**Java guessing game**

**Video demo URL:**

[**https://www.youtube.com/watch?v=Yq7oDlcCUiA&feature=youtu.be**](https://www.youtube.com/watch?v=Yq7oDlcCUiA&feature=youtu.be)

**ABOUT PROJECT:**

For my final project I decided to make a word game, with my unique touches. The name of the game is called “GUESS”, where a user will be given a time of 3 minutes (180 seconds) to guess as much words as possible.

The game is designed to improve the spelling skill for the younger generation. Since we are in the age of technology, no child wants to pick up a book and do a crossword puzzle.

**HOW TO PLAY/ RULES:**

The player to guest the highest amount of words will be ranked the highest on the score board.

If a player gets 5 strikes the game will be over

If the timer of 3 minutes runs out the game would be over

The user can also click to exit the game

The timer does not begin until the user clicks the first letter.

The strike section will turn red for every incorrect letter input.

If a user does not enter a name their user name will be e.g. “Anonymous 1” which is used as the primary key.

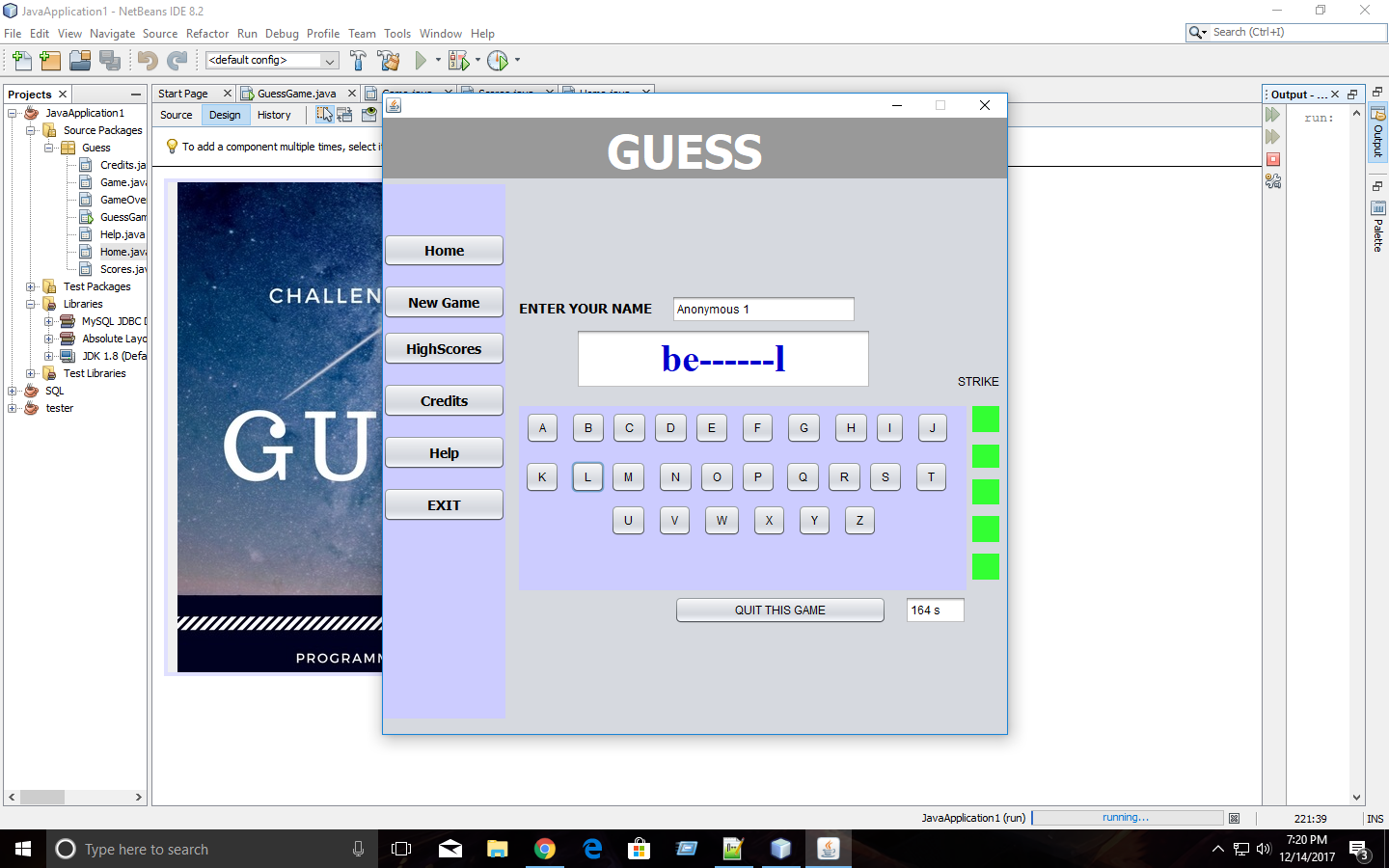
On the start of every game, an encrypted version of the word is displayed along with a hint of the first letter. E.g. if “costaatt” is the word the display c------- will be shown to the user.

