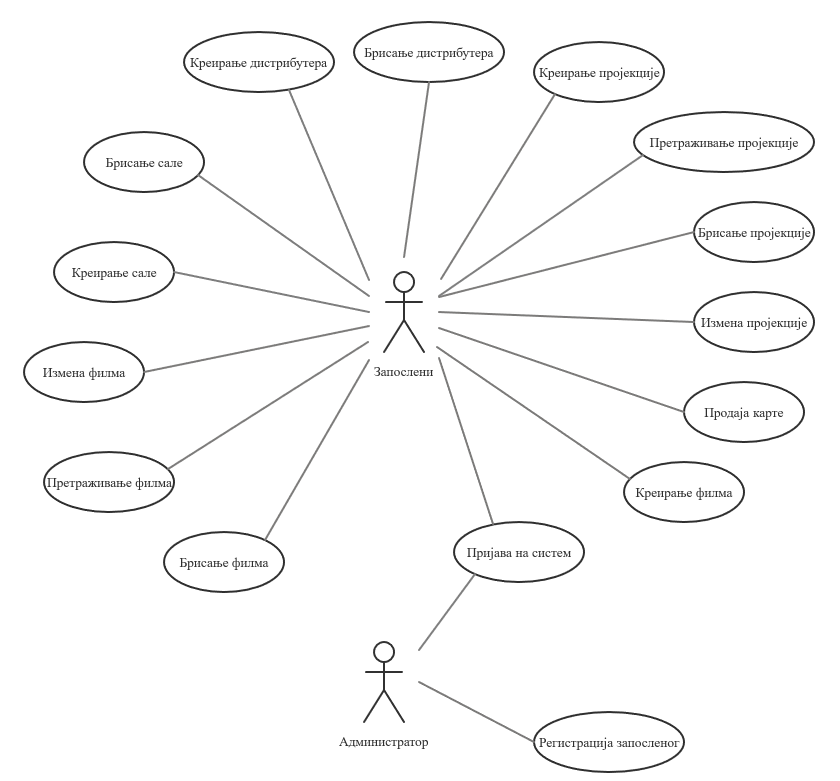
Процес продаје карата се одвија тако што купац долази на шалтер биоскопа и купује од запосленог карту за одређену пројекцију. Запослени уноси карту у систем а потом предаје купцу папирни примерак карте.

Администратор се пријављује на посебан налог и његова улога је да региструје нове запослене у систему.

Случајеви коришћења су следећи:

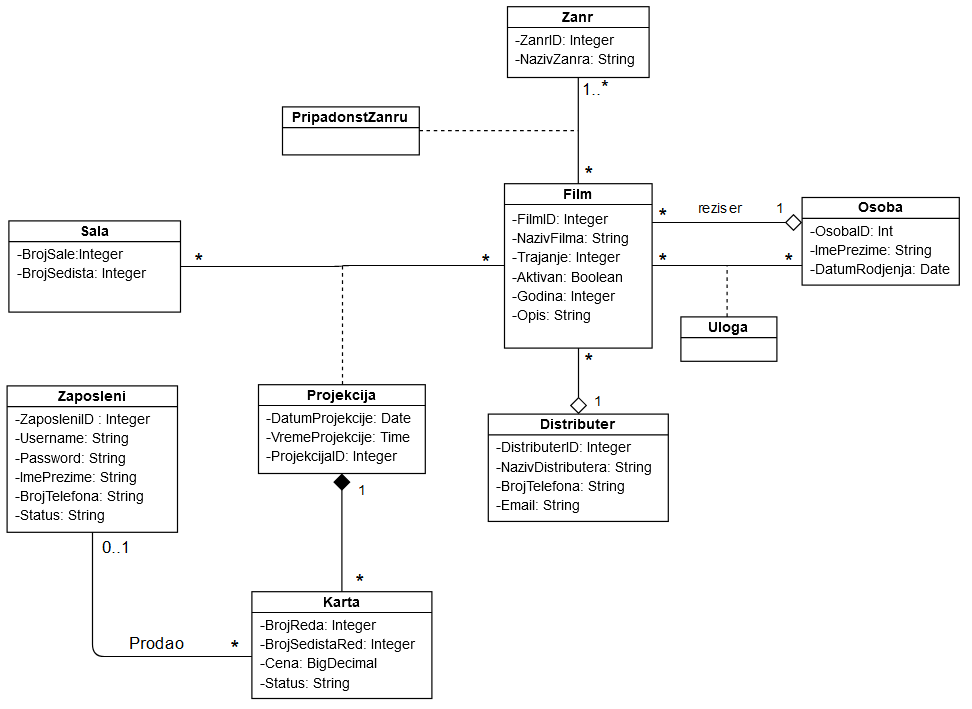
1. Пријава на систем
2. Регистрација запосленог
3. Креирање филма
4. Брисање филма
5. Претраживање филма
6. Промена података о филму
7. Креирање сале
8. Брисање сале
9. Креирање дистрибутера
10. Брисање дистрибутера
11. Креирање пројекције
12. Претраживање пројекције
13. Брисање пројекције
14. Промена пројекција
15. Продаја карте

# Модел случајева коришћења



UML модел случајева коришћења

# Структура софтверског система – Концептуални модел



Концептуални модел (дијаграм класа)

# Структура софтверског система – Релациони модел

Zaposleni(ZaposleniID, Username, Password, ImePrezime,BrojTelefona,Status);

Sala(BrojSale, BrojSedista);

Distributer(DistrubuterID, NazivDistributera, BrojTelefona, Email);

Zanr(ZanrID, NazivZanra);

Osoba(OsibaID, ImePrezime, DatumRodjenja);

PripadnostZanru(ZanrID, FilmID);

Uloga(OsobaID, FilmID);

Film(FilmID, NazivFilma, Trajanje, ZanrID, DistributerID, Aktivan, Godina, Opis, ReziserID);

Projekcija(ProjekcijaID, BrojSale, DatumProjekcije, VremeProjekcije, FilmID);

Karta(BrojReda, BrojSedistaRed, ProjekcijaID, Cena, ZaposleniID, Status);

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Zaposleni** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT / UPDATE CASCADES Karta DELETE RESTRICTED Karta |
| ZaposleniID | Integer | not null |
| Username | String | not null |
| Password | String | not null |
| ImePrezime | String | not null |
| BrojTelefona | String |  |
| Status | String |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Sala** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT / UPDATE CASCADES Projekcija  DELETE RESTRICTED Projekcija |
| BrojSale | Integer | not null |
| BrojSedista | Integer | not null and > 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Distributer** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT / UPDATE CASCADE Film DELETE RESTRICTED Film |
| DistributerID | Integer | not null |
| NazivDistributera | String | not null |
| BrojTelefona | String |  |
| Email | String |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Zanr** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT / UPDATE CASCADES PripadnostZanru DELETE RESTRICTED PripadnostZanru |
| ZanrID | Integer | not null |
| NazivZanra | String | not null |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Osoba** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT / UPDATE CASCADES Uloga DELETE RESTRICTED Uloga |
| OsobaID | Integer | not null |
| ImePrezime | String | not null |
|  | DatumRodjenja | Date | not null |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Uloga** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT RESTRICTED Osoba, Film UPDATE RESTRICTED Osoba,Film DELETE / |
| OsobaID | Integer | not null |
| FilmID | Integer | not null |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела PripadnostZanru** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT RESTRICTED Zanr, Film UPDATE RESTRICTED Zanr,Film DELETE / |
| ZanrID | Integer | not null |
| FilmID | Integer | not null |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Film** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT RESTRICTED PripadnostZanru,Distributer  UPDATE CASCADE Projekcija RESTRICTED PripadnostZanru, Distributer DELETE CASCADE PripadnostZanru RESTRICTED Projekcija |
| FilmID | Integer | not null |
| NazivFilma | String | not null |
| Trajanje | Integer | not null and > 0 |
| DistributerID | Integer | Not null |
| Aktivan | Boolean |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Projekcija** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT RESTRICTED Film, Sala  UPDATE RESTRICTED Film, Sala  DELETE RESTRICTED Karta |
| ProjekcijaID | Integer | not null |
| BrojSale | Integer | not null |
| DatumProjekcije | Date | not null |
| VremeProjekcije | Time | not null |
| FilmID | Integer | not null |

