**Project Final Report**

**Project Name :** Q&A –BG ( Board Game)

**Group Members and Project Contributions :**

* **Y\*\*\*s E\*\*e K\*\*a :** Made the network functions.
* **Burhan Sancakli :** Made the visual engine and merged all our codes together.
* **C\*\*\*\*\*e Y\*\*\*\*\*i :** did the encryption and the decryption.
* **M\*\*\*\*\*\*d F\*\*\*\*n Ö\*\*\*l :** Created the question and answer parts.
* **R\*\*a A\*\*\*\*i :** Made the regular expressions required for the chat.

|  |  |
| --- | --- |
| Members | IDs |
| Y\*\*\*s E\*\*e K\*\*a | 2\*13 |
| Burhan Sancaklı | 2\*83 |
| C\*\*\*\*\*e Y\*\*\*\*i | 2\*63 |
| R\*\*a A\*\*\*\*i F\*\*\*\*\*d | 2\*75 |
| M\*\*\*\*\*\*d F\*\*\*\*n Ö\*\*\*l | 2\*17 |

**Team Rituals Roles:**

We have been holding meetings every week. In the beginning, we have discussed our ideas about the project subject, so after choosing the Q&A Board Game, we have talked about the different requirement that such a game could have, and accordingly, we distributed the tasks between us so that every member focuses on learning a specific part of the project. We then, held meetings, and had 4 sprints, each time we test and see if there are any bugs or errors to fix them.

**Application Specifications:**

* The game is basically a custom multiplayer board-game and a Q&A game at the same time:
* The game is started by running the “run.cmd” file, the file then installs all the required dependencies and runs the “main.dart” file in the “bin” folder.
* People join and the game starts (people can also hot join into the game as well).
* A new question comes.
* It gets answered by writing "/answer d" or "/answer b" etc.
* Every time someone answers a question correctly, he/she gets 10 points and his character moves 1 tile.
* The game ends when players run out of questions to answer (we have a pool of 20 questions so it is not too long).
* The player with the highest score basically wins when the game ends.

**Implementation and Validation:**

**Network and the Game Engine Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case # | Test Case Description | Test Data | Expected Result | Actual Result | Pass/Fail |
| 1 | Host a game without any other players. | Entering the correct answers. | The game should play without any problem. | The game plays out without any problem. | Pass |
| 2 | Try to join a game with incorrect ip. | An incorrect ip to enter in join menu. | The game should not join any lobby because there is not any. | The game does not join any lobby and gives error. | Pass |
| 3 | Host and play a game with 4 players. | Correct ip to enter in the join menu. | The game should play without any problem. | The game plays out without any kind of problem. | Pass |
| 4 | Try to join a game as a 5th player. | Correct ip to enter in the join menu. | The game should not let the 5th player to join. | The 5th player experiences error and cannot join the game, while others do not experience anything about it. | Pass |

**Screenshot of 4 players playing the game at the same time:**

A picture containing text, scoreboard

Description automatically generated

**Game Engine Main Cycle Flowchart:**

Diagram

Description automatically generated

**Game Engine Main Cycle Algorithm:**

1. Start
2. Initialize the Display Object
3. Initialize the cmd keyboard listener class
4. Initialize main menu scene and its rerouting functions.
5. Set i to 0
6. Set frameLimit to 10
7. While True
8. Start Stopwatch timer
9. Clear the buffer in the display object
10. Call update method of every on the screen
11. If i >= frameLimit
12. Set i as 0
13. Call secondUpdate method of every scene.
14. Increase i value by 1
15. Update the buffer with the data from everyscene
16. Print out the buffer onto the screen
17. Clear keyboard history
18. Continue from the infinite while loop, which is number 7 outside of this loop, after 16 – timer.elapsedMilliseconds milliseconds.

