Sprint Retrospective Iteration # 3

Group:42
Project:Air Hockey

User Stories	Task/Issue #	Assigned To	Estimated Effort	Actual Effort	Done(yes/no)	Notes
-The player shall be able to see the top 5 scores in the game at the end of each play	#24 Top 5 scores at the End	Robert and Ionut	2h	2h	yes	The scores of the top players are now displayed
-A new interface named PuckState will be introduced, that will define the behavior of the puck since it depends on its position on the board (Gate Aligned and Out of Gate range).	#32 State Design Pattern to Puck	Robert	2h	2h	yes	We have chosen the State design pattern for the puck as its behavior changes depending on its position
-When the 2 player game is going to start, the second player will also be asked about his credentials, in order to save the statistics of the game after it ends for both players.	#31 Login Feature for the Second Player	Robert & Darwin	5h	5h	yes	Now 2 players can now login to play the game. Improvements to the login functionality have also been made
- The player will hear a hit sound on hitting the puck and the puck hitting the wall and hear a horn sound on scoring	#12 Sound Effects	Jaron	2h	2h	yes	Now the sound effects on hit ad scoring work properly
- The player will see a leaderboard, which contains the current top 10 player from the database, when clicking scores on the menu screen and it'll disappear when clicked again	#18 Leaderboard	Jaron & Ionut	3h	4h	yes	The leaderboard has been designed to be easily extended (in size) in future versions
-Improve the code by implementing the Data Access Object design pattern for the database controller classes to create a clear separation between game logic and database logic. Improve code reuse and maintainability.	#33 DAO pattern refactoring	Ionut	3h	7h	yes	Took longer than expected but the DAO pattern was implemented and code quality was improved
-The player will be able to see the following statistics for his account: number of games won, number of games lost, total points, most recent game history and the individual leaderboard position	#10 Performance statistics	Jaron & Ionut	3h	3h	yes	Just the first user that logs in will be able to see his account stats

Sprint Overview (Team Level)

This sprint we managed to complete the more fine tuned aspects of our application. We have also worked and are still working on improving the code. Communication was good, although some persons reacted to problems too close to the final deadlines, taking some maybe unnecasary risks, so scheduling was a problem.

Main Problems Encountered	Reaction
Robert	
#1 The Software Architecture document and diagrams exercise was more ambiguous than expected, especially because we had to think how to fit our application in one and because there are not that many examples on how to make component diagrams in the lectures	Had to search on the internet for some examples
Darwin	
#2 Handling of user input causing failed authentication and registration	
Jaron	
#3	
Ionut	
#4 Some tests were failing because the equals method was not implemented in some of the classes (could not understand the failing reason)	Generate equals methods
#5 For a specific test class the method calls from one mocked class were not recognized for some reason	Cleaning caches and rebuilding worked without changing anything to the code itself
#6 PMD static analysis warnings would give a warning for some things that were solved (such as closing connections after use).	Suppres some PMD warnings (explanations given)
#7 Refactoring was not easy and for a few times I had to delete some code I wrote because it proved to be inefficient/useless	
#8 Issues when reusing ResultSet and PreparedStatement for the private getSalt method that was called during authentication. Error messages said something was wrong with the hashing function but actually the problem was related to the database interaction	Create separate ResultSet and PreparedStatement instances instead of reusing the existing ones

Adjustment for next Sprint:

1. Try to implement features a lot earlier if possible, in order to prevent any struggles during the last days of the sprint (same problem as last sprint)
2. Some code improvements for the final product are in order
3. Carefully consider what and how we want to implement something in order to avoid writing code that later has to be deleted.
4. Reconsider the way we handle exception errors (printing them to the console might not be useful for the user)
5. Try to make smaller merge requests when possible.