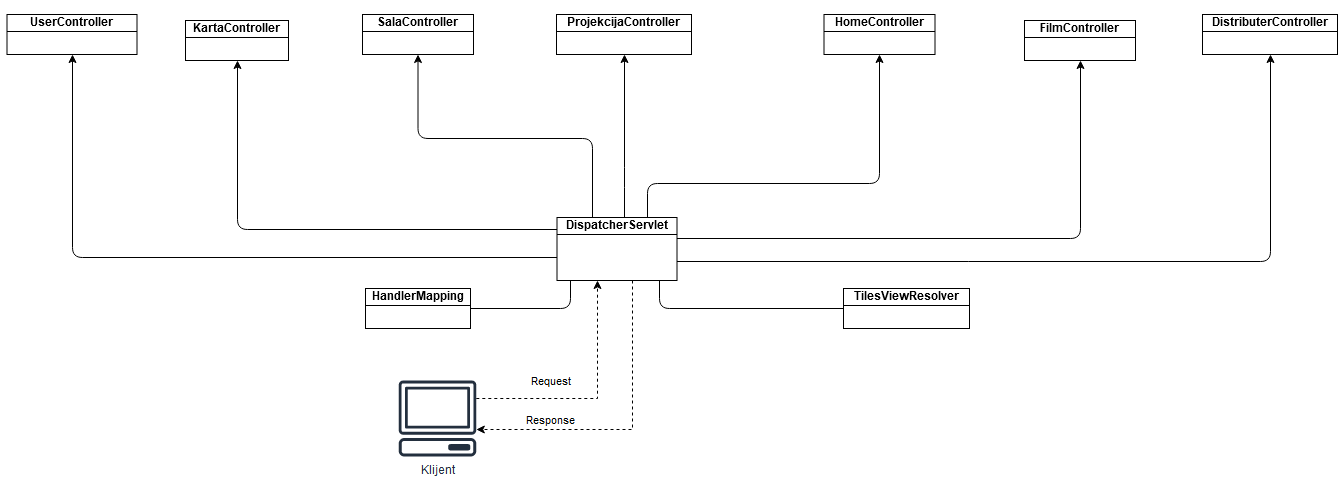
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Табела Karta** | | **Просто вредносно ограничење** | | **Структурно ограничење** |
| **Атрибути** | **Име** | **Тип атрибута** | **Вредност атрибута** | INSERT RESTRICTED Projekcija, Zaposleni UPDATE RESTRICTED Projekcija, Zaposleni DELETE / |
| BrojReda | Integer | not null |
| BrojSedistaRed | Integer | not null |
| ProjekcijaID | Integer | not null |
| Cena | BigDecimal | > 0 |
|  | ZaposleniID | Integer |  |  |
|  | Status | String |  |  |

# Пројектовање пословне логике

## Контролер апликационе логике

Класа која је задужена за примање захтева и слање одговора од стране клијента је DispecherServlet. Осим диспечера, постоје контролери који су задужени за обраду захтева везани за доменске објекте.

Након што диспечер прими захтев, проналази који контролер треба да га обради и њему прослеђује захтев. Контролери користе класе за валидацију и сервисе како би обрадиле захтеве након чега одговор прослеђују диспечеру. Диспечер клијенту враћа добијени одговор као и одговарајућу страницу.



## Пословна логика

Понашање софтверског система реализује се помоћу сервиса. Сервиси су веза између контролера и репоситорија. Сваки доменски објекат има одговарајући сервис који се бави његовом обрадом. Улога сервиса је такође да проследи доменски објекат одговарајућем репоситоријуму како би се извршила њихова перзистенција.

Сервиси доменских објекатра су изведени из *AbstractService*-а.