**Network Algorithm:**

**Server:**

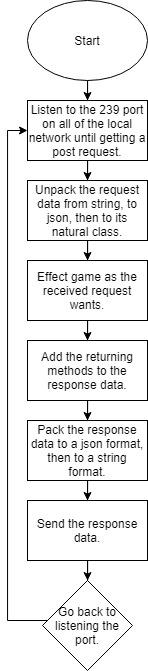
1. Listens to every kind of socket interaction in the 239th port in network.
2. Upon getting a post request, tries to unpack the data first as a string, then a json, then converts it into the supposed class based on the className value in the json.
3. Does the required stuff the class is supposed to do.
4. Returns a response, which includes host’s instructions.

**Client:**

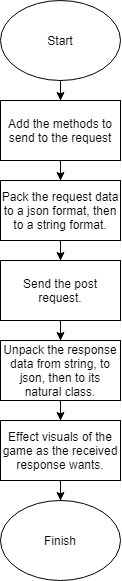
1. Sends data in a post request to the server the same way it is expected to be unpacked whenever required. Client also sends post requests every second, we call them “sanity check”. Which, even if it is empty, it allows server to send its own instructions in the response.
2. Gets the response of the host, unpacks the response the same way host unpacks client’s requests.
3. Does the required stuff the class is supposed to do the same way host does.

**Network Flowchart:**

**Server:**

****

**Client:**

****

**Encryption :**

**Screenshot of both code and terminal :**

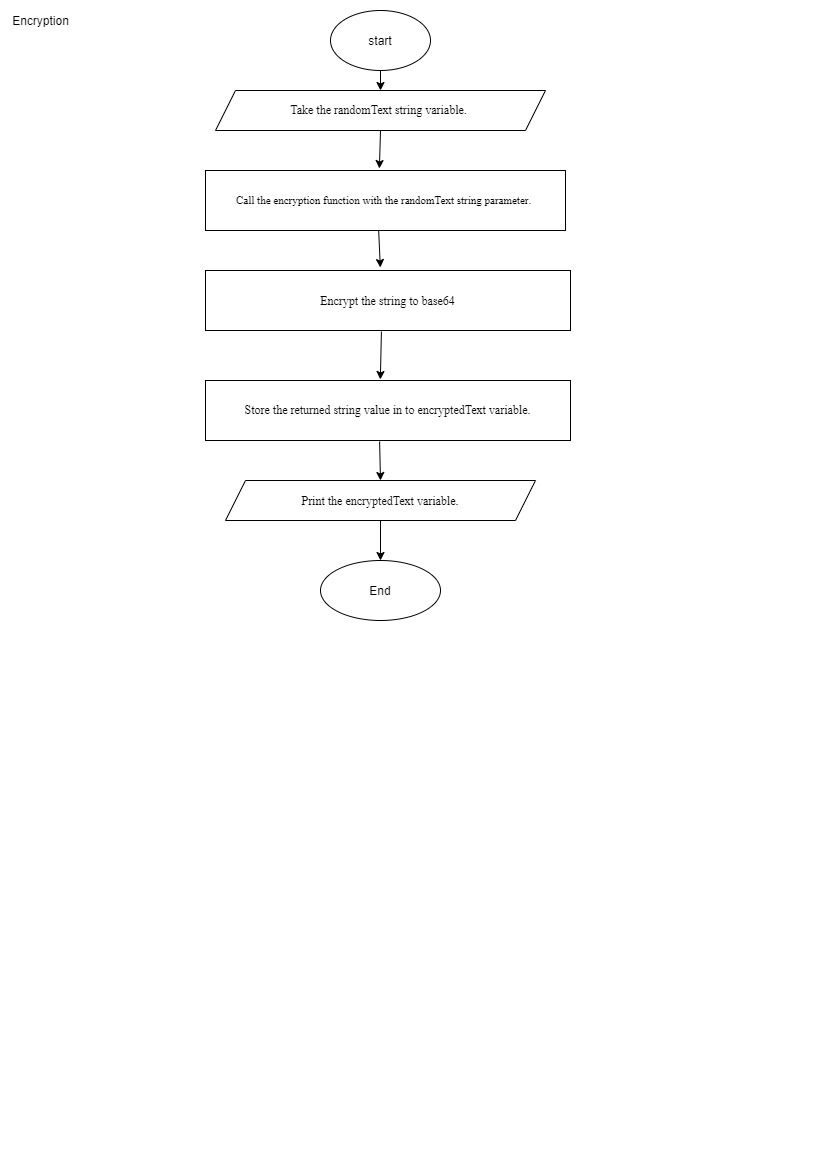
A screenshot of a computer

Description automatically generated

**Algorithm :**

* Take the randomText string variable.
* Call the encryption function with the randomText string parameter.
* Encrypt the string to base64
* Store the returned string value in to encryptedText variable.
* Print the encryptedText variable.

**Flowchart :**

****

**Test Cases :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Test Data** | **Expected**  **Results** | **Actual**  **Results** | **Pass/Fail** |
| **TC01** | Ability to enter any kind of string | 1-Run the the code.      2-Enter a random string, with characters and numbers, letters | Entering random strings- | Program takes the string to encrypt it | As Expected | Pass |
| **TC02** | Check if the program turns the string intered into encrypted.base | 1- Run the code      2—Enter my string | chaimae | Program  Should turn that string into an encrypted base | As Expected | Pass |
| **TC03** | Check if the code will output the encrypted string | 1- Run the program      2-Enter my string  3- Check if the encrypted output is there. | chaimae | Program  Should encrypt the string entred and show an encrypted output | As Expected  v/2hEc8h7ICkQasdm40pjg== | Pass |

**Decryption :**

**Screenshot of both code and Console:**

A screenshot of a computer

Description automatically generated

**Algorithm :**

* Take the 64 base encrypted code in to randomText variable
* Call the decryption function with randomText string parameter
* Decrypt the 64 base encrypted code in to string txt
* Return decryptedText to the main.
* Store the returned string value in to decryptedText variable.
* print the decryptedText