**Requirements**

1. **Database**

For this assignment I decided to use a database because it was best suited to the application I am building oppose to networking. The database is used to keep track of user scores and their information related to that particular game. File storage can also be used to store this but a database is much faster and effective. The additional benefit of having a MYSQL database is the ability to sort my results base on the top performers which means less code to be written in JAVA.

1. **Threads**

I used threads to implement my timer.

Timer delays when the user uses this part of the application.

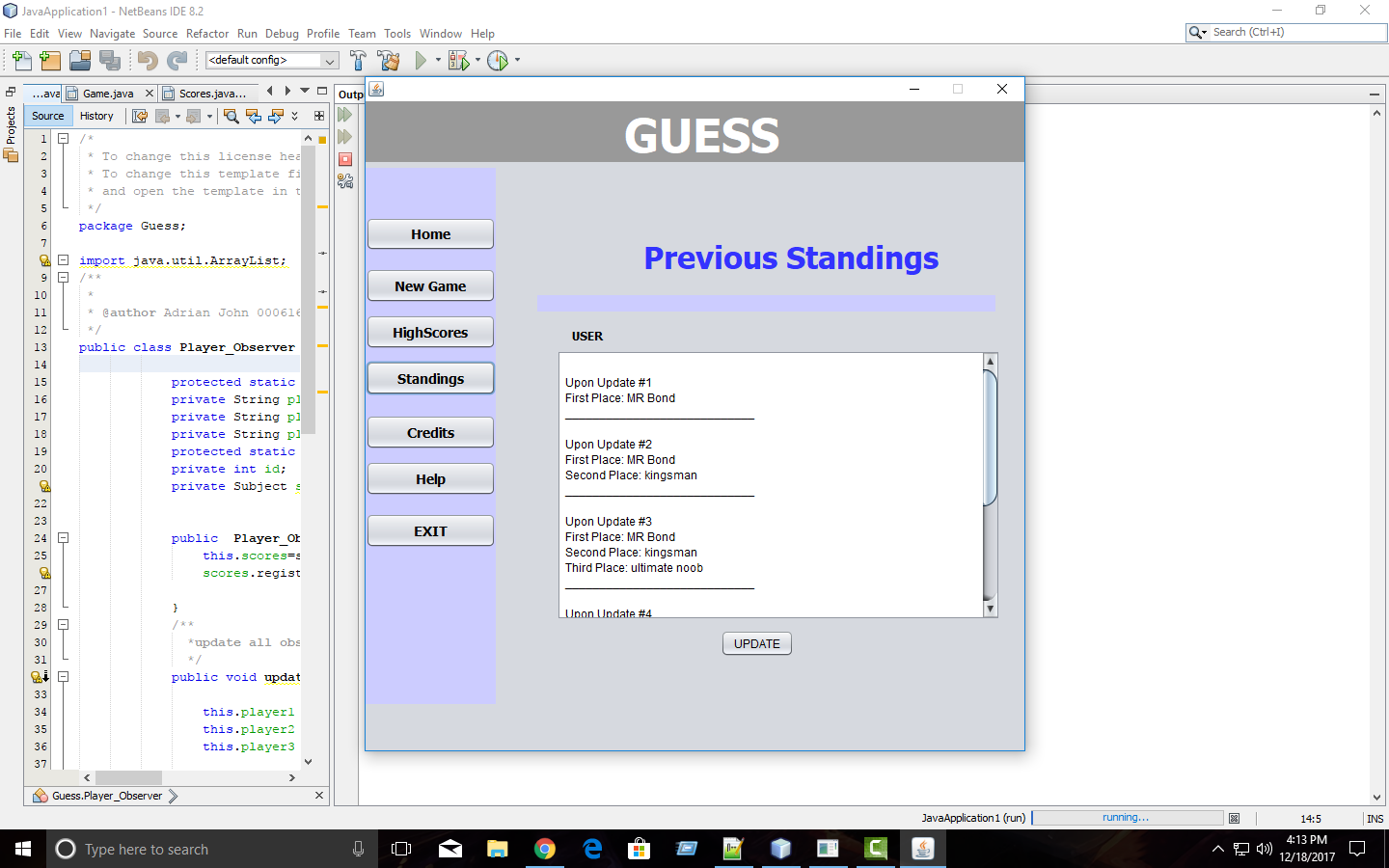
There was a thread running my entire application and there was a thread running my timer. Without threads, if the user enters a letter for example, the timer would delay or stick for a couple Nano second which would make the game very ineffective. Threads solved this problem causing the application and timer to run smooth and separately.

