MTCG Protocol

1) **Design:** The MTCG is divided into 3 different projects. MTCG Server, MTCG Battle, MTCG Battle Test.

- **MTCG Server:** The server part is implemented just to handle client request and perform action accordingly. First of all, there is class **HttpClientWatcher**, which is implemented just to accept client requests, once a request comes to server then **HttpClientWatcher** accepts it the transfer it to **HttpClientHandler**. **HttpClientHandler** is implemented to handle requests. **HttpClientHandler** contains a **HandlerManager**, who is responsible for everything. HandlerManger has **HttpRequestHandler**, **HttpResponseHandler** and **NetworkStreamWriter**. **HttpRequestHandler** has a **NetworkStreamReader**, which is implemented to read request from stream and if data is available then then the data is transferred to **HttpResponseHandler**. This class is implemented to handle response. First of all, it is check that, whether the request route is correct or not, if its correct then, by the help Callable delegate the request will transferred to **Controller**. **Controller** is implemented, in order to perform request action accordingly. For example, get user, create package etc. Once the data is inserted or read then a response is sent to **NetworkStreamWriter**, which is responsible just to write response on stream.
- **MTCG Battle:** The battle part is implemented where actually battle is happened. There is a main class Battle where two players come and do battle. And the reason behind the extra project of battle, is to make the implementation of unit test easy.
- **MTCG Battle Test:** The battle test part is implemented, where all test cases are written.

2) **Learned Lessons:** As I invested many hours to build this project. I did learn many things.
   - LINQ
   - Threading
   - Unit test / Mocking
   - PostgreSQL
   - And how to create our own http server.

3) **Unit Test:** Total 21 Unit test are implemented to test battle part.
   - **CollisionDetectorBattleOnCollisionDetected_Test():** This test is very important. This test is written to check, whether battle is working fine or not.
   - **CalculateDrawCalculateTotalDrawSteak_Test():** In order to check, whether the function is calculating correct draws.
   - **IsCollied_Test():** In order to check whether the function is returning true or not, when two cards have same position (x and y axis)
   - Some test cases for card class are written, in order to check card name, direction etc.
   - Some test cases for player class are written, in order to check player functionalities. For example, is player deck empty, has player selected a random card, can card be added in his deck etc. All the player functionalities have been tested, in order to avoid any problem during battle.
   - Some test cases for round class are written. For example, test battle result where spell and monster fight, spell and spell fight or monster and monster fight. And one more test case is written, which calculates damage. All test cases were necessary to write, the battle can be played without any problem

4) **Unique Feature:** The unique feather of my program is that the cards during the battle can move and when both cards have same x and y axis then the damage will be calculated.

5)Total Time: Almost 60 hours
6)Git Link: