# Sistēmas Apraksts

Praktiskā darba risinājumam izvēlējos Spring MVC Java ietvaru ar PostgreSQL datubāzi. Šādas tehnoloģijas izvēlējos tādēļ, ka nekad iepriekš neesmu strādājis ar Java ietvariem un vienmēr esmu vēlējies izmēģināt Spring MVC, kā arī ar PostgreSQL nekad iepriekš nebiju darbojies un vēlējos izmēģināt arī to.

Spring MVC tiek izmantots pieejamo failu datu apstrādei, to attēlošanai, kā arī lai sazinātos ar PostgreSQL datubāzi. Spring projektā ir viens kontrolieris “HomeController”, kas veic atbilstošās darbības, lai lietotājs varētu navigēties pa sistēmas saskarnes lapām, ir viena Spring servisa klase “ChampionshipDataParser”, kura cikliski apstrādā visu spēļu datus, kuri atrodās resursu mapītē “championship-data”.

## Lietotāja saskarne

Kad lietotājs ienāk galvenajā sistēmas lapā, tam ir iespēja redzēt sistēmas galveno lapu un izvēlni, kurā var pāriet uz lapu “Turnīra statistika”, kā arī izvēlēties kādu no papildus darbībām – apskatīt sistēmā ielādētos failus, ielādēt jaunus datus un dzēst visus datus no sistēmas.

Kad lietotājs pāriet uz lapu “Turnīra statistika”, viņam ir iespēja apskatīt visus sistēmā ielādētos datus – kopējo un detalizēto komandu statistiku, labākos un rupjākos spēlētājus, tiesnešu statistiku un vārtsargu statistiku.

Ja lietotājs izvēlas “Dzēst visus datus no sistēmas”, viņam tiek pārjautāts, vai tiešām viņš to vēlas darīt.

Ja lietotājs izvēlas “Ielādēt datus sistēmā”, sistēma ielādē datus sistēmā un parāda par to paziņojumu. Ja visi dati jau ir ielādēti un nav neviena jauna faila, tiek parādīts atbilstošs paziņojums.

Saskarnes izstrādei tika izmantots Bootstrap HTML/CSS/JS ietvars, kurš nodrošina to, ka lapa ir responsīva un viegli pārskatāma uz kādas no viedierīcēm.

## Datu apstrāde

Praktiskā darba ietvaros izvēlējos apstrādāt JSON tipa failus, jo iepriekš esmu darbojies ar šāda tipa failiem. Sistēma apstrādā datus no *src\main\resources\championship-data* mapes. Svarīgi, lai spēļu datu mape atrastos sistēmas *resources* mapē un sauktos “championship-data”. Šai mapei ir jāsatur visu spēļu raundu mapes, piemēram, FirstRound, SecondRound, N-thRound. Šo mapju nosaukumiem nav nozīmes, taču ir svarīgi, lai mapēs atrastos tikai JSON tipa faili. Tātad, lai pievienotu jaunus datus, nepieciešams izveidot jauna raunda mapīti un ievietot tajā jaunos JSON datus.

Sistēma *apstaigā* visas mapītes un nolasa visus JSON failus. Apstrādājot katru failu, tiek veiktas dažas unikalitātes pārbaudes – ja eksistē tāda spēle, komanda, spēlētājs vai tiesnesis, tad jaunus ierakstus neveidos. Pretējā gadījumā – tiek izveidoti jauni ieraksti un ar *Data Access Object* un *Hibernate* palīdzību tiek saglabāti datubāzē. Kad visi dati ir apstrādāti un saglabāti datubāzē, tiek nodots darbs “HomeController” kontrolierim, kas veic visas nepieciešamās statistikas funkciju izsaukšanas un datu savākšanas darbības, lai varētu lietotājam parādīt turnīra statistikas datus. Kad dati tiek savākti, tie tiek saglabāti modelī kā atribūti un nodoti skatam, kurš šos datus parāda lietotājam.

Dažos gadījumos, piemēram, lai iegūtu spēlētāja vai tiesneša “nospēlēto” laiku vai lai iegūtu spēlētāju sodu skaitu, tika izmantotas PostgreSQL datubāzē izveidotās procedūras.

## Datubāze

Datubāzē dati ir sadalīti 10 tabulās (entītijās). Datubāzē ir divas galvenās tabulas – spēle un komanda, ar kurām ir saistīti visi pārējie dati. Tabulas datubāzē ir saistītas sekojoši:

