Sport League Database

CS 7330 Final Project Developer Manual

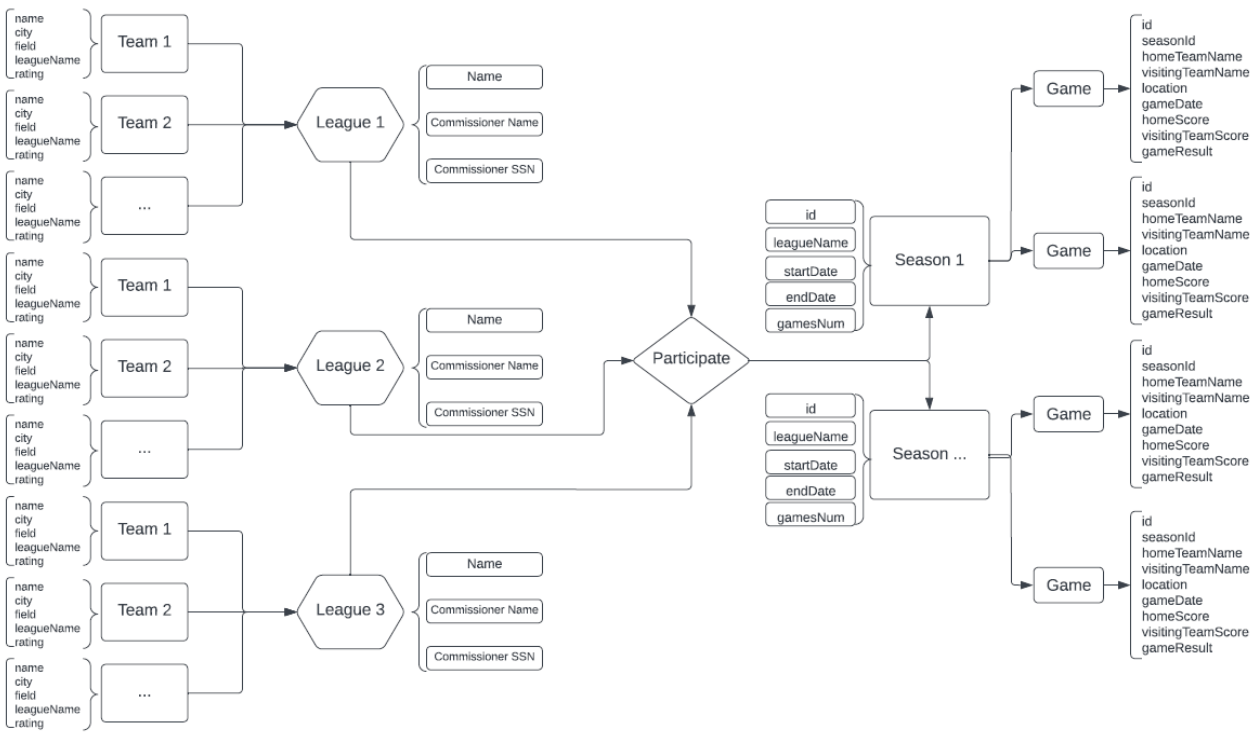
Author: Ke L., Wang Z., Zheng S. (alphabetical order)

Table of Contents:

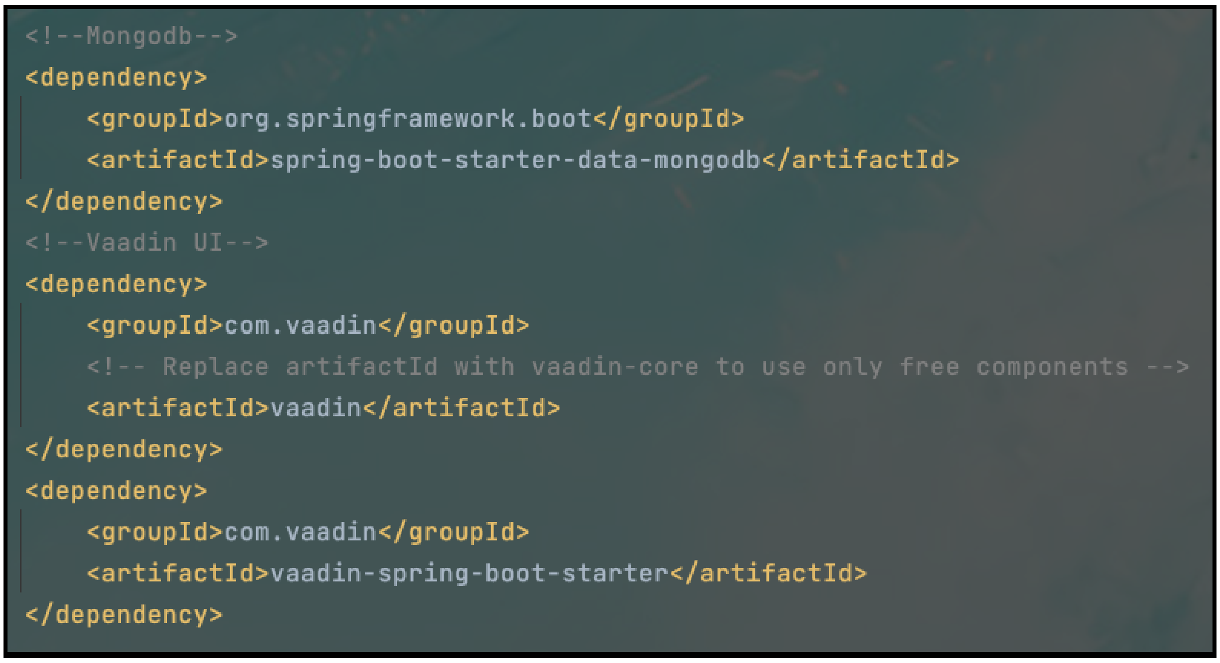
Introduction

The project is built upon a NoSQL database. It functions as a sports league tracking tool. The tool's backend would implement the basic CRUD operations with regard to each league, team, game, and season by first creating each object in MongoDB, using Spring Boot Framework (Spring Boot 2). Secondly, objects' repository interface would bridge between each object type and Mongo service repository. Lastly, more services or functions are created in each object's service interface (e.g. GameService), and those would be implemented via each implementation class through impl/ package. The frontend user interface was designed with Vadeen web design tool. Users would be able to log in via executing DatabaseTeamProjectApplication, and operate the DBS through the web page created.

Backend 1. ER Diagram



Backend 2. Dependencies



```xml
<!--Mongodb-->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-mongodb</artifactId>
</dependency>
<!--Vaadin UI-->
<dependency>
    <groupId>com.vaadin</groupId>
    <!-- Replace artifactId with vaadin-core to use only free components -->
    <artifactId>vaadin</artifactId>
</dependency>
<dependency>
    <groupId>com.vaadin</groupId>
    <artifactId>vaadin-spring-boot-starter</artifactId>
</dependency>
```

Project dependencies mainly include Mongodb and Spring Boot connection dependencies, as well as Vaadin component library dependencies, and Spring Boot web, as well as unit testing, logging and other basic dependencies.

Backend 3. Object Data Fields

Object 1. ChampionVo

Description: Stores champion information for each season and projected to the frontend.
Data Fields:

Data Type	Variable Name	Description
ObjectId	seasonId	season ID of the champion
String	teamName	Team name of the champion
String	seasonDuration	Duration of the game season
Double	points	Points the team earned from the season

Object 2. Game

Description: stores the information of each game played
Data Fields:

Data Type	Variable Name	Description
ObjectId	id	ID that are associated to individual game
ObjectId	seasonId	(Foreign key) ID that are associated to the season where the game was played
String	homeTeamName	The name of the home team
String	visitingTeamName	The name of the visiting team
String	location	The location name of the game
LocalDate	gameDate	The date scheduled for the game
Double	homeScore	the score of the home team in the game
Double	visitingScore	the score of the visiting team in the game
String	gameResult	the final result containing the scores of the two teams

Object 3. League

Description: Stores information about each league
Details:

Data Type	Variable Name	Description
String	name	name of the league
String	commissionerName	name of the commissioner

String	commissionerSsn	SSN of the commissioner

Object 4. Season

Description: stores information about each season of the game
Data Fields:

Data Type	Variable Name	Description
ObjectId	id	Unique ID of each season
String	leagueName	Name of the league that plays in the season
LocalDate	startDate	The start date of the season
LocalDate	endDate	The end date of the season
Integer	gamesNum	The number of games in the season