**Design Pattern - Observer**

For this assignment I made use of the observer pattern. The goal was to display the player in first, second and third place after a new player plays the game.

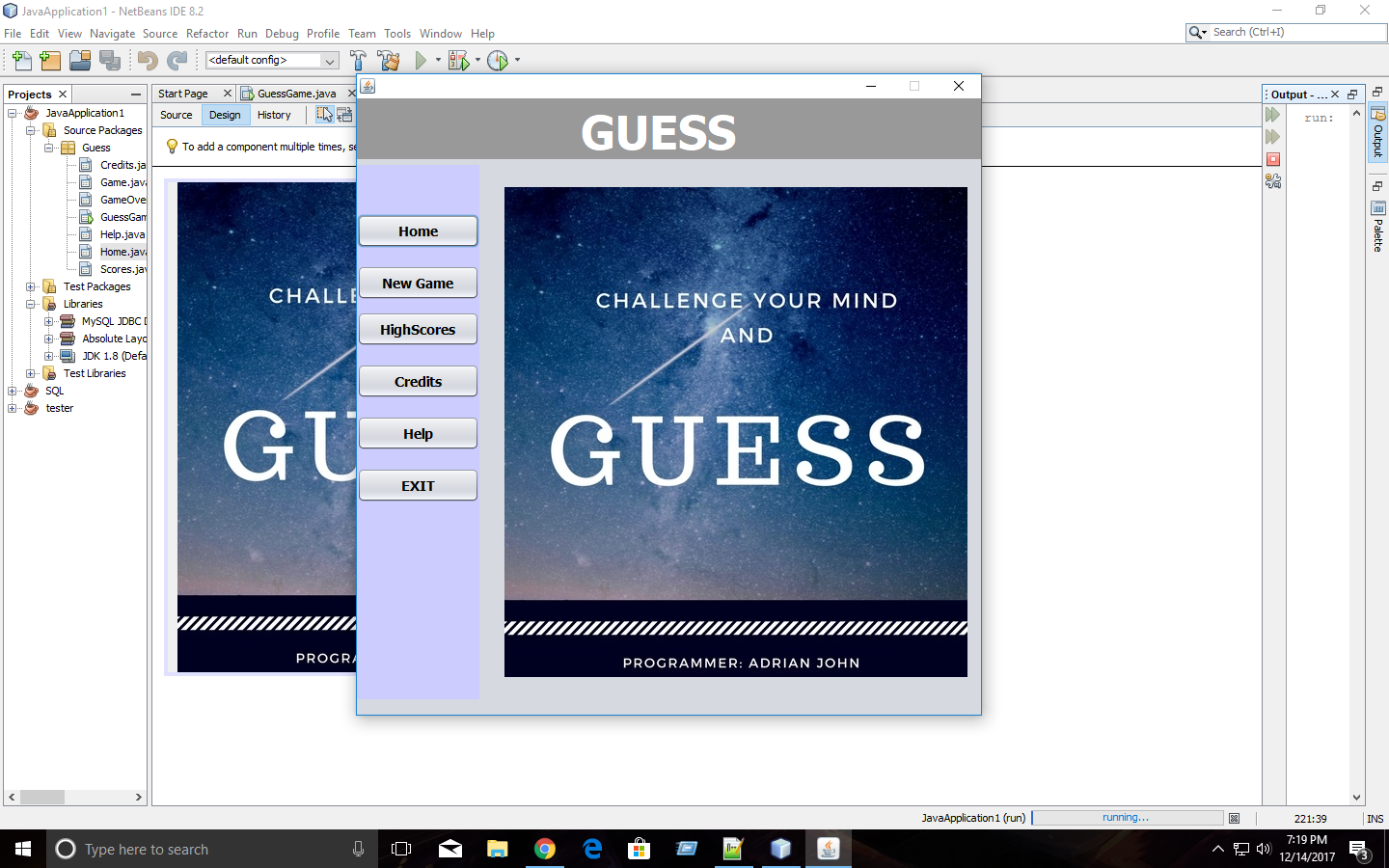
Without using the observer pattern, I would have to write a lot of if statements and unnecessary looping. The observer pattern allows the program to run more efficiently.

I read the data from the database in the order that the player played the game; e.g. The very first player who played the game is set first and last player is set last.

