State Board of Cricket Council –V6.0 *

State Board of Cricket Council

State Board of Cricket Council (SBCC) is one of the leading cricket selection academies in the state. They are in need of an automated system that should manipulate the player details provided and also find the players who have secured star rating between a specific range from the database.

You being their software consultant have been approached to develop a pilot java application which can be used by the admin for the above mentioned requirement.

Click below to download Requirement Document(s)

Requirement Document - 1/7

Requirement Document - 2/7

Requirement Document - 3/7

Requirement Document - 4/7

Requirement Document - 5/7

Requirement Document - 6/7

State Board of Cricket Council – Running Case study

Requirement 6: Manipulate Players using Collection

The State Board of Cricket Council (SBCC) is very happy with the performance of the system. Hence they decide to store a set of Player details and do some manipulations. The admin of SBCC needs to automate the system so that he can find the top three Players based on the playerType and starRating. Hence you being their software consultant have been approached to integrate this functionality into the existing system.

Component Specification: SBCCService class

Component Name	**Type (Class)**	**Attributes**	**Methods**	**Responsibilities**
	SBCCService	SBCCBoard sb=new SBCCBoard();		
Invoke the addPlayerObject method in DAO	SBCCService		public int addPlayerObject(String [] pDetails)	This method has to invoke the addPlayerObject method in

class and return the count.				SBCCBoard class by passing the input string array as an argument to perform the implementation and return the count.
Invoke the findTopPlayerDetailsmethod in DAO class and return the Map.	SBCCService		public Map<String,Double> findTopPlayerDetails(String playerType)	This method has to invoke the findTopPlayerDetails method in SBCCBoard class by passing the playerType as an argument to perform the implementation and return the Map.

To store a Player into the list

The addPlayerObject method in the SBCCBoard class takes a string array as an argument. This string array will have details of N players. This method has to invoke the parsePlayerDetails method in the SBCCUtilty class by passing each record in the input array. If the parsePlayerDetails method returns a valid player, then store that Player into the playerList. This method has to return the count of Players added to the playerList.

Note: The parsePlayerDetails method has to invoke the validatePlayerId method in the SBCCUtility class. If the playerId is valid create the player based on the player type. If the playerType is "Batsman" then, create a player of type Batsman and return the same. Else if the playerType is "Bowler" then, create a player of type Bowler and return the same. If the playerId is invalid then this method has to return null.

Component Specification: SBCCBoard(DAO class)

Component	Type	Attributes	Methods	Responsibilities

Name	**(Class)**			
	SBCCBoard	List<Player> playerList	Include the getter and setter for the playerList attribute	
Adding the object into the Collection	SBCCBoard		public int addPlayerObject(String [] pDetails)	This method has to invoke the parsePlayerDetails method in the SBCCUtilty class by passing each player details record in the input array. If the parsePlayerDetails method returns a valid player, then store that Player into the playerList. This method has to return the count of Players added to the playerList.

To find the top three Players based on the playerType and starRating

The findTopPlayerDetails method in the SBCCBoard class has to find the top three Players with the highest starRating based on the playerType passed as an argument. This method has to store the top three players' details into a Map with the playerId as key and starRating as value and return the Map.

For example if the playerList attribute of SBCCBoard class contains the following value as playerId, playerType and starRating

ASDF1234H,Batsman,3.2

ZXCV4321V,Bowler,2.0

DSAY7654D,Bowler,4.1

MNBV9876J,Bowler,1.2

HGFD8754J,Batsman,7.0

BVCH9843L,Batsman,5.2

NBVT9807O,Bowler,7.8

HGTY9854,Batsman,6.5

And If the playerType passed as argument is **Batsman** then the Map should contain the following values

Key (playerId)	**Value (starRating)**
HGFD8754J	7.0
HGTY9854	6.5
BVCH9843L	5.2

If the playerType passed as an argument is **Bowler** then the Map should contain the following values

Key (playerId)	**Value (starRating)**
NBVT9807O	7.8
DSAY7654D	4.1
ZXCV4321V	2.0

Component Specification: SBCCBoard

Component Name	**Type (Class)**	**Attributes**	**Methods**	**Responsibilities**
Finding the	SBCCBoa		public	This method has to

top three Players based on the playerType	rd		Map<String,Double> findTopPlayerDetails(String playerType)	manipulate the playerList and find the top three players with the highest starRating based on the playerType passed as an argument. If the playerType passed as an argument to this method is **Batsman**, then this method has to store the top three **Batsman** details with the highest starRating into the Map with the playerId as key and starRating as value and return the Map. If the playerType passed as an argument to this method is Bowler, then this method has to store the top three Bowler details with the highest starRating into the Map with the playerId as key and starRating as value and return the Map.