**Flowchart :**

**Diagram

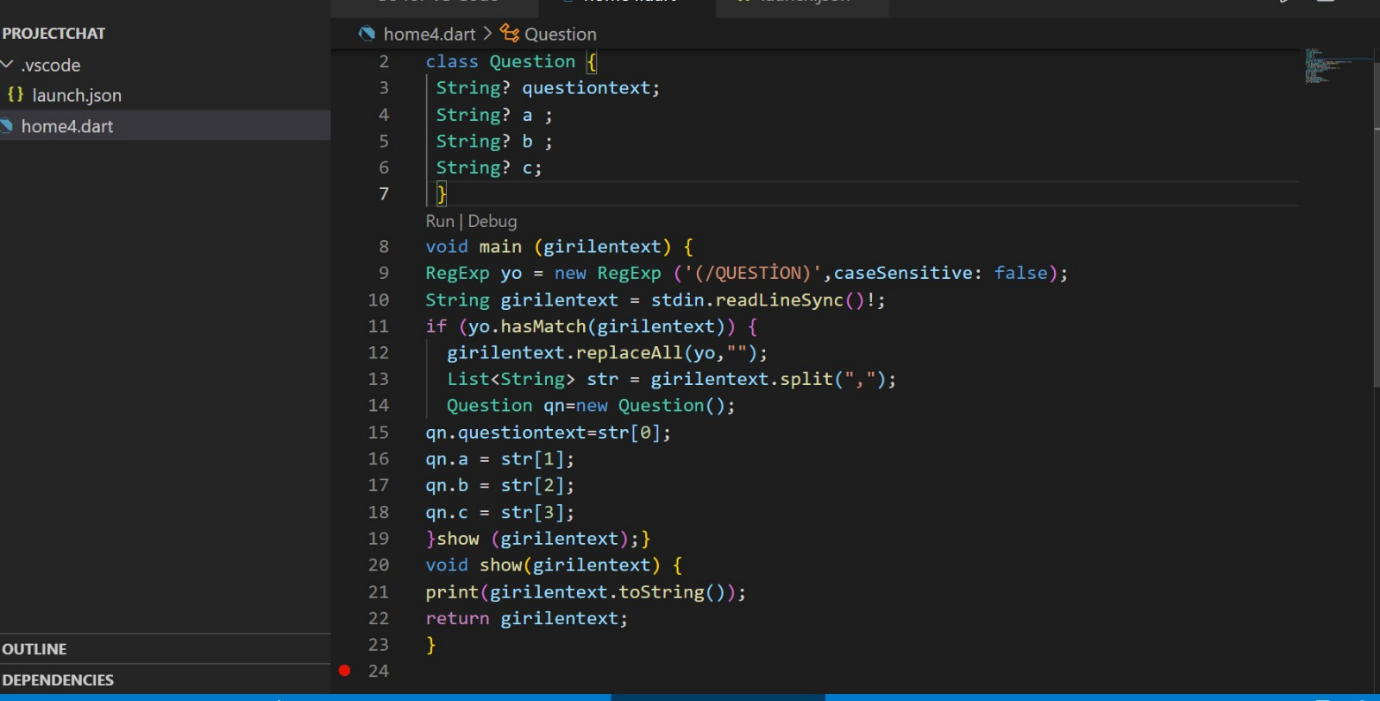
Description automatically generated**

**Test Cases :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Test Data** | **Expected**  **Results** | **Actual**  **Results** | **Pass/Fail** |
| **TC01** | Ability to enter any kind of encrypted string | 1-Run the program      2-check if code accepts any kind of encrypted string | Enters random  Encrypted  strings | Code should accept them all |  | Pass |
| **TC02** | Check if program turns the encrypted string into a normal string | 1- Run the program      2—Enter the encrypted string | lbWsF9QlqfDCPaQSlIImgQ== | Program  Should  turn the encrypted base into a string | As Expected | Pass |

**Regular Expressions:**

1. Start
2. Set classes for progress
3. By using RegExp method do whatever processing you want depending on MatchedRegEx indicates which regex matched
4. Regex matching a string in RegExList
5. Inside the loop you can then process the match according to whether it is a comment or a string
6. Set the call function to driver classes

****

**Results and Conclusions :**

**R\*\*a A\*\*\*\*i:** During this project, I worked on "request express" as a developer and it helped me get information about this method as a result of my research. The group conversations and meetings we had as a group and the work between the codes gave me a lot of experience. I personally believe our group is working hard on this project.

**C\*\*\*\*\*e Y\*\*\*\*i:** This project made me realize the practical steps the developer goes by, It helped me see how important it is to be very specific about the kind of the requirements the project needs, because being specific about it means being specific about what I would learn in order to achieve the desired goal, I had to learn how to encrypt strings and use another decryption code to decrypt it in a different application, I had to learn how to install new libraries and use the new syntax in it in order to make it usable. It was challenging at first, because it’s a new experience for me, but with the help of my team members and their encouragements, I succeeded and I made sure to make it work and take valuable lessons from it that would serve me in my future projects.

**Burhan Sancaklı:** The project made me acknowledge the fact that teamwork is much different than solo work. Working on just 1 aspect of the whole project, without knowing how the other code is going to be working, makes me nervous, but I realized that this is very unnecessary. While working with a team, everyone tend to code much more clear and compact, so that their friend can easily use it. This has greatly helped to make progress faster while making the game.

**M\*\*\*\*\*\*d F\*\*\*\*n Ö\*\*\*l:** In this project, I have prepared the question and answer part (algorithms). This project added to my coding experiences many things. I learned how to code with team and code to my team. This I think is very valuable and will allow me to better get used to future and work.

**Y\*\*\*s E\*\*e K\*\*a:** While preparing for this project, we asked ourselves what could be a good project and a good way to improve ourselves. I picked sockets part in this project because I wanted to learn more about is structure. In making I have learned a great deal about “server-client” relationship and logic of online games. In the end I can say our choice of project helped me improve myself a lot.

**References**

https://www.guru99.com/test-case.html