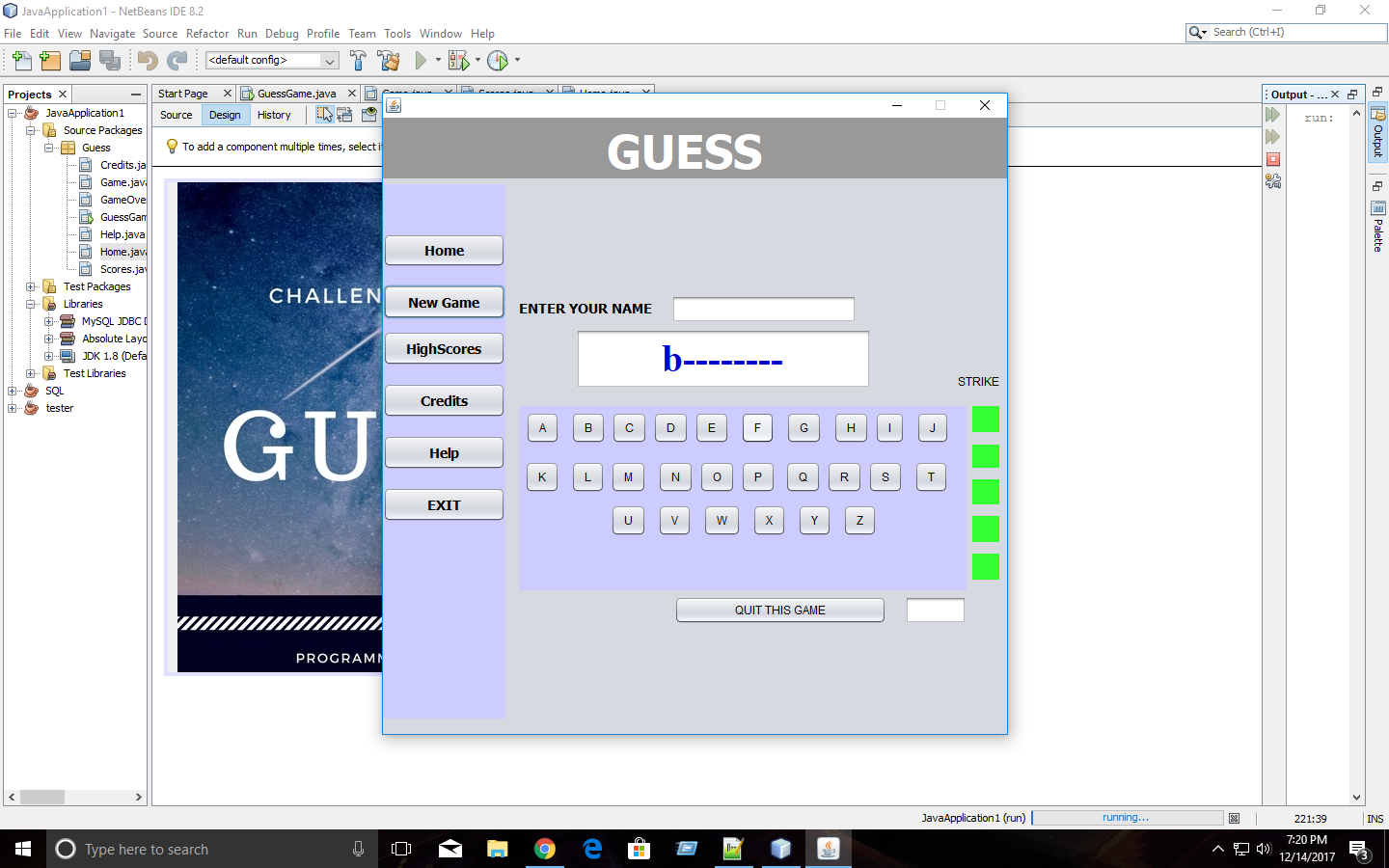
****

**As shown to the left, every time the first, second or third place gets updated, all observers are notified.**

**Panels of the application:**

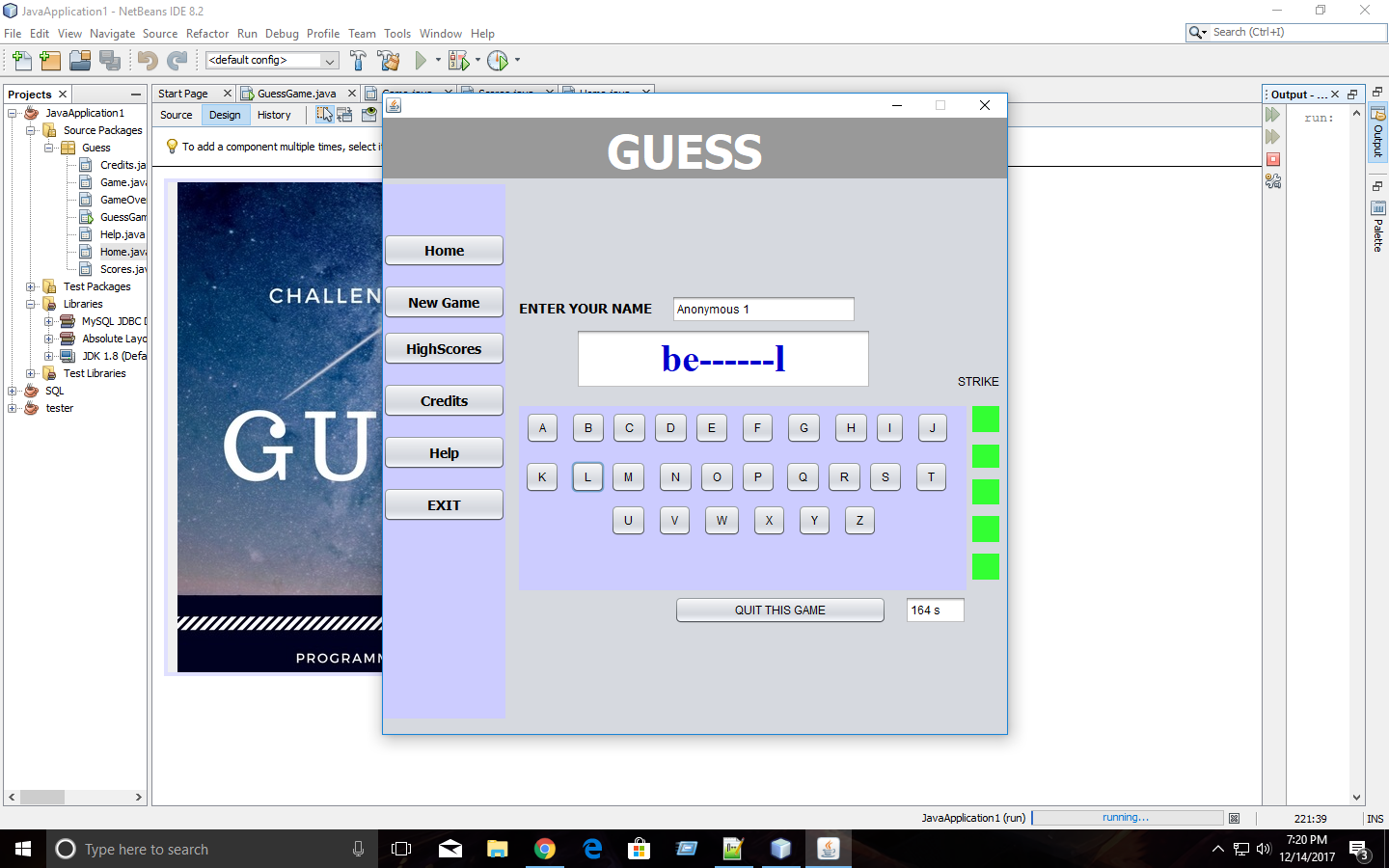
**Home screen:**

Upon opening the application, the user will be greeted with a user friendly home screen and a menu with the following options.