*@Service*

*public abstract class AbstractService<T,R> {*

*@Autowired*

*protected R repository;*

*public AbstractService(R repository) {*

*this.repository = repository;*

*}*

*public abstract T findById(int id);*

*public abstract List<T> getAll();*

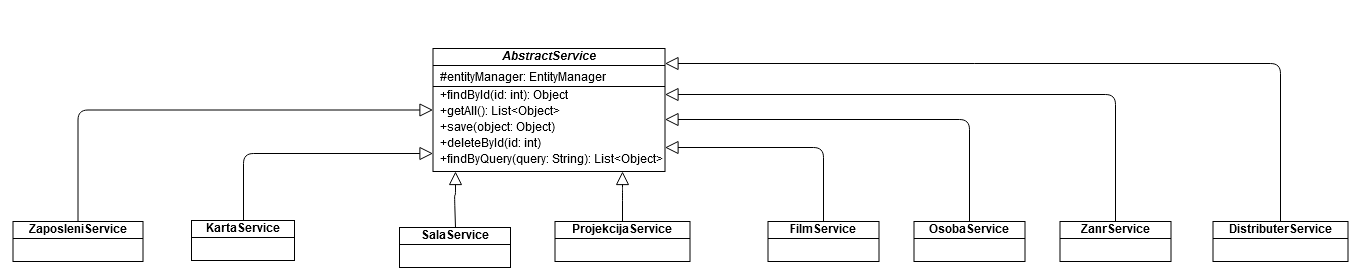
*public abstract void save(T object);*

*public abstract void deleteById(int id) throws Exception;*

*public abstract List<T> findByQuery(String query);*

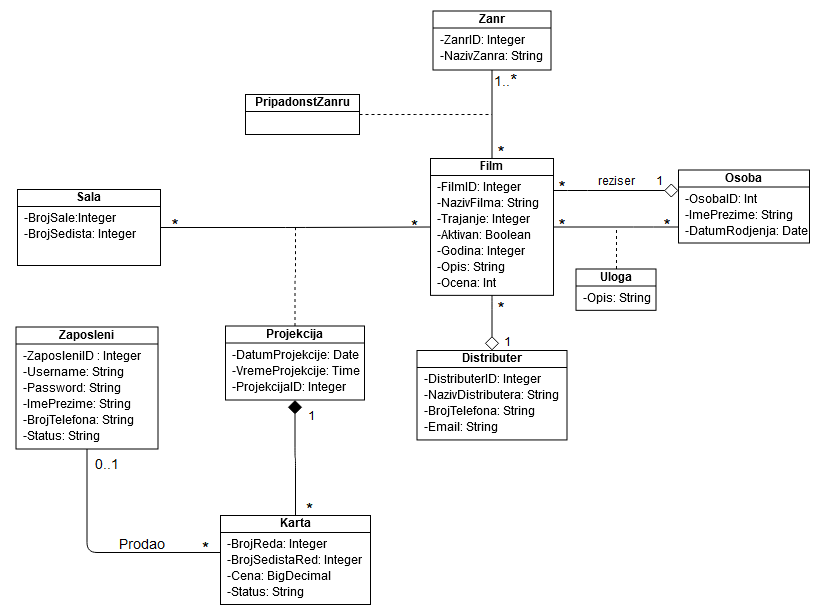
*}*

Параметар Т представља тип доменског објекта док параметар R одговарајући репозитори за тај објекат.



## Структура система

Структура система се састоји од скупа доменских класа. Доменске класе су пројектоване на основу концептуалног модела. Доменским класама су придружене одговарајуће анотације како би се омогућила њихова перзистенција преко JPA и Hibernate-a.



# Перзистенција података

За перзистенцију доменских објеката користе се *Repository* класе. Сваки доменски објекат има одговарајућу *Repository* класу која је задужна за имплелемтацију CRUD операција над тим објектом.

Све *Repository* класе су изведене из *AbstractRepository*-ja који даје спецификацију метода за извршавање CRUD операција. Осим абстрактних метода, *AbstractRepository* такође поседује објекат *entityManager* који је задужен за комуникацију са базом података и извршавање упита над њом.