Assumption: The playerList will contain minimum three players with the **top starRating**

Component Specification: SBCCUtility Class

Component Name	**Type(Class)**	**Attributes**	**Methods**	**Responsibilities**
Parse data, and Construct	SBCCUtility		public Player parsePlayerDetails(String	This method takes the String which holds all the player details as an argument. This

Player Object if the playerId is valid			playerDetails)	method has to invoke the validatePlayerId method in the SBCCUtility class by passing the playerId as a parameter, If the playerId is valid create the player based on the player type. If the playerType is "**Batsman**" then, create a player of type Batsman and return the same. Else if the playerType is "**Bowler**" then create a player of type Bowler and return the same. If the playerId is invalid then this method has to return null.
Validating the playerId and Exception Handling	SBCCUtility		boolean validatePlayerId(String playerId)	This method should validate the playerId, if valid return **true** else this method has to throw a user defined Exception "**InvalidPlayerIdException**" with a message **"Player with Id <<playerId>> is not valid".**

Component Specification: InvalidPlayerIdException (Exception Class)

Component Name	**Type(Class)**	**Attributes**	**Methods**	**Responsibilities**
Creating an Exception class	InvalidPlayerIdException		public InvalidPlayerIdException(String message)	

Component Specification: Player (Model class)

Component Name	Type (Class)	Attributes	Methods	Responsibilities
Create an abstract method	Player		public abstract void findStarRating()	
	Player	String playerId String playerName int matchesPlayed int runScored String playingZone	Include all necessary Getters and Setters for all the attributes Provide a no argument and a five argument constructor in the given order playerId, playerName, matchesPlayed, runScored and playingZone.	
Calculate total runs scored by the Player	Player		public int calculateTotalRuns(String[] securedRuns)	This method takes a String array as an argument which contains the runs scored by the player in each match. It has to calculate the total runs scored by the player by summing the runs scored by the player in each match and return the sum.

Component Specification: Batsman (Model Class)

Component Name	Type (Class)	Attributes	Methods	Responsibilities
	Batsman	int noOfHundreds int noOfFifties double starRating	Include all necessary Getters and Setters for all the attributes Provide a no argument and a seven argument constructor in the given order playerId, playerName, matchesPlayed, runScored, playingZone, noOfHundreds and noOfFifties	
concrete sub class	Batsman		public void findStarRating ()	This method has to calculate the rating of the player based on the number of hundreds and number of fifties scored by the batsman and set this rating value to the starRating attribute in the Batsman class.

Component Specification: Bowler (Model Class)

Component Name	Type (Class)	Attributes	Methods	Responsibilities
	Bowler	int noOfMaiden int noOfHattrick	Include all necessary Getters and Setters for	

		double starRating	all the attributes Provide a no argument and a seven argument constructor in the given order playerId, playerName, matchesPlayed, runScored, playingZone, noOfMaiden and noOfHattrick	
concrete sub class	Bowler		public void findStarRating ()	This method has to calculate the rating of the player based on the number of maiden overs and number of hat-trick wickets taken by the bowler and set this rating value to starRating attribute in the Bowler class.

In UserInterface class, **In the main method provided, fill the code to produce the output as shown in the Sample input and Output.**

When user selects the option **1 i.e., Validate player details**, it should get the player details from the user, and invoke the method to parse the player details. If valid player is returned then display the player details such as playerId, playerName, matchesPlayed, runScored and playingZone, else display "**Please provide a valid record**" along with the exception message "**Player with Id <<playerId>> is not valid**" if invalid player details found.

When user selects the option **2 i.e., Create Batsman or Bowler**, it should get the player details from the user, and invoke the method to parse the player details. If valid player is returned then display the player details based on the player type (Batsman or Bowler), else display "**Please provide a valid record**" along with the exception message "**Player with Id <<playerId>> is not valid**" if invalid player details found.

When user selects the option **3 i.e., Validation with InvalidPlayerIdException**, it should get the player details from the user, and invoke the method to parse the player details. If valid player is returned then display the player details based on the player type (Batsman or Bowler), else display "**Please provide a valid record**" along with the exception message "**Player with Id <<playerId>> is not valid**".

When user selects the option **4 i.e., Add Player Details**, it should get n player details from the user, and invoke the addPlayerObject method in SBCCService class, in turn this method has to invoke the addPlayerObject method in SBCCBoard class to perform the implementation and return the count. Display the count returned by that method as "**<<count>> valid records found**" along with the exception message "**Player with Id <<playerId>> is not valid**" if invalid player details found.

When user selects the option **5 i.e., Top three players**, it should get player type (Batsman or Bowler) from the user, and invoke the findTopPlayerDetails method in SBCCService class, in turn this method has to invoke the findTopPlayerDetails method in SBCCBoard class to perform the implementation and return the Map containing player details. Display the player details returned by the method.