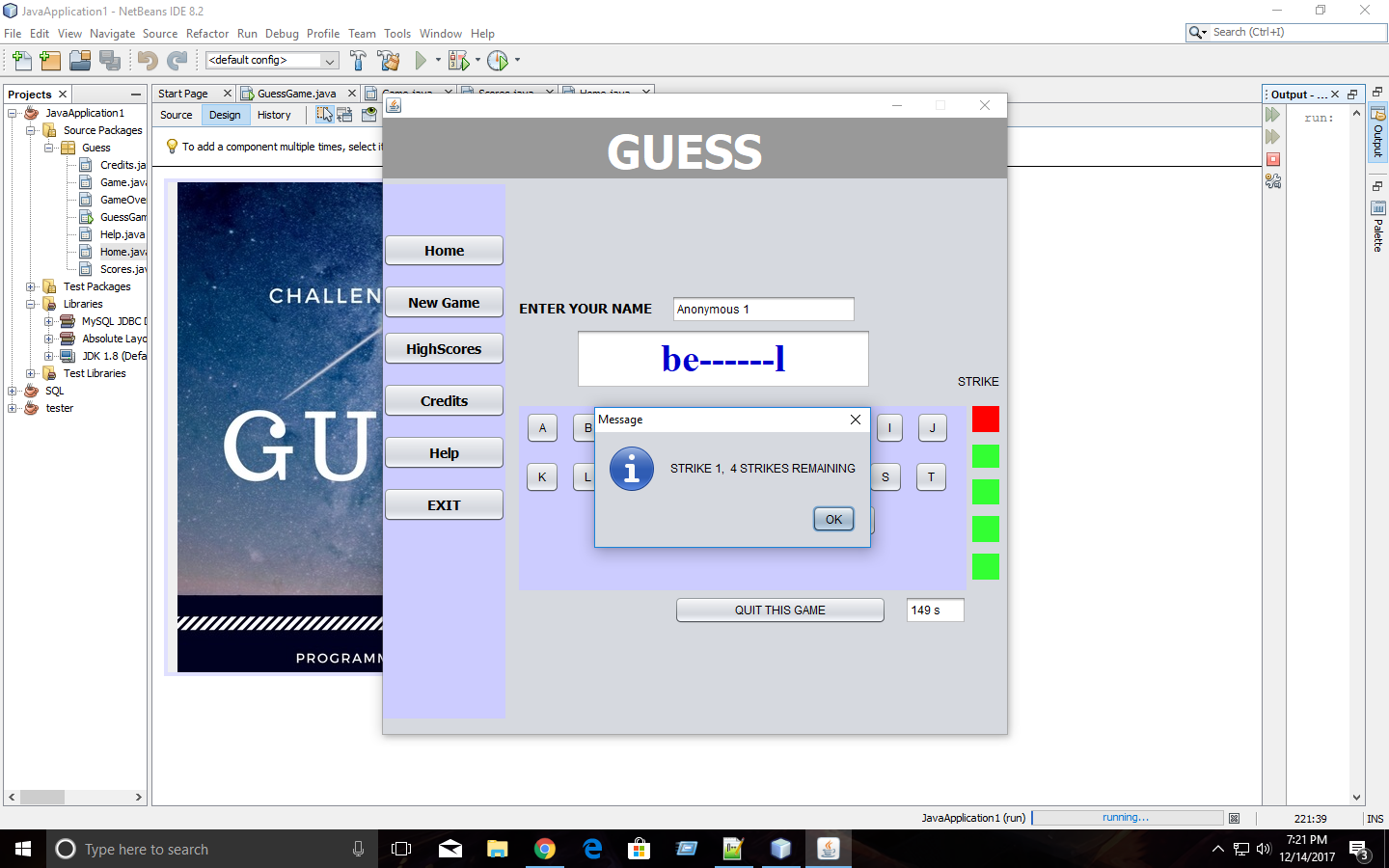
**New Game:**

If the user selects a new game, the following screen will be displayed. First the user must enter a username, which would later be used to identify players in the database. If the user does not input a name, the application automatically assigns a username for e.g. “anonymous 1 etc.”

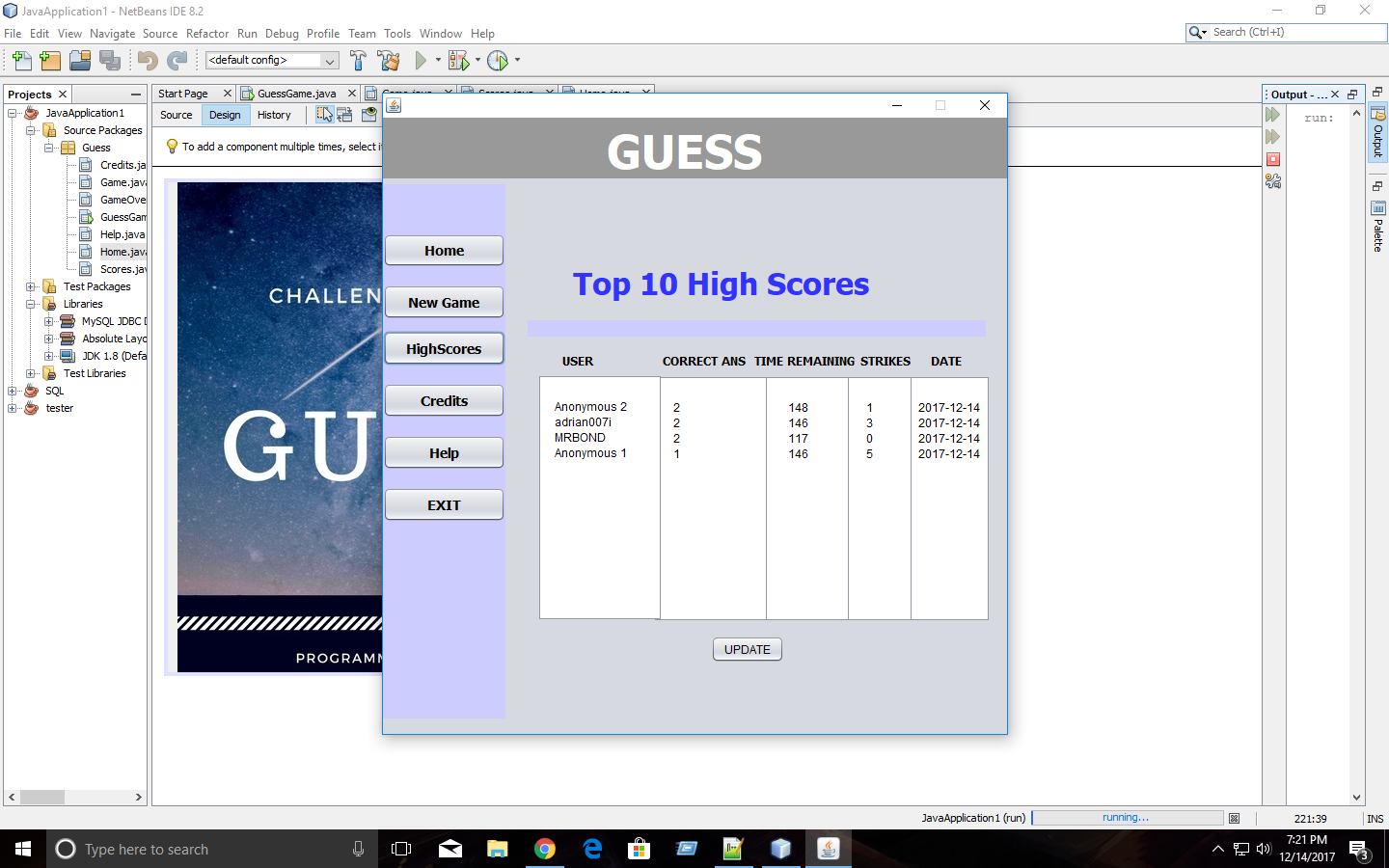
As shown to the left, a random word is selected for the user to spell, showing the first letter of a word along with encrypted characters of the other letters.