*@Repository*

*@Transactional(propagation = Propagation.REQUIRED)*

*public abstract class AbstractRepository<T> {*

*@PersistenceContext(type = PersistenceContextType.TRANSACTION)*

*protected EntityManager entityManager;*

*@Transactional*

*public abstract T findById(int id);*

*@Transactional*

*public abstract List<T> getAll();*

*@Transactional*

*public abstract void save(T object);*

*@Transactional*

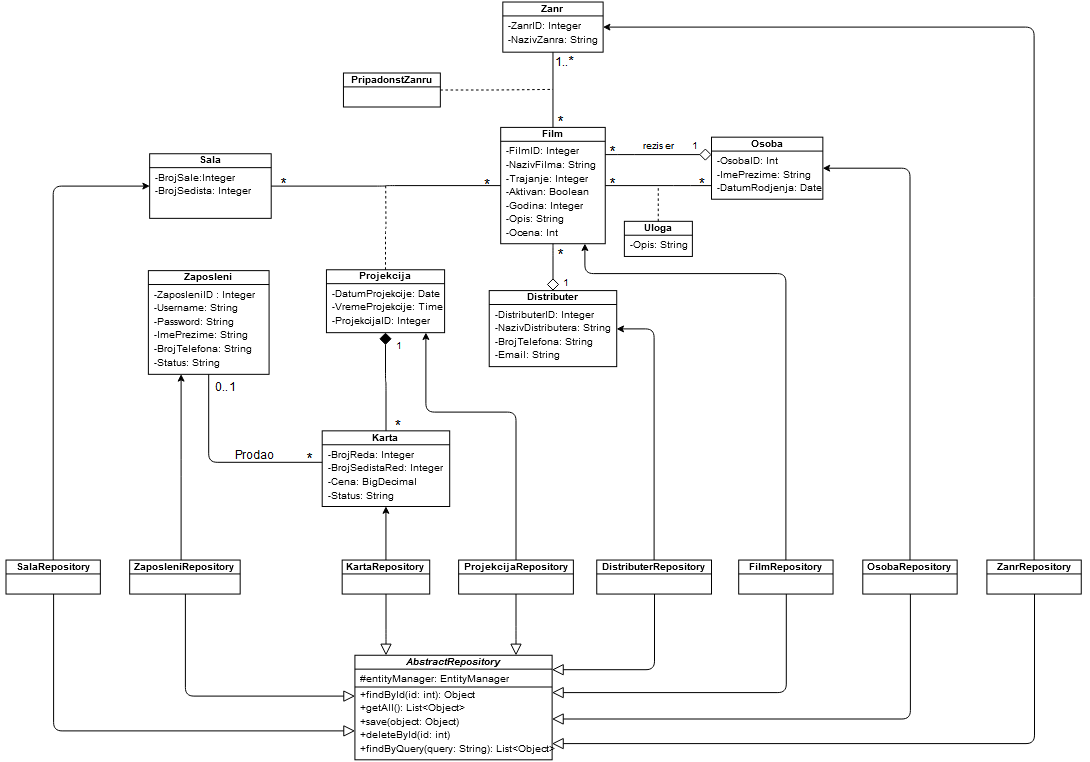
*public abstract void deleteById(int id) throws Exception;*

*@Transactional*

*public abstract List<T> findByQuery(String query);*

*}*

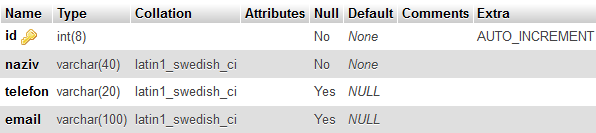
Параметар Т представља тип доменског објекта.



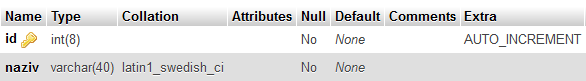
# Пројектованје складишта података

На основу доменских класа пројектују се релације базе података.

