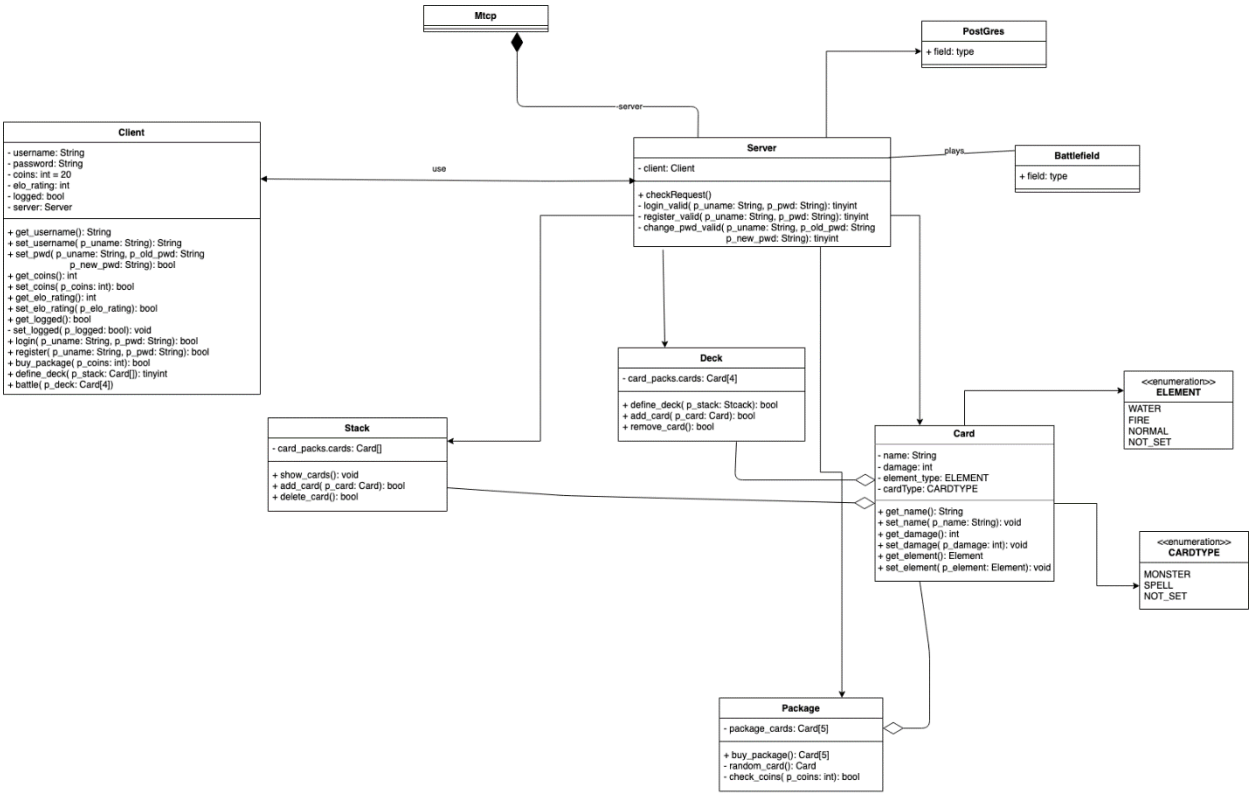
MTCG Dokumentation

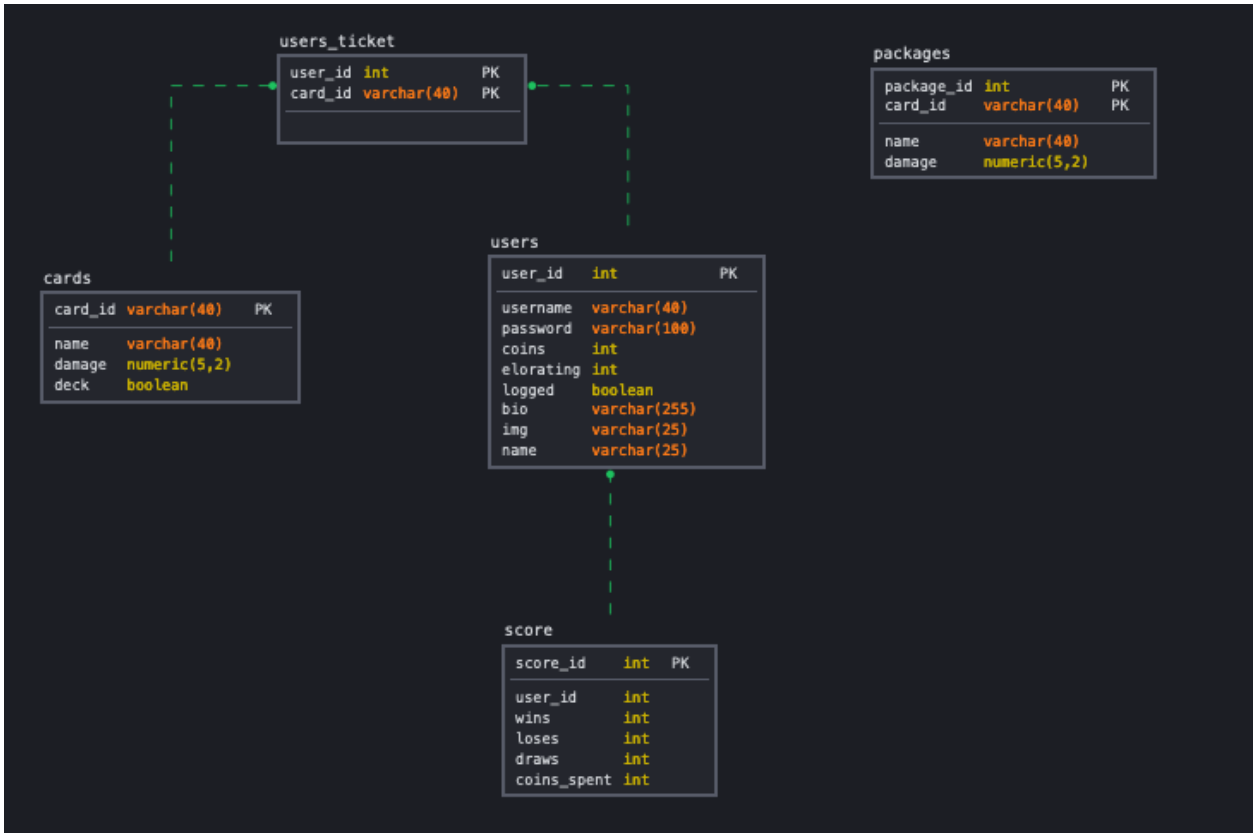
- **Designs:**
  - UML:



