MTCG – Protocol

1. Design Description

To manage incoming requests, the server employs a solitary TCP socket, which serves as the entry point for all requests. By utilizing this socket, the server can distribute the workload among distinct threads, each dedicated to handling a specific request. Consequently, the server can process numerous requests concurrently, therefore enhancing its efficiency and response. This approach of employing a single TCP socket and assigning a thread to each request enables the server to effectively manage a substantial volume of requests simultaneously, while also ensuring prompt and efficient handling of individual requests.

The server's design incorporates a single TCP socket, which offers numerous advantages. Firstly, it enables seamless scalability as additional resources can be incorporated to accommodate a growing number of requests. Furthermore, this design streamlines server management by channelling all incoming requests through a central entry point, facilitating monitoring and troubleshooting processes. To summarize, the server efficiently manages incoming requests by utilizing a solitary TCP socket as a gateway. Each request is then assigned to a separate thread, enabling the server to handle multiple requests concurrently. This design promotes easy scalability and simplified server management.

2. Lessons Learned

- Concurrency with threads: The importance of utilizing threads to handle multiple requests simultaneously was learned. This allows for increased efficiency and responsiveness in the server.

- TCP sockets in a scalable web app: The use of a single TCP socket as a gateway for all incoming requests to the server was implemented. This allows for easy scaling of the server as more resources can be added to handle an increasing number of requests.

- Secure database connection: The importance of securing the connection to the database was learned. This includes implementing secure authentication methods and properly handling sensitive information.

- User authorization through token: The use of tokens for user authorization was implemented. This allows for secure and efficient management of user access to the system.

- General good practices: The use of design patterns, such as the singleton pattern, was implemented to improve the overall design and maintainability of the code.

3. Unit Testing

The unit tests are conducted in the “test”-folder and are working without any issues. A total of 20 tests are being performed which includes CombatTest, DamageCalculationTest, DeckTest, ResponseHandlerTest and UnwrapperTest.

4. Estimated Tracked Time

Approximately 50-55 hours were spent working on the whole project. This includes:

- Configuring the database and establishing a connection to the program.
- Configuring the server and dividing the workload into separate threads.
- Developing each class inside the manager folder.
- Developing each class in the project folder.
- Writing and running tests for the project.

5. GitHub Link

https://github.com/Enes3854/MonsterTradingCardGame.git