When user selects the option **6 i.e., Exit**, display the message "**Thank you for using SBCC application**" and end the program.

OVERALL DESIGN CONSTRAINTS:

- The Player class should be inside the package com.sbcc.model
- The Bowler class should be inside the package com.sbcc.model
- The Batsman class should be inside the package com.sbcc.model
- The SBCCUtility class should be inside the package com.sbcc.utility
- The UserInterface class should be inside the package com.sbcc.main

- The InvalidPlayerIdException class should be inside the package com.sbcc.exception
- **The SBCCService class should be inside the package com.sbcc.service**
- **The SBCCBoard class should be inside the package com.sbcc.dao**
- Adhere to the design specifications mentioned in the case study.
- **The classes and methods should be declared as public and all the attributes should be declared as private.**
- **Do not change or delete the class/method/attributes, names or return types which are provided to you as a part of the base code skeleton.**
- Please make sure that your code does not have any compilation errors while submitting.

In the main method provided, fill the code to produce the output as shown in the Sample input and Output.

In the Sample Input / Output provided, the highlighted text in bold corresponds to the input given by the user and the rest of the text represents the output.

Sample Input and Output 1 [Values given in bold represents the input]:

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

4

Enter the number of player details

10

Enter the player details

ABCD1234N:SarathSingh:10:109:51:102:65:99:101:86:100:50:183:North:Batsman:5:5

HXCB1234N:DhoniMahee:5:124:150:0:0:50:North:Batsman:2:1

HX1234N:MaratSah:5:24:23:0:76:50:North:Bowler:8:3

DSAP2345S:Hareesh:3:50:10:15:South:Bowler:3:1

ABCD3456E:FarhanSingh:8:24:76:6:98:1:65:3:72:East:Bowler:5:3

ABCD36E:Rohitkuna:4:24:6:6:9:East;Batsman:0:0

BXTP9876W:Jadeja:2:179:98:West:Batsman:1:1

SADF2365S:AshwinKumar:6:20:0:0:30:10:0:South:Bowler:2:3

MNHG5439E:KapilDevRaj:7:10:50:100:0:51:60:43:North:Batsman:1:3

MNFG5439F:Rajeev:1:0:North:Bowler:8:1

Player with Id HX1234N is not valid

Player with Id ABCD36E is not valid

8 valid records found

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

5

Enter the player type

Batsman

ABCD1234N 7.5

MNHG5439E 1.75

HXCB1234N 1.25

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

5

Enter the player type

Bowler

ABCD3456E 4.4

SADF2365S 2.4

DSAP2345S 0.75

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

1

Enter the player details

HXCB1234D:Dhoni:5:20:130:55:102:100:North:Batsman:3:1

Player id: HXCB1234D

Player name: Dhoni

No. of matches played: 5

Total runs scored: 407

Playing zone: North

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

1

Enter the player details

HXC234D:Dhoni:5:20:130:55:102:100:North:Batsman:3:1

Player with Id HXC234D is not valid

Please provide a valid record

// Note: Display the message << Please provide a valid record>> if the parsePlayerDetails method returns null

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

2

Enter the player details

HXCB1234D:Dhoni:5:50:130:55:102:100:North:Batsman:3:2

Player Id: HXCB1234D

Player Name: Dhoni

No. of matches played: 5

Total runs scored: 437

Playing zone: North

Number of Hundreds: 3

Number of Fifties: 2

Star Rating: 2.0

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

2

Enter the player details

SAFG1243P:Mahee:3:20:30:55:South:Bowler:4:0

Player Id: SAFG1243P

Player Name: Mahee

No. of matches played: 3

Total runs scored: 105

Playing zone: South

Number of Maidens: 4

Number of Hattricks: 0

Star Rating: 0.6

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

2

Enter the player details

HXC234D:Dhoni:5:20:130:55:102:100:North:Batsman:3:1

Player with Id HXC234D is not valid

Please provide a valid record

// Note: Display the message << Please provide a valid record>> if the parsePlayerDetails method returns null

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

3

Enter the player details

HB1234D:Dhoni:5:120:50:55:198:100:North:Batsman:3:2

Player with Id HB1234D is not valid

Please provide a valid record

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

3

Enter the player details

SAFG1243P:Mahee:3:20:30:55:South:Bowler:4:0

Player Id: SAFG1243P

Player Name: Mahee

No. of matches played: 3

Total runs scored: 105

Playing zone: South

Number of Maidens: 4

Number of Hatricks: 0

Star Rating: 0.6

1. Validate player details

2. Create Batsman or Bowler

3. Validation with InvalidPlayerIdException

4. Add Player Details

5. Top three players

6. Exit

Enter your choice

6

Thank you for using SBCC application