**Techninė ataskaita**

Komanda „Pyragas“

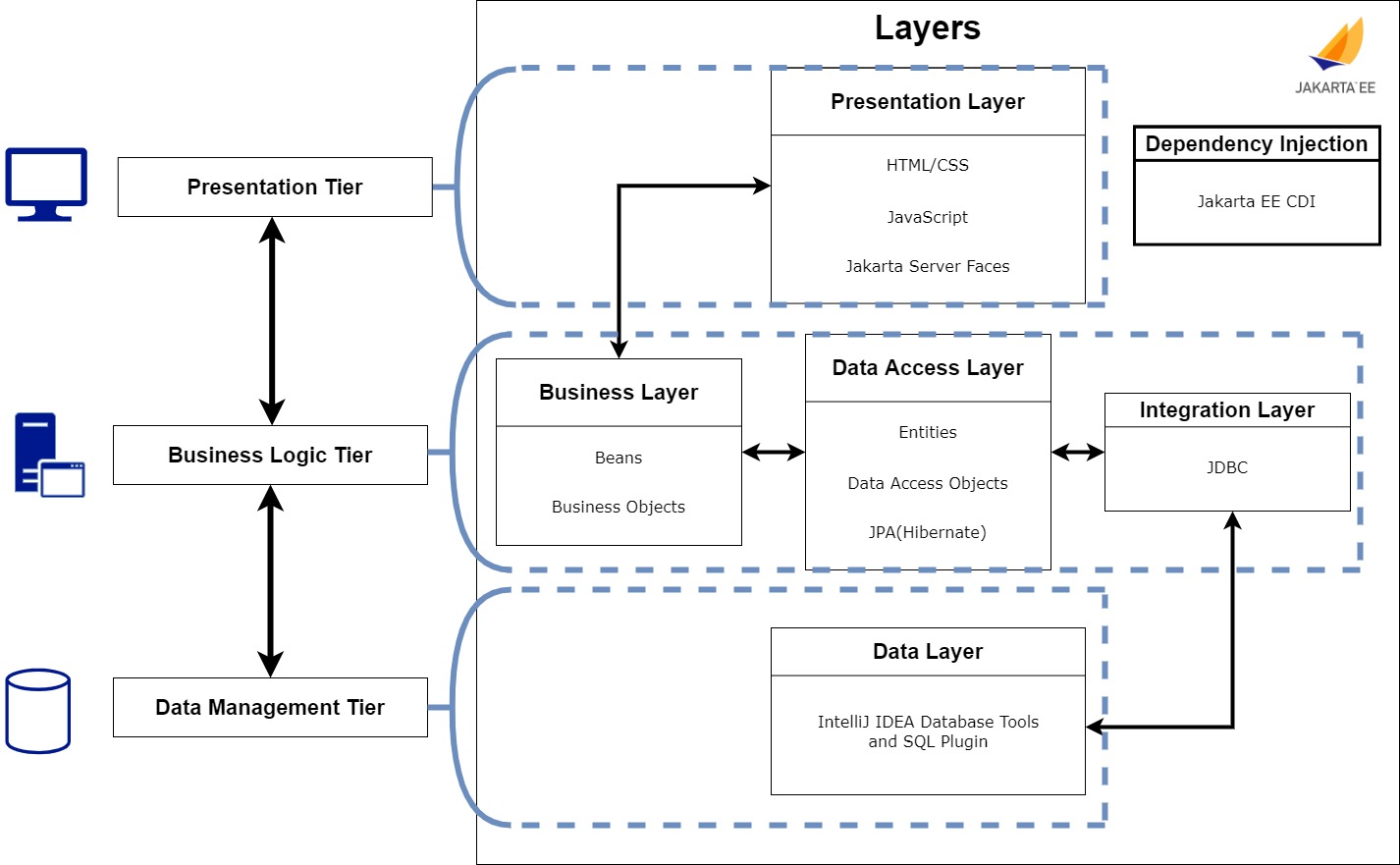
Gintautas Jurgelevičius Marijus Jasinskas Raminta Urbonavičiūtė Mantas Kryževičius Vladislav Kolupayev