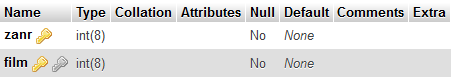
**Table: distributer**



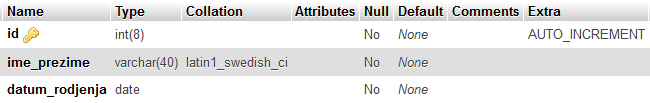
**Table: zanrovi**



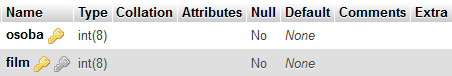
**Table: pridapnost\_zanru**



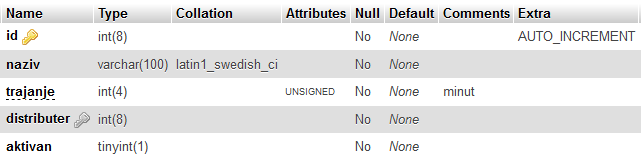
**Table: osoba**



**Table: uloga**



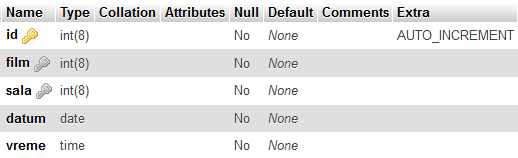
**Table: filmovi**



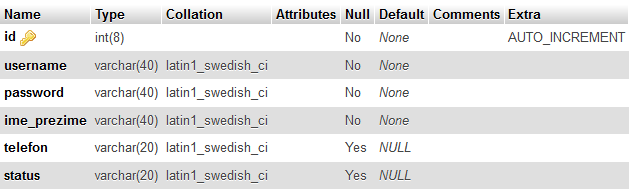
**Table: sale**



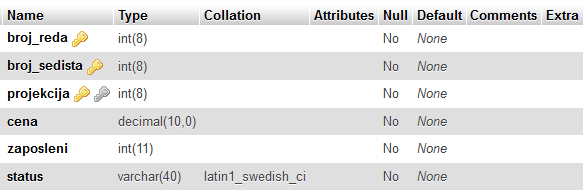
**Table: projekcije**



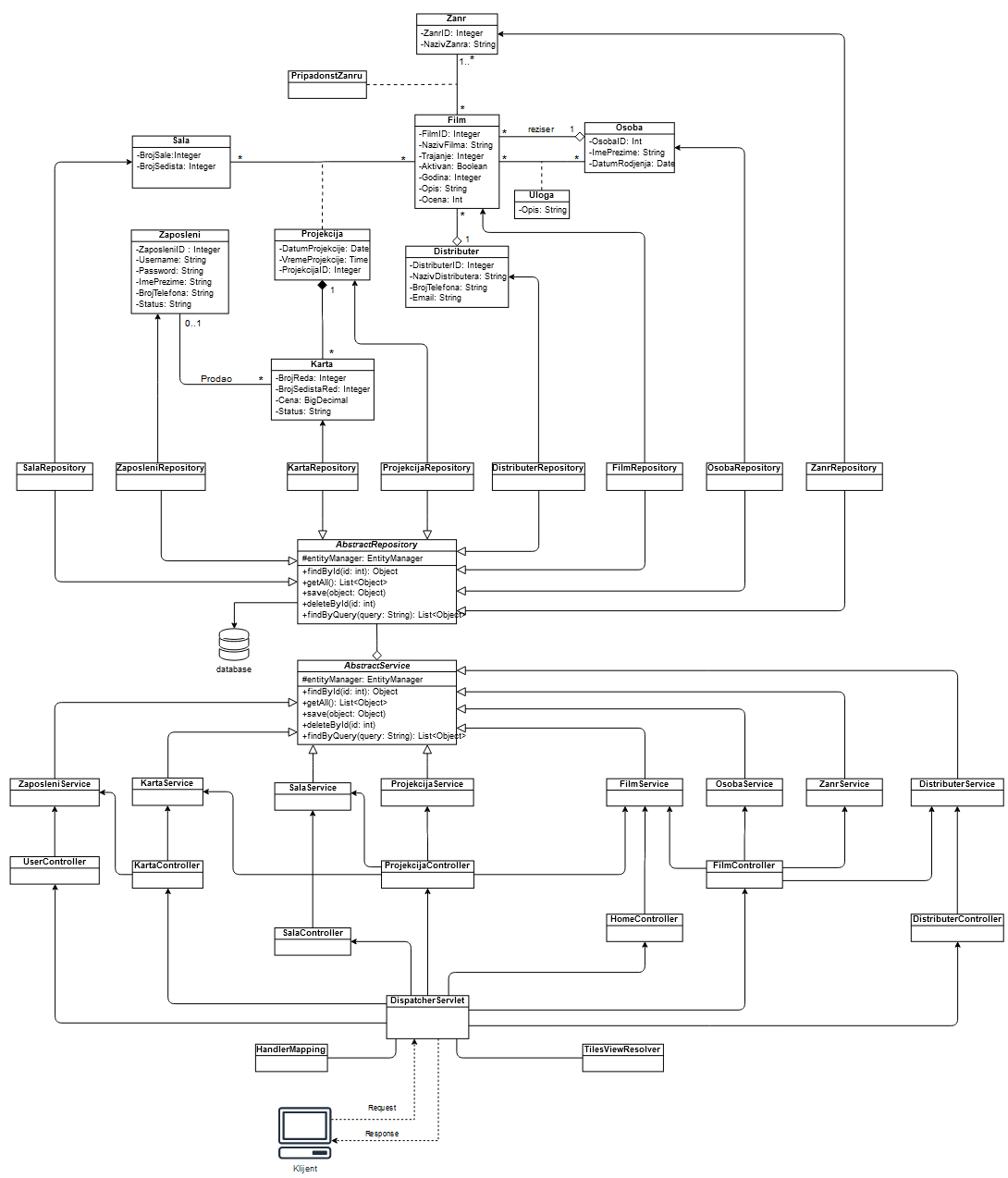
**Table: zaposleni**



**Table: karte**



# Архитектура софтверског система



# Имплементација

Софтверски систем је развијен у програмском језику јава. Окружење које је коришћено за развој је NetBeans IDE. Систем за управљање базом података је MySQL.

