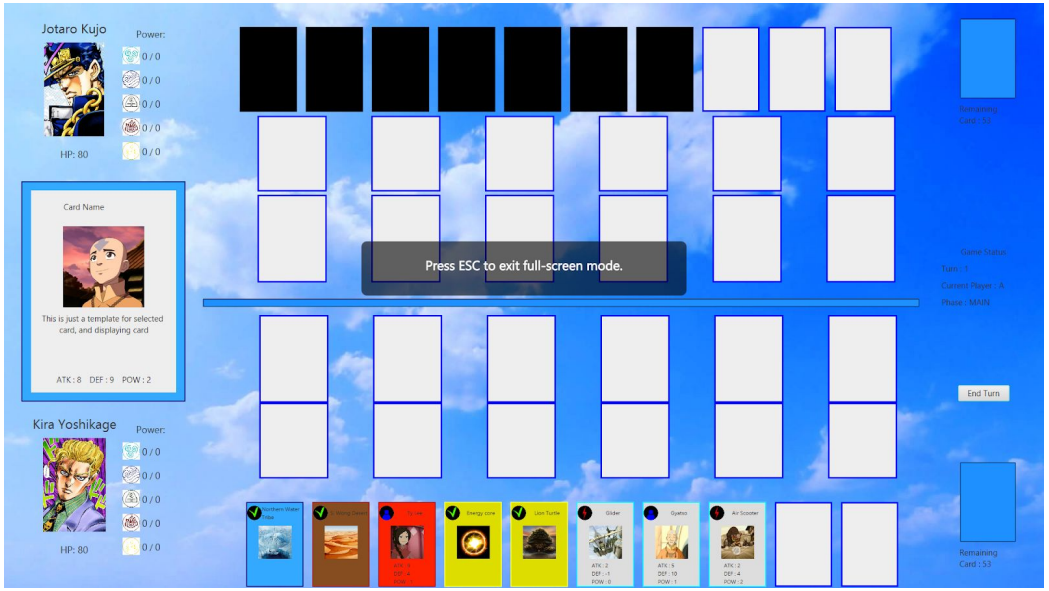
Kelas : 03
Kelompok : 10
1. 13518090 / Arthur Edgar Yunanto
2. 13518108 / Vincent Hasiholan
3. 13518111 / Muhamad Mirza Fathan Al Arsyad
4. 13518114 / Mario Gunawan
5. 13518138 / William
Asisten Pembimbing : Ahmad Fahmi Pratama

## 1. Deskripsi Umum Aplikasi


Aplikasi yang kami buat pada tugas besar 2 IF2210 kali ini merupakan suatu aplikasi permainan kartu berbasis *turn-based*. Tema dari aplikasi ini sendiri bertemakan game kartu layaknya permainan Yu Gi Oh, berjudul Avatar Duel yang dapat dimainkan oleh 2 (dua) orang. Dalam game ini, 2 pemain saling bertukar turn dalam memainkan kartu - kartu yang dimilikinya. Di awal permainan, seorang pemain akan melakukan penarikan kartu (draw) dari decknya masing - masing sebanyak 7 kartu. Untuk decknya sendiri, deck pemain akan terdiri dari kartu secara acak yang dipilih dari database kartu dalam game. Komposisi susunan kartu yang dimiliki deck pemain memiliki rasio 2 (kartu karakter) : 2 (kartu land) : 1 (kartu skill). Pemain dapat melakukan beberapa aksi dalam turn. Aksi yang dapat dilakukan tersebut berbeda tiap phase-nya. Dalam 1 (satu) turn, pemain memiliki 5 (lima) phase yang berbeda, yaitu draw phase, main phase 1, battle phase, main phase 2 dan end phase. Permainan berakhir ketika salah satu health point pemain yang bermain dalam Avatar Duel menyentuh angka 0.

Saat suatu game sedang berlangsung, terdapat beberapa hal yang dapat dilakukan oleh player bergantung terhadap phase turn tersebut. Tombol - tombol aksi yang ditampilkan pada player berubah seiring phase berjalan. Player dapat melihat deskripsi karakter dengan



meng-hover kartu yang ingin dilihat deskripsinya. Selain itu, kartu pada tangan player dapat dimainkan dengan cara menghover kartu di tangan dan menekan tombol “Play”. Untuk melakukan aksi di field sendiri, player dapat meng-hover kartu di field dan tombol aksi akan dapat ditekan user (baik aksi menyerang, berubah posisi, dan lain - lain).

