# **Checkpoint 2**

# Meeting Room Reservation System

Verze dokumentu: 2.3.0

**Datum odevzdání:** 15.01.2023

**Název projektu:** Meeting Room Reservation System

**Odkaz na stránky projektu:** [B221\_B6B36EAR/rastvdmy](https://gitlab.fel.cvut.cz/B221_B6B36EAR/rastvdmy)

**Řešitele:** Dmytro Rastvorov

**Termín cvičení:** ST – 18:15

**Jméno cvičícího:** Miroslav Holeček

**Obsah**

* Téma
* Krátký popis očekávané funkcionality
* Druhy uživatelů
* Funkce pro jednotlivé typy uživatelů
* Systémová omezení
* Objektový model (UML class diagram)
* Databáze
* API
* Popis aplikace a její struktury
* Poznámky k použití projektu
* Docker Compose
* Instalace aplikaci
* Odkazy
* Zpětná vazba
* **Téma:**

Meeting Room Reservation System

* Systém, který podporová správu jednacích místností a jejich rezervaci.
* **Krátký popis očekávané funkcionality:**

1. **Seznam hlavních funkcí aplikace:**
   * Systém správy se skládá z uživatelské a administrativní části:

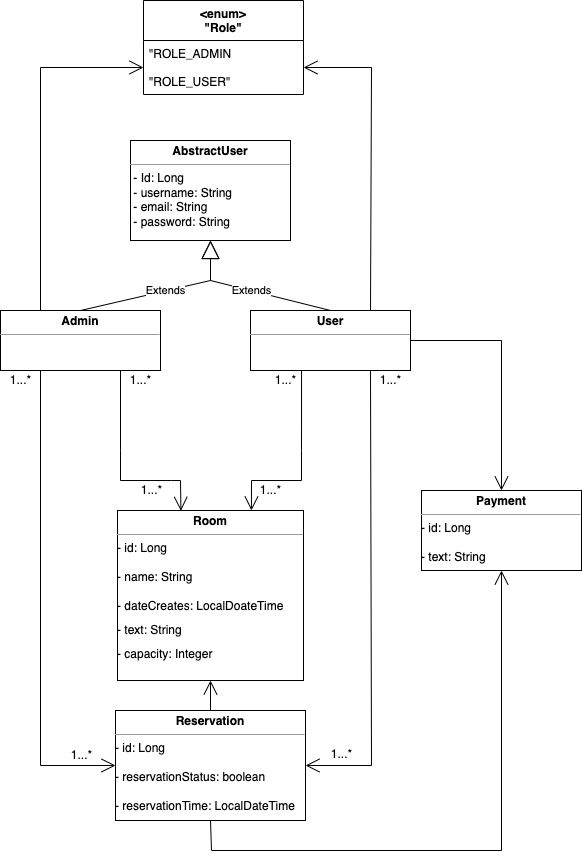
* + - **Uživatel**
    - **Administrator**

1. **Komu bude výsledný systém určen:**

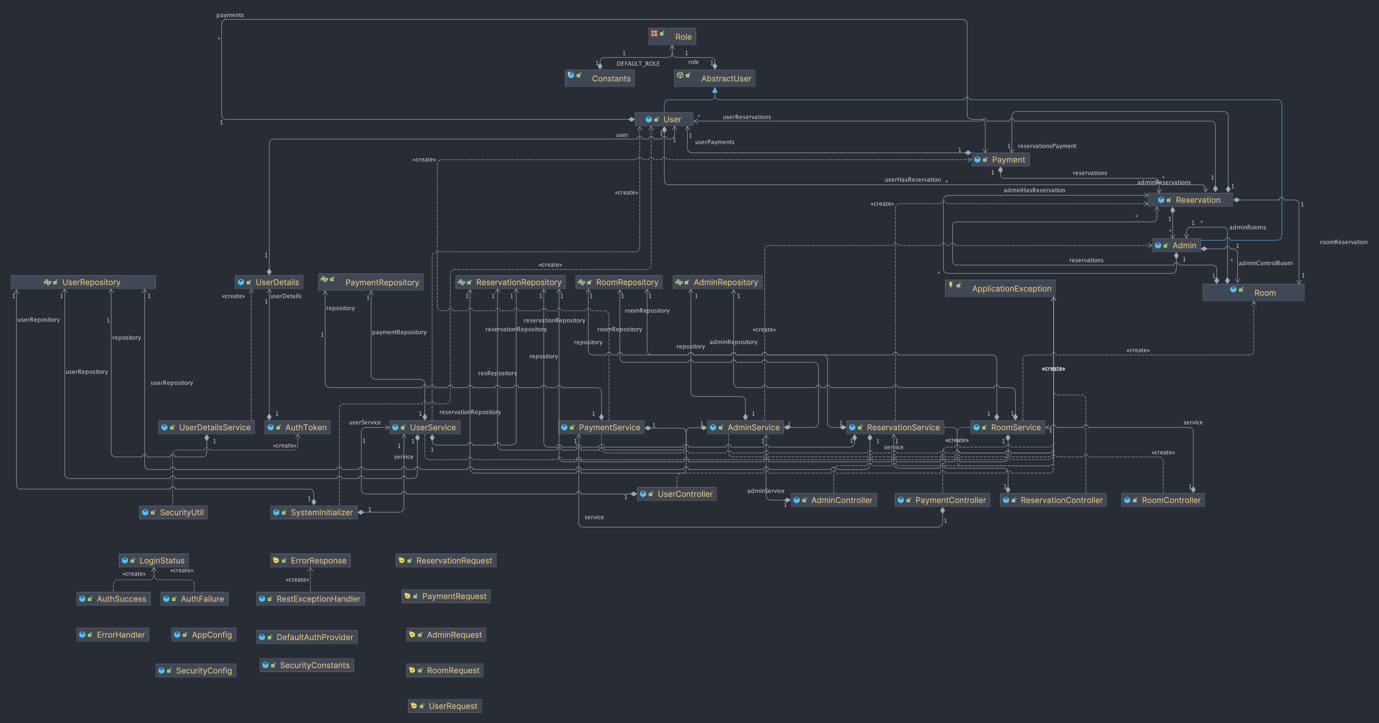
* Tento systém je určený pro skupinu osob, které plánují schůzky v konferenčních místnostech.
* **Druhy uživatelů:**
* Systém je určen pro 2 typy uživatelů: **Uživatel** a **Administrátor**.
  + **Uživatel – je** uživatel, který komunikuje s aplikací pro rezervaci konferenčních místností.
  + **Administrator – je** uživatel, který spravuje rezervace konferenčních místností.

**Jaké funkce bude plnit**

* Systém obsahuje následující funkce pro uživatele:  
  + **Uživatel – má** možnost rezervovat místnost, zrušit rezervaci, uvidí stav místnosti a čas, kdy bude rezervována.
  + **Administrator – má** možnost přidávat místnost, odstranit místnost, změnit vlastnosti místnosti a má stejné možnosti, které má Uživatel.
* **Systémová omezení**
* **Systém má pro uživatele omezení.**
  + Uživatel si nemůže:
    - Rezervovat 1 pokoj 2 krát (stejně jako i administrátor)
    - Vytvořit a odstranit místnost.
    - Upravit vlastnosti místnosti.
    - Zobrazit uživatele, kteří mají rezervované pokoje kromě sebe
* **Taky máme omezení pro aplikaci:**
  + - Máme omezení v čase podle reálného času. To jest my nemůžeme udělat rezervaci na příští rok nebo měsíc, ale můžeme jen v tuto dobu v tomto měsíci.
    - Nemůžeme zadávat chybná data. Budeme dostávat chybu.
* **Objektový model (UML class diagram)** 
  + UML diagram (vytvořen pomocí [draw.io](https://drawio-app.com))  
    (Demonstruje vazbu mezi entity)



* + UML diagram vytvořen pomocí[**IntelliJ IDEA**](https://www.jetbrains.com/help/idea/class-diagram.html)  
    (Demonstruje plnou vazbu mezi třídami)



* **Databáze**

Projekt využívá databázi typu PostgreSQL.

* **API**

Projekt využívá rozhraní REST API pomocí aplikace Postman.

V dokumentaci je možný najit všechna API, které pokrývají logiku projektu.

Dokumentace REST API (Postman) je možný najít zde:

<https://documenter.getpostman.com/view/22903223/2s8ZDU4Nzj>

* **Popis aplikace a její struktury**

Tato aplikace umožňuje dělat rezervaci místnosti, kde user může

vytvořit rezervaci, zobrazit jeho hodinovou cenu a zjistit

informace o něm.

Na druhou stranu má admin mnohem více možností, než user.

Kromě toho, co může dělat user, má admin plnou volnost při vytváření

místností, rezervací, userů a dalších adminů a může je také měnit a

mazat.

* **Poznámky k použití projektu**

Projekt využívá Security podporu, proto aby uživatel mohl

program používat, musí si vytvořit vlastní profil.

Při spuštění v aplikaci Postman ve složce

**REST API/Authorization/POST** můžete přihlášení potvrdit a dole se zobrazí output (viz níže obrázek) který říká, že přihlášení se dopadlo úspěšně.

Graphical user interface, text, application

Description automatically generated

## **Docker Сompose (Připojení aplikace k databázovému kontejneru****)**

## Pro připojení aplikace k databázovému kontejneru pomocí Docker Compose je potřeba zajistit, že konfigurace aplikace je správně nastavena pro připojení k databázovému kontejneru. Pro to musíme udělat následující kroky:

### **Krok 1: Konfigurace aplikace**

**•**Ujistěte se, že vlastnosti pro připojení k databázi v souboru **application.yaml** odpovídají konfiguraci databázového kontejneru definovaného ve souboru **docker-compose.yml** jako např jméno, heslo, databáze a port.

### **Krok 2: Spuštění kontejnerů aplikace a databáze**

* • Otevřete terminál nebo příkazový řádek a přejděte do kořenového adresáře projektu.
* • Spusťte následující příkaz pro spuštění kontejnerů:

**docker-compose up -d**

• Pomocí přepínače **-d** spustíte kontejnery v odpojeném režimu, takže budou běžet na pozadí.

### **Krok 3: Ověření připojení**

* • Po spuštění kontejnerů můžete ověřit připojení mezi aplikací a databázovým kontejnerem.
* • Zkontrolujte logy kontejneru aplikace, zda nejsou žádné chyby týkající se připojení:
* **docker-compose logs app**

• Pokud je připojení úspěšné, nebudete vidět žádné chyby související s připojením k databázi.  
  
Je to vše! Úspěšně jste připojili aplikaci k databázovému kontejneru pomocí Docker Compose. Nyní můžete používat aplikaci s připojenou databází.

* **Instalace aplikaci**

**!! DŮLEŽITÉ:** *Projekt funguje ve spojení s JDK 18. Před instalací a spuštěním projektu se ujistěte, že je vaše verze aktuální.*

1. Zkopírujte SSH projektu ze stránky [Git](https://gitlab.fel.cvut.cz/B221_B6B36EAR/rastvdmy).
2. Otevřete na svém počítači terminál a napište **git clone** (odkaz SSH)
3. Soubor se objeví ve složce, ke které jste zadali cestu v terminálu.
4. Otevřete soubor projektu v prostředí IDE.  
   (Doporučuji [IntelliJ IDEA](https://www.jetbrains.com/idea/))
5. Poté je třeba nakonfigurovat databázi v souboru **src/main/resources/application.yaml** a taky když chceme využívat Docker, musíme nakonfigurovat v souboru **docker-compose.yml**.
6. Otevřete **Maven**, přejděte na položku Lifecycle, rozbalte ji a klikněte na položku package   
   (poté, když budete něco změnit, tak musíte na začátku vybrat clean a pak package). Tím se projekt vygeneruje, projde všemi potřebnými operacemi a bude připraven ke spuštění.
7. Po dokončení tohoto procesu přejděte do složky **src/main/java/reservation/room/meeting/sem/ear**

a vyberte soubor **Application.java**.

1. Poté klikněte na šipku spuštění a projekt se začne kompilovat.
2. Pak spustíme [Postman](https://documenter.getpostman.com/view/22903223/2s8ZDU4Nzj), otevřeme složku **Authorization/POST**, napíšeme nakonfigurované přihlašovací údaje a můžete projekt využívat.

* **Odkazy**

Odkaz na projekt: <https://gitlab.fel.cvut.cz/B221_B6B36EAR/rastvdmy>

Odkaz na dokumentaci v Postmanu: <https://documenter.getpostman.com/view/22903223/2s8ZDU4Nzj>

* **Zpětná vazba**

Pro mě byl tento předmět něčím novým a vzrušujícím. Dozvěděl jsem se, jak funguje Java Spring Boot, k čemu slouží a jaké má funkce.

Při psaní semestrálního projektu jsem se mohl naučit o psaní Spring Boot aplikaci, propojení k databázi, použití Dockeru a Postmanu.

Semestrální projekt mě také naučil pracovat samostatně a řešit chyby, které se mi zdály obtížné.

Rád bych vyjádřil zvláštní poděkování přednášejícím a také cvičícímu Miroslavu Holečkovi za příjemný zážitek z tohoto předmětu a vklad do našich znalostí!

Děkuji vám za pozornost!