# Projekto informacija

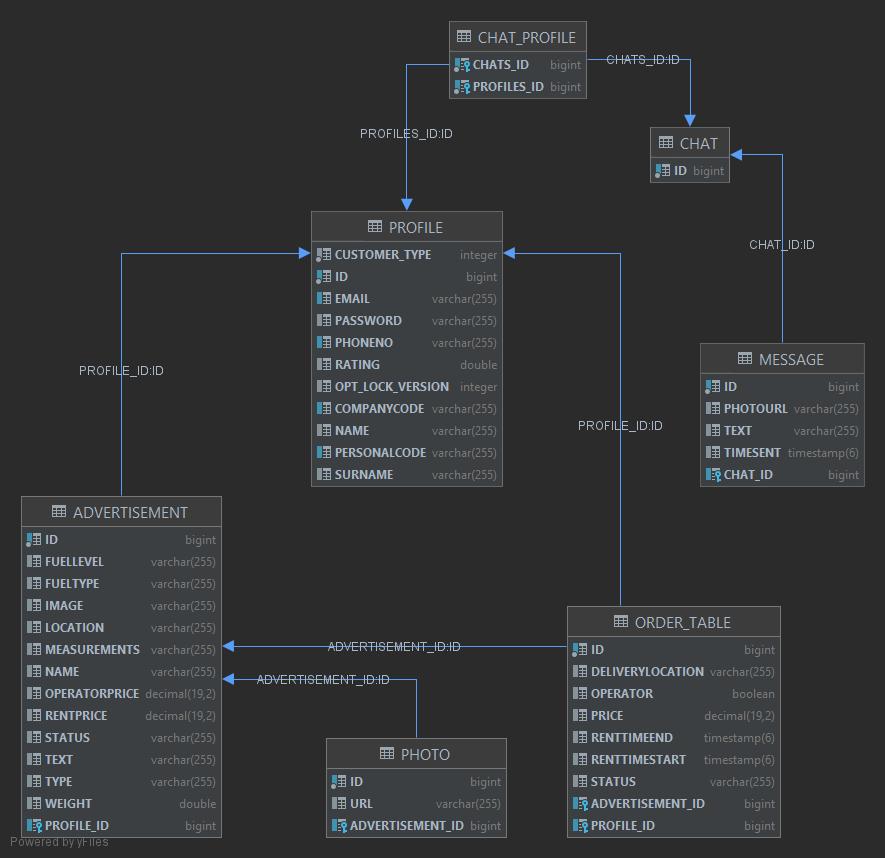
Užduotis: statybinės įrangos dalijimosi platforma

[Projekto repozitorijos nuoroda](https://github.com/Pyragas/tool-rent/tree/master)

# Sistemos architektūra



# Sistemos duomenų bazės schema



# Kokybiniai sistemos reikalavimai

## Concurrency

Vartotojo prisijungimas išlieka keliuose languose. Prisijungimo įsiminimui naudojame ExternalContext klasės getSessionMap() metodą. Gautame Map laikome vartotojo prisijungimo būseną ([src/main/java/com/psk/pyragas/ToolRent/usecases/Registration.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/usecases/Registration.java):47 eilutė):



Kai vartotojas atsijungia, atitinkamai pakeičiame ir ExternalContext esančią prisijungimo būseną [(src/main/java/com/psk/pyragas/ToolRent/usecases/Logout.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/usecases/Logout.java):8 eilutė):



## Security

Klasėje ProfilesDAO naudojame query parametrus, kad sukurtume užklausą, kuri yra atspari SQL atakoms ([src/main/java/com/psk/pyragas/ToolRent/dao/ProfilesDAO.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/dao/ProfilesDAO.java): 29 eilutė):



Užklausos sakinys su užvardintais parametrais ([src/main/java/com/psk/pyragas/-ToolRent/entities/Profile.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/entities/Profile.java): 14 eilutė):