When the user clicks the first letter, the timer will begin.

If the letter entered by the user is contained within the word, the letter is added to the appropriate position of the encrypted display.



If the user enters a incorrect letter, the would be alerted by a JOptionPane and the would get a strike. Each strike the user gets will change the color of the box from green to red.

**High Scores:** 

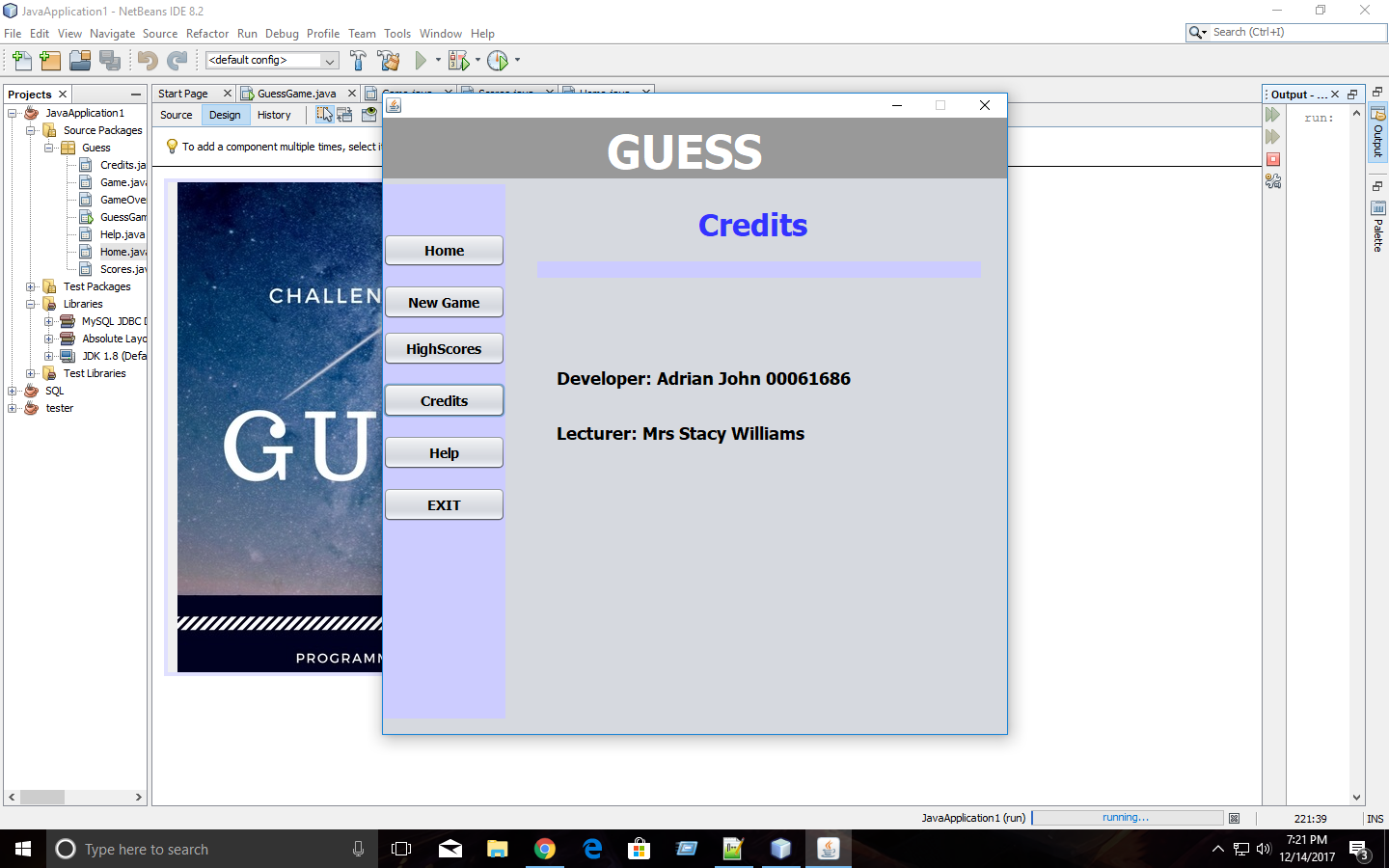
The players with the top 10 high scores will be displayed. The data to the left is stored in a database.

