CSC207 CRC Card Model

Team Scouts:

Tobey Brizuela, Daniel Lazaro, Kaartik Issar, Matthew Parvaneh, Michael Umeh, Aditya Peri

Enterprise Business Rules

(Entities)

Player

• Stores player’s **name** (string), **age** (int), **value** (int) • Stores **physical attributes**: height, weight (double) • Stores the **team** they belong to (Team object) • Stores whether they had **been scouted or not** (boolean) • Stores player’s position: forward, defender, etc. (String) • Stores **skill attributes**: tackling, ball control, speed, etc. (HashMap)	• PlayerDatabase • Team • PlayerStatsCalculator

Team

- Stores **name** of team (string), **number of players** (int)
- Stores **players in current roster** (List)
- **Net Worth** of the team (double; sum of value of every player)

- TeamDatabase
- Player
- TeamStatsCalculator

Scout

- Stores scout’s **name** (string), their **Team** (that they scout for)
- Scores **history** (List of players they have scouted in the past)
    - maybe HashMap to track dates as well
- Stores their **type**: player scout or tactical scout

- ScoutDatabase
- Player
- Team

Application Business Rules

(Use Cases)

PlayerDatabase	
• Stores an **ArrayList of Players** in the system (private, static) • addEntity for adding Players • getEntities for returning the list of Players	• Player • CSVAdapter

TeamDatabase

• Stores an **ArrayList of Teams** in the system (private, static) • addEntity for adding Teams • getEntities for returning the list of Teams	• Team • CSVAdapter

ScoutDatabase	
• Stores an **ArrayList of Scouts** in the system (private, static) • addEntity for adding Scouts • getEntities for returning the list of Scouts	• Scout

PlayerStatsCalculator	
• **rating**: generates an **overall** rating for the player based on all skill attributes • **defense**: returns an average across all “defensive” skill attributes (*e.g.* interceptions, tackling) • **offense**: returns an average across all “offensive” skill attributes (*e.g.* crossing, finishing, penalties) • **goalkeeping**: returns an average across all goalkeeping skill attributes	• Player

TeamStatsCalculator

- **rating**: generates an **overall** rating for the team based on player ratings
- **defense** and **offense** methods (also averaged from players)
- **netWorth**: sums the values of the players in the team

- Team
- Player
- PlayerStatsCalculator

SearchByPlayerAttributes

Responsibilities	Collaborators
• **searchPlayer**: search for a subset of Players in PlayerDatabase, given specific conditions to be met ◦ *i.e.* given physical attributes, value, playstyle, etc.	• Player • PlayerDatabase • PlayersPresenter

(SearchByTeamAttributes will have similar functionality)

SearchForPlayer	
• **searchPlayer**: for a specific player in the database (by name) • Returns a list of all matches (incl. partial matches)	• Player • PlayerDatabase • PlayersPresenter

(SearchForTeam will have similar functionality)

Interface Adapters

(Controllers, Gateways, and Presenters)

Interface

PresentData

- Implemented by classes that display data about a given entity
  - Entity depends on the specific implementation; player, list of players, team, etc.

- PlayersPresenter
- TeamsPresenter

PresentData

PlayersPresenter

- **outputResults**: takes in a List of Players, and displays data about each individual player (name, weight, cost, skills, etc.) in the format of a table

- Player
- SearchForPlayer
- SearchByPlayerAttributes
- StatsCalculator

PresentData

TeamsPresenter

• **outputResults**: takes in a List of Teams, and displays data about each individual team (name, net worth, roster, rating, etc.) in the format of a table	• Team • SearchForTeam • SearchByTeamAttributes • StatsCalculator

Interface

InputData

- Implemented by classes that get input from the user for a specific request
    - The prompts displayed and input accepted depends on the specific implementation

- InputPlayerAttributes
- InputPlayerName

InputData

InputPlayerAttributes

- **run**: prompts the user to enter a series of attributes
- If all inputs are valid, pass the inputs to SearchByPlayerAttributes.searchPlayer

- SearchByPlayerAttri butes

InputData

InputPlayerName

- **run**: prompts the user to enter a name, then accepts input
- If input is valid, pass the input string to SearchForPlayer.searchPlayer

- SearchByPlayerAttributes

Interface

InputAdapter

- Implemented by classes that interact with the external soccer database
    - Implementation depends on the data source (CSV, SQL, etc.)

- CSVAdapter

InputAdapter

CSVAdapter

- **stringToInt**: a helper method to format strings in the CSV to integers
- **makeHashMap**: a helper method to format all skill attributes into a HashMap
- **dataDump**: a method which iterates over the rows in the CSV input file and calls PlayerDatabase.addEntity for each row

- CSVAdapter

Frameworks & Drivers

(User Interface)

CommandLine	
• Responsible for the flow of the program • **main:** loads all players and teams into memory by calling CSVAdapter.dataDump • **runPrompts:** prints out info messages and calls all the Input___ methods in a predefined sequence	• CSVAdapter • All the InputData implementations • Basically every other class (indirectly)