Object 5. Team

Description: stores the information about each team
Data Field:

Data Type	Variable Name	Description
String	name	the name of the team
String	city	the city to which the team belongs
String	field	the home field of the team
String	leagueName	(Foreign Key)the league that the team plays for
Double	rating	The rating of the team

Object 6. Scoring Criteria

Description: stores the scoring result of each team
Data Field:

Data Type	Variable Name	Description
ObjectId	id	the unique id of each team’s scoring result
OBjectId	seasonId	the season where the game belongs to
Double	wonPoints	points won
Double	drawnPoints	points drawn
Double	lostPoints	points lost

Backend 4. Object Repository

5 object repository interfaces are created and extended from MongoRepository (org.springframework.data.mongodb.repository.MongoRepository) to implement CRUD operations via. MongoDB. Each method naming method follows the naming rule of Spring Framework naming standard. The 5 repositories include:

1. GameRepository
2. LeagueRepository
3. ScoringCriteriaRepository
4. SeasonRepository
5. TeamRepository

Backend 5. Object Service and Implementations

Object service uses interface - class instantiation format, which on one hand enhances its stability and on the other hand it makes developer/database administrators' job a lot more manageable. The five service interfaces are: GameService, LeagueService, ScoringCriteriaService, SeasonService and TeamService. Impl\ package includes all the class files that implement those service interfaces.

GameService and its implementation

Method	Input	Output Type	Description
findAllGames	(String homeTeam, String visitTeam)	List<Game>	Find all games played by the two teams input
findGamesBySeason	(ObjectId seasonId)	List<Game>	Find all games by specified season
saveGame	(Game game)	String	Save games with constraints applied
autoGenerateGamesBySeason	(ObjectId seasonId)	Stirng	Automatically generate games in certain season
findGameRecordsByTeam	(String teamName)	List<TeamGameRecordVo>	Find all the games by the name of team input
updateCurrentDate	(LocalDate date)	Boolean	Set a current date chosen by the users in the system
saveSeason	Season	String	Save the season

LeagueService and its implementation

Method	Input	Output Type	Description
countLeagues	NA	Long	Count the number of leagues
findAllLeagues	(String leagueName)	List<League>	Return all leagues from DBS

saveLeague	(League league)	void	Save the league
deleteLeague	(League league)	void	delete the league
findSeasonNums	(String leagueName)	Integer	find season numbers for each league
findChampions	(String leagueName)	List<ChampionVo>	find the champions of the league in each season

ScoringCriteriaService and its Implementation

Method	Input	Output Type	Description
findBySeason	(ObjectId seasonId)	ScoringCriteria	Find the scoring criteria by seasonId foreign key
save	(ScoringCriteria scoringCriteria)	void	Save the scoring criteria

SeasonService and its Implementation

Method	Input	Output Type	Description
findSeasonsByStartDate	(LocalDate startDate)	List<Season>	Return the seasons by their start dates equal to
findAllSeasons	NA	List<Season>	Return all the seasons from the database
saveSeason	(Season season)	String	Save the data from the season
deleteSeason	(Season season)	void	Delete the season from database
findById	(ObjectId id)	Season	Find the season by the season id provided
updateCurrentDate	(LocalDate localDate)	Boolean	Set a date as current date in the system
findSeasonByCurrentDateAndLeague	(LocalDate currentDate, String leagueName)	Season	Find the season according to the current date, and league name provided

TeamService and its implementation

Method	Input	Output Type	Description
findAllTeams	(String teamName)	List<Team>	Find all teams whose name contains certain string
saveTeam	(Team team)	void	Save the team object into the database
deleteTeam	(Team team)	void	Delete the team object from database
countTeamsByLeague	(String leagueName)	Long	Return how many teams are in the league input
findFieldByTeamName	(String teamName)	String	Find the home field of the team input
findAllTeamsName	NA	List<String>	Find all the teams and return a list including their

			names
moveTeam	(Team team, String leagueName)	String	Move the team from one league to another league
findTeamNamesByLeagueName	(String leagueName)	List<String>	Find the team by the league name