The data includes;

* The username of the player
* Amount of answers the user get right
* The amount time remaining
* The amount of strikes
* And the date the player got the score

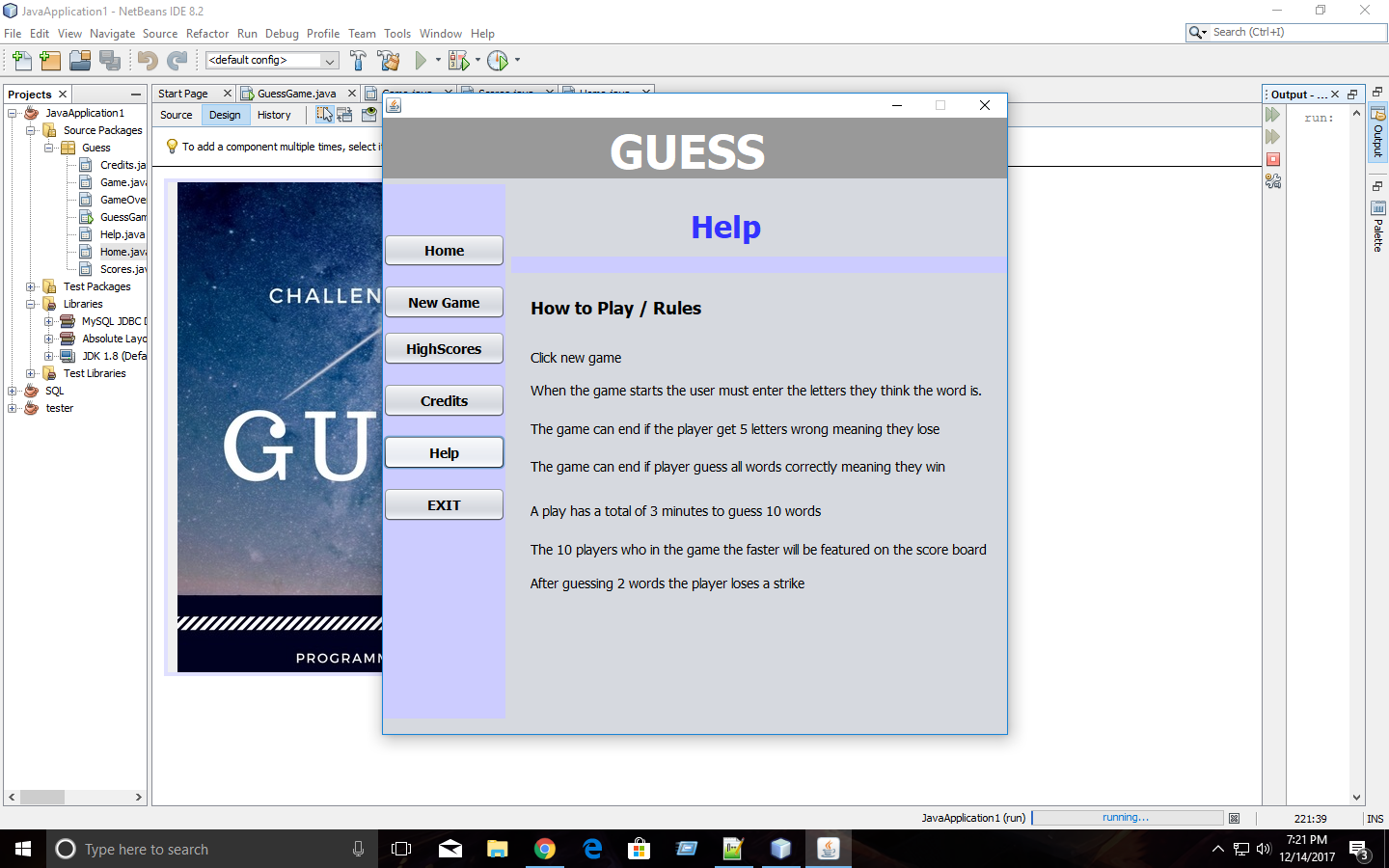
The scores are sorted using the correct answer and the time remaining.

The player with the most answered questions and most time to spear will be at the highest rank.

 **Credits:**

Just a display of the people involved with the creation of the application.

**Help:**

A display of the rules and tips of the game.