# 2. Konsep OOP

## 2.1. Inheritance

Konsep Inheritance yang digunakan dalam aplikasi Avatar Duel dalam aplikasi ini :

- Kelas CharacterCard (src/main/java/com.avatarduel.model.card.CharacterCard.java) merupakan anak dari kelas abstrak Card (src/main/java/com.avatarduel.model.card.CharacterCard.java)

- Kelas SkillCard (src/main/java/com.avatarduel.model.card.SkillCard.java) merupakan anak dari kelas abstrak Card (src/main/java/com.avatarduel.model.card.CharacterCard.java)
- Kelas SkillAuraCard (src/main/java/com.avatarduel.model.card.SkillAuraCard.java) merupakan anak dari kelas abstrak SkillCard(src/main/java/com.avatarduel.model.card.SkillCard.java)
- Kelas LandCard(src/main/java/com.avatarduel.model.card.LandCard.java) merupakan anak dari kelas abstrak Card.

## 2.2. Composition

Konsep Composition yang digunakan dalam aplikasi Avatar Duel dalam aplikasi ini :

- Kelas Game (src/main/java/com.avatarduel) memiliki 2 (dua) kelas Player (src/main/java/com.avatarduel.model.player_component.Player.java), tepat 1 (satu) static kelas instance Game.
- Kelas Player (src/main/java/com.avatarduel.model.player_component.Player.java) memiliki tepat 1 (satu) kelas Deck (src/main/..), tepat 1 (satu) kelas Field, tepat 1 (satu) kelas Hand, dan tepat 1 (satu) kelas PowerManager

## 2.3. Interface

Daftar penggunaan konsep interface di dalam aplikasi Avatar Duel sebagai berikut :

- Interface IEvent
- Interface IField
- Interface IPhase
- Interface ExceptionCause
- Interface Request

## 2.4. Method Overriding

Daftar penggunaan konsep Polymorphism dalam aplikasi Avatar Duel sebagai berikut :

- Method setCard(Card card) pada kelas CardInHandController (src/main/java/com.avatarduel.guicontroller.Card.CardInHandController.java)
- Method createClone pada kelas CardFactory, dimana method ini dapat di-override untuk 5 jenis kelas yang berbeda, yaitu CharacterCard. SkillAuraCard, LandCard, SkillDestroyCard dan SkillPowerUpCard
- Method getMessage() dan getOperation() pada kelas InvalidOperationException.java (src/main/java/com.avatarduel.exception.InvalidOperationException.java)

## 2.5. Polymorphism

Daftar penggunaan konsep Polymorphism dalam aplikasi Avatar Duel sebagai berikut :

- Method dari executeEvent pada kelas BoardController (src/main/java/com/avatarduel/guicontroller/Board/BoardController.java) menerima interface IEvent. Hal ini membuat method ini dapat menerima setiap jenis dari Event yang ada pada game dan dapat mengeksekusi event tersebut.
- Method dari executeEvent pada kelas BoardController (src/main/java/com/avatarduel/guicontroller/Board/BoardController.java) juga menerima exception dengan kelas InvalidOperationException (src/main/java/com/avatarduel/exception/InvalidOperationException.java). Hal ini membuat setiap jenis exception / error yang memiliki parent class InvalidOperationException yang terjadi pada game dapat diterima / di-catch oleh block ini.

## 2.6. Java API

Daftar penggunaan konsep Java API dalam aplikasi Avatar Duel sebagai berikut :

- Kelas dalam package dao menggunakan API:
    - CardDAO : java.util.list untuk mendapatkan list of cards
    - CSVCardDAO: java.util.list, arraylist, dan collectors untuk memanage list of characters
- Kelas dalam package event menggunakan API:
    - ActivateDestroyEvent: java.util.list dan stream.collectors, keduanya untuk memanipulasi list of cards dari player A dan B yang di destroy
    - AttackEvent : menggunakan API yang sama dengan activate destroy event dengan tujuan yang sama

    - RemoveSkillCardEvent: menggunakan API yang sama dengan activate destroy event dengan tujuan memanipulasi list of skill card in field dari suatu player
- Kelas dalam package guicontroller.board menggunakan API :
    - BoardController : java.io.File dan java.io.URL untuk memainkan musik
    - FieldController : java.util.HashMap, List, dan Map untuk menyimpan berturut turut