## Технологије

Веб апликација имплементирана је у јава програмском језику уз коришћење следећих технологија:

**Spring** – фрејмворк на коме се заснива апликација. Главна одлика спринг фрејмворка је спрингов контејнер. Контејнер је заслужан за креирање објеката, њихово повезивање, подешавање као и управљање њиховим животним циклусом. Објeкти којима контејнер управља називају се *Beam*-ови.

|  |  |
| --- | --- |
| 1-PIFheZ8h8OO_hss9PETXIw.png | spring_ioc_container.jpg |

За подешавање контејнера могу се користити XML фајлови, анотације и јава код.

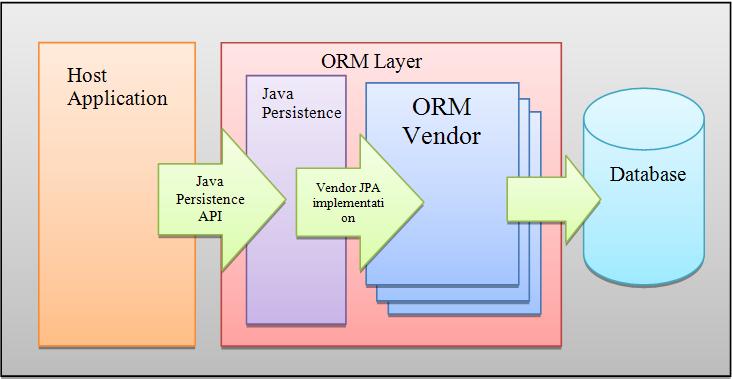
**Spring MVC** – део спринг фрејмворка који обезбеђује *Model-View-Controller* архитектуру. Овај фрејмворк обезбеђује *Dispecher servlet*  чија улога је да прихвата клијентске захтеве. Након тога, диспечер се обрађа *Handler Mapping*- у и, на основу на URL захтева, проналази одговарајући контролер који треба да обради захтев. Након што контролер обради захтев, он диспечеру враћа објекат *ModelAndView* који садржи одговарајуће објекте који требају бити враћени клијенту као и назив View-a који одређује која страница треба да се прикаже. Након тога, на основу назива View-a, диспечер тражи од ViewResolver-a да му врати одговарајући View. На крају клијенту се као одговор враћа View са одговарајућим моделом.



**Spring security** – део спринг фрејмворка који се користи за аутентификацију и ауторизацију корисника. Сам врејмворк обезбеђује проверу параметара за аутентфикацију, управљање сесијом пријавњених корисника, управљање привилегијама корисника за приступ одређеним ресурсима, енкрипцију лозинке итд.

**Hibernate ORM** – фрејмворк који мапира објектни модел (јава класе) у релациони модел (табеле базе података) како би се једноставније спроводила перзистенција доменских објеката.

**Java persistence api (JPA)** - спецификација коју имплементирају објектно релациони мапери (ОRМ). JPA обезбеђује само стандардизовану спецификацију за сваки ОRМ али као таква се не може самостално користити већ се мора обезбедити и имплементација те спецификације коју, у овој апликацији, обезбеђује Hibernate ORM.



**Java server pages (JSP)** – технологија која се користи за креирање динамичких веб страница. У HTML странице се убацују JSP тагови како би се страница повезала са backend-ом. Oсим основних тагова, додате су и следеће библиотеке JSP тагова:

* JSP Standard Tag Library (JSTL) – додатни ЈSP тагови за итерације, if условe итд.
* FMT – тагови за приказивање локализованих порука
* Spring form – спринг тагови који обезбеђују мапирање атрибута добијених из HTML елемената у доменски објекат

**Apache Tiles** – фрејмворк који се користи за креирање фрагмената страница као и layout-а у који се могу фрагменти убацивати. На овај начин се обезбеђује да се одређени делови страница који се често појављују, као што су *header* и *footer*, не морају писати на свакој HTML страници. Уместо тога, страници је додељен layout у који су додати header и footer у облику фрагменaта.

**Bootsrtap** – CSS фрејмворк за једноставније и брже дизајнирање веб страница.