  - Der Projekt wurde auf 3 große Packages geteilt:
    - **Card_packs**
    - **Client**
    - **Server**
  - **Card_packs** package:
    - **Card** package:
      - **Card** Class: Hier wurde definiert wie die Karten sein werden und was für Attribute die bekommen
      - **CARDTYPE** Enum: Karten Typen
      - **ELEMENT** Enum: Karten Elemente
    - **Deck** Class: Hier definieren wir 4 Karten die der Deck erhalten wird. Hier überprüfen wir ob der Benutzer ein Deck hat und welche Karten drinnen sind
    - **Package** Class: Hier kaufen wir die erste Karten vom Server und wir bekommen zurück 4 verschiedene Packages die jede 5 Karten enthält

- **Stack** Class: Hier geben wir alle Karten die der Benutzer hat innerhalb eine Liste. Leider gibt es keine Trade Funktion in der Projekt und die Karten werden nicht ändern

- **Client** Package:
  - **Client** Class: Hier wird der Benutzer definiert mit seine eigene eigenschaften sowie einloggen Daten und Spiel Attribute. Die Methoden wo man die Benutzer Daten sehen will sind hier definiert
- **Server** Package:
  - **Battlefield** Class: Hier wird die Logik von der Spiel gemacht wo 2 Benutzer gegen einander Kämpfen. Jeder hat seine eigene 4 Karten die Random erstellt wurden und dann überprüfen wir wer gewonnen hat und wir zeigen es
  - **PostGre** Class: Hier ist alles was mit der Datenbank zu tun hat drinnen, Wir erstellen zuerst eine Connection mit unsere Datenbank im Constructor und dann erstellen wir die Methoden die von der Datenbank rufen und in der Datenbank ändern (select und insert into)



        - **Server** Class: Die wichtigste und die schwerste Klasse zum entwickeln. Es wurde REST verwenden um die verbindung mit dem Client zu erstellen und die Klasse wurde in basis von die Curl requests entwickelt.
        - **Verb** Enum: Die REST Methoden die wir für die Server Klasse verwenden (GET, POST, PUT, DELETE)
    - **Main:**
        - **Mtcp** Class: Auf die main Methode gibt es einen Loop das für jedes neues Client ein neues Socket erstellt wird. Dann werden die Client requests bearbeitet

- **Failures and selected Solutions:**
    - Es gab genug Faliures und Correcturen sowie die Battle Logik zu erstellen, obwohl es teoretisch nicht kompliziert ausschaut, man muss ein bisschen daran denken und es hat ein bisschen Zeit gedauert. Ich habe einen Weg gesucht wo man nicht viele if elses braucht um die Karten Stärte mit ein ander zu vergleichen aber leider ohne Erfolg. Ich wurde nächstes Mal früher mit die Datenbank entwicklung anfangen da es hat genung in der Projekt geändert wenn ich es erstellt habe. Mit REST gab es auch ein bisschen Probleme da ich habe immer eine Leere Message bekommen statt die richtige request wenn ich den Message teilen wollte. Es war kompliziert weil man musste auf Zeichen wie “{” “[” und “\” aufpassen. Auch wenn man die Request liest und was zu tun soll, habe ich mit case switch gemacht da es was nicht so klar wie es anders gemacht werden kann, obwohl in der Unterricht wurde emfehlt mit Case switch nicht zu arbeiten. In den Projekt wurden es auch Arrays statt Collections verwendet da es war auch leichter zum entwickeln. Lombok wurde auch nicht verwendet als Plugin weil ich habe es ohne Lombok angefangen und wollte kein extra Zeit dazu verwenden um das zu implementieren.

- **Unit tests:**
    - Es gibt insgesamt 18 Unit tests. Ich bin nicht sicher ob man eine Test die der Datenbank leer macht, als Unit test bezeichnen kann. Die Tests sind nicht mit Code Coverage getestet. Es gibt eine Test für fast jede wichtige Methode die mit die Cards und die Benutzer zu tun hat.

- **Time Tracking:**
    - Die Zeit wurde nicht genau gemesst aber ich wurde rechnen rund 60 Stunden
- **Github Link:**
    - https://github.com/ShkembiAnis/MTCG.git