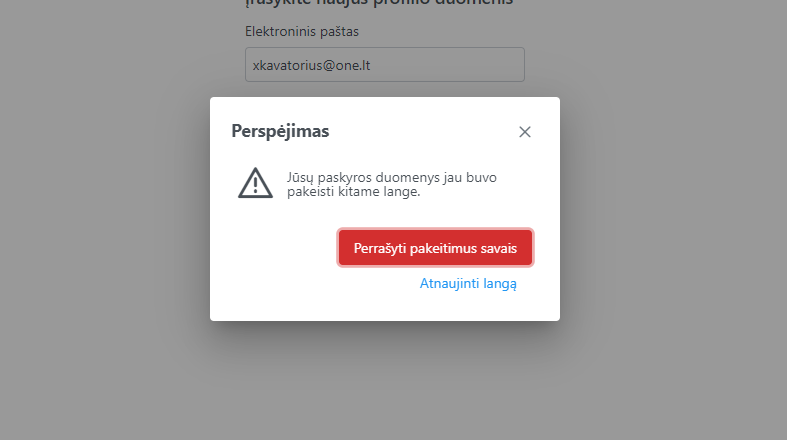
## Data Access

Nesu įsitikinęs, ką čia tiksliai parašyti. Tarsi pas mus yra @Model klasės, o tos, kurios yra @ViewScoped turi Transactional metodus. Tad tokią informaciją ir pateikti? Kad dėl šito yra įgyvendintas reikalavimas?

## Data consistency and Optimistic locking

Klasėje EditProfile implementavome metodą updateProfile(), kuris leidžia tam pačiam arba keliems vartotojoms redaguojant tos pačios paskyros duomenis pasirinkti, ar perrašyti kito vartotojo pakeitimus, ar nieko nedaryti ir atnaujinti langą ([src/main/java/com/psk/pyragas-/ToolRent/usecases/EditProfile.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/usecases/EditProfile.java): ?? eilutė):





## Memory management

SessionScoped komponentų nenaudojame nenaudojame use-case įgyvendinimui. Verslo klasės projekte yra arba @Model (RequestScoped) arba @ViewScoped. Pavyzdžiui, klasė CreateAdvertisement yra @Model ([src/main/java/com/psk/pyragas/ToolRent/usecases/-CreateAdvertisement.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/usecases/CreateAdvertisement.java)):



Klasėje EditProfile teko naudoti @ViewScoped. EditProfile gyvavimo metu nenaudoja daug resursų ([src/main/java/com/psk/pyragas/ToolRent/usecases/EditProfile.java](https://github.com/Pyragas/tool-rent/blob/master/src/main/java/com/psk/pyragas/ToolRent/usecases/EditProfile.java)):



## Reactive programming and Asynchronous/non-blocking communication

Šitą dar reikėtų įgyvendinti.

## Cross-cutting functionality and Interceptors

Dalykinio funkcionalumo veiksmus žurnalizuojame klasėje MethodLogger. Tai klasė su @Interceptor anotacija ir metodu logMethodInfo(). Šiame metode į žurnalizavimo failą įrašoma tam tikra informacija prieš ir po metodo įvykdymo. Žurnalizuojame, kada metodas buvo pradėtas vykdyti, kokia klasė kvietė metodą ir koks buvo metodo rezultatas po vykdymo (src/main/java/com/psk/pyragas/ToolRent/interceptors/MethodLogger.java: Įdėti nuorodą ir eilutės numerį):



## Extensibility/Glass-box extensibility

Naudojame dekoratorių SummerSaleAdvertisementsDAO, kuris implementuoja IAdvertisementsDAO interfeisą ir užtikrina, kad visi skelbimai turėtų žemesnę nuomos kainą vasarinio išpardavimo metu (src/main/java/com/psk/pyragas/ToolRent/decorators/Summer-SaleAdvertisementsDAO.java: Įdėti nuorodą):