---

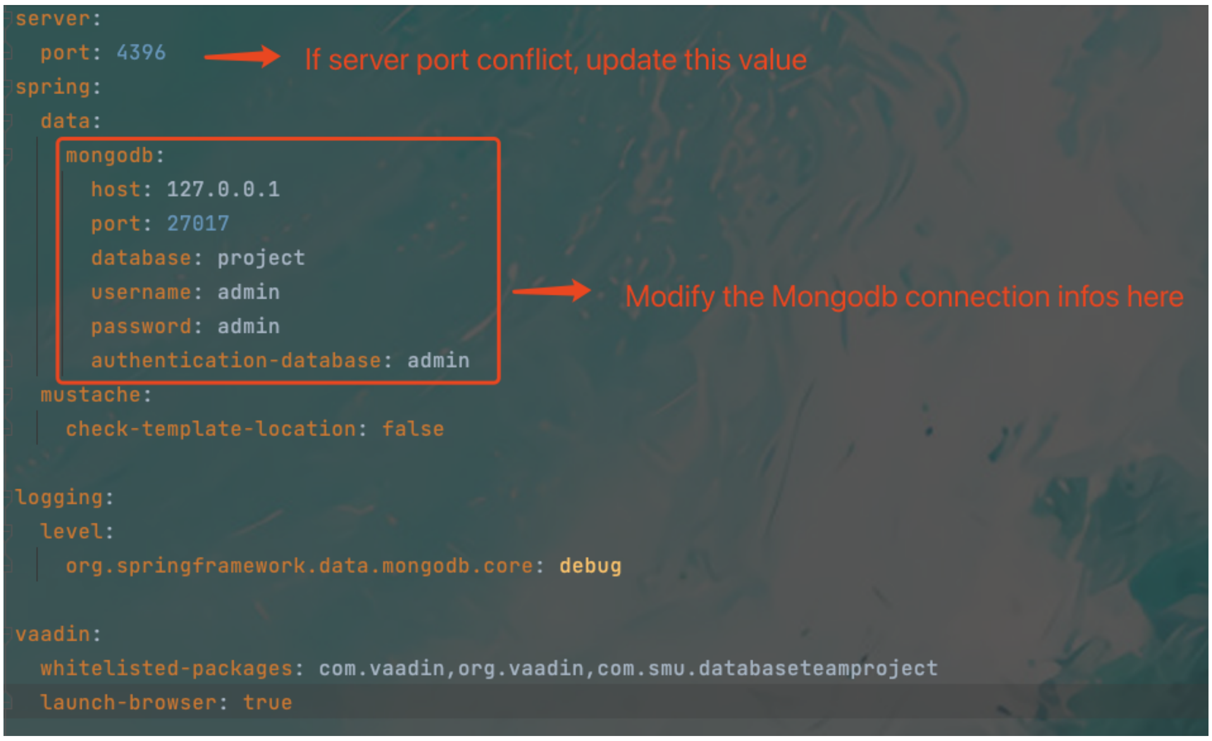
Frontend. Vaadin Developer Tool

The GUI for this project is implemented through Vaadin, the modern web application platform for Java. We use Vaadin's Grid, Dialog, Form, and other components to achieve page layout and interaction, and the Binder to achieve front and back-end data transfer, verification, and display.
To use this tool, you have to JDK 11 or higher and Node.js 16 or higher.

Cofiguration and Deploy

The final project package will be named as 'DatabaseProject_Group8.zip'.

1. Unzip DatabaseProject_Group8.zip
2. cd DatabaseProject_Group8
3. Modified the application.yml as follow:



```
server:
  port: 4396
spring:
  data:
    mongodb:
      host: 127.0.0.1
      port: 27017
      database: project
      username: admin
      password: admin
      authentication-database: admin
  mustache:
    check-template-location: false

logging:
  level:
    org.springframework.data.mongodb.core: debug

vaadin:
  whitelisted-packages: com.vaadin,org.vaadin,com.smu.databaseteamproject
  launch-browser: true
```

If server port conflict, update this value
Modify the Mongodb connection infos here

4. Execute the command 'java -jar DatabaseProject_Group8-0.0.1-SNAPSHOT.jar', and then the web page will open in your default browser automaticly.(If not, please open your browser and visit http://localhost:4396/)
5. Don't close the command window while you using the system.