**Guess the number Spring Boot Rest service functionality and explanation**

**Functionalities implemented:**

* CRUD methods for database in the data access object side – able to create a game object, read the objects in array list, update a current object when a guess is made, delete a game once its finished or just to delete it purposefully.
* Java class model
* Uses Spring Boot application with dependencies all in the POM file and connectivity to database in the application.properties
* HTTP METHODS are all in the Gamecontroller class – implemented the post mapping which will request information from the client side or postman in this case and send back to the server after. In my program I used a post method to begin a game which needs to be directed at the url “/begin” to be able to retrieve the correct resource from correct destination.
* Another post was to let the user have a guess during the game which will then update the current games in the system (local memory array list or database side). The guess method was crucial in order for the game to be updated and ensure there is actual gameplay, the user must pass in the gameId first of all to be able to modify the correct details and then input 4 numbers as a guess which will be compared with the randomly generated number.
* There is a get request which returns all the current games that have happened by returning the List of each game number object created.
* A get request which will return the gameId the user is looking for and the extra information of that specific game.
* These methods are all in the controller class and they are called through the service layer and implemented in the actual data layer. The service layer is to provide abstraction and reusability of code for the future.
* Have tested the game rules but not implemented full unit testing – screenshots are provided to show functionality working
* The controller handles all of the requests and never directly access the dao implementation – only invoked through service layer.
* Annotations are all specified in the controller class as stated in the requirements

**Errors couldn’t resolve properly:**

When adding the object to the database and trying to do a post to make another guess it would give the error that duplicate id’s exist in the database. So to still be able to test the game I removed the object from being added into the database first and test it then I tried it with adding it to database.

Text

Description automatically generated

Testing the methods

Graphical user interface, text, application, email

Description automatically generated

**Database:**

**Graphical user interface, text, application, table

Description automatically generated**

**Testing**

Current list of games before anything started:

Graphical user interface, text, application

Description automatically generated

Creates a game and returns the ID

Graphical user interface, text, application, email

Description automatically generated

When refreshing the app/game you can see it has updated - **get game request**

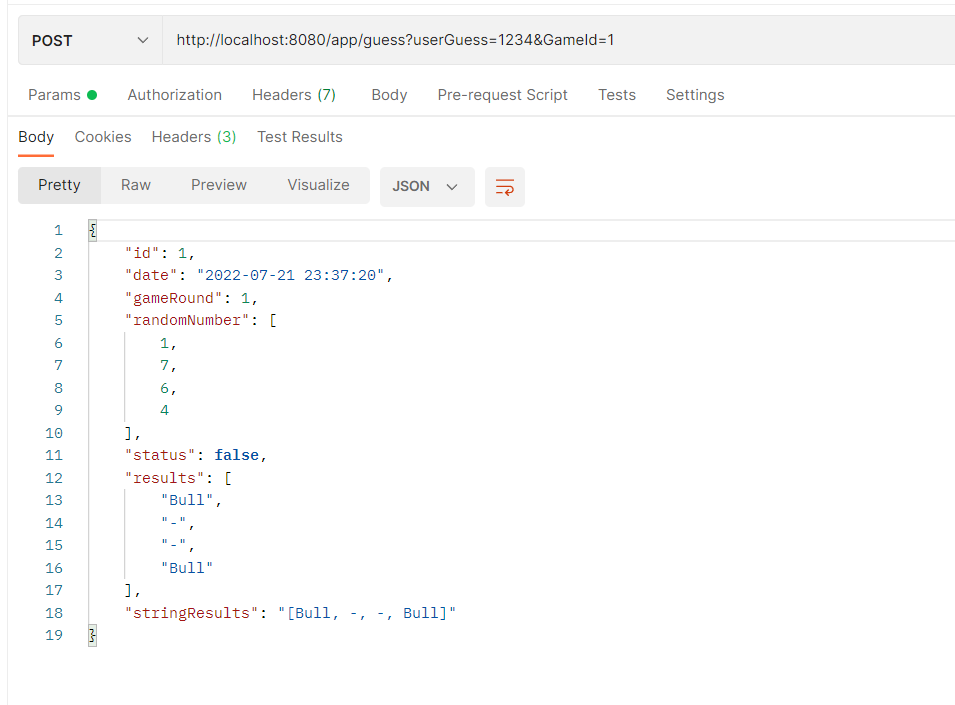
A picture containing website

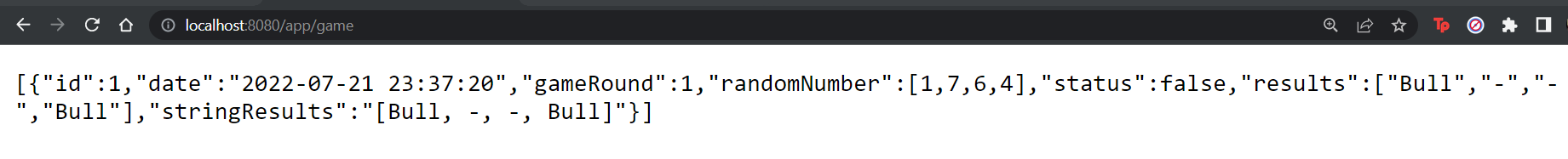
Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

After making a guess in POSTMAN





Round updates when trying again

Graphical user interface, text, application, email

Description automatically generated