Meeting Scheduler - Rodr, Umlášek

# Popis aplikace

Naše aplikace slouží k jednoduchému plánování meetingů v pracovních týmech. Implementovaná část se soustředí na správu týmu a meetingů. Každý uživatel může být členem více týmů. Zároveň každý člen v týmu má svou roli, podle které má určité pravomoce. Role jsou: OWNER (majitel týmu), MAINTAINER (správce týmu) a MEMBER (obyčejný člen). Každý uživatel aplikace si může založit profil a následně si zobrazit detail svého profilu. Dále může založit nový tým, čímž se stavá jeho majitelem. Obyčejný člen týmu si může zobrazovat ostatní členy v jeho týmech a reagovat na založené meetingy pomocí třídy Feedback. Maintainer může zakládat meetingy, návrhy na meetingy , přidávat členy a vyhazovat členy (MEMERS) týmu. Majitel týmu má všechny již zmíněné pravomoce a k tomu může navíc měnit role členů týmu, vyhazovat maintainery z týmu a také může tým rozpustit.

## Struktura aplikace

Aplikace je rozdělena na vrstvy, podle funkcí (model, service, dao, rest, security). K aplikaci nebyl vytvořen frontend, takže její funkcionalitu jsme otestovali pomocí testů v /test package a hlavně pomocí aplikace Postman, která testuje interakci našeho programu se serverem a databází.

## Databáze

Použili PostgreSQL databázi poskytnutou naším cvičícím. Spojení aplikace s databází je definováno v adresáři src/main/resources/application.properties

## Server

Server poskytuje Java Spring. Naše aplikace běží na adrese <http://localhost:8080/meeting_scheduler_ear/>.

# Návod na spuštění aplikace

1. Stáhnout aplikaci

2. Otevřít projekt

3. Buildnout projekt s mavenem

4. Spustit aplikaci

5. Testování funkcionality aplikace přes postman (přikládáme testovací parametry – je potřeba koukat do databáze, protože jsou potřeba zadat správná id)

# Zkušenosti po vypracování SP

V našem týmu jsme s podobnými aplikacemi, které by pracovaly se Springem neměli žádné zkušenosti. Nejtěžší pro nás bylo správné mapování modelu a hlavně správné ukládání do databáze přes kaskádní persist. Tyto problémy jsme řešili debugingem přes aplikaci Postman. V průběhu práce jsme se toho hodně se Springem naučili a rozhodně bychom naší aplikaci z počátku pojali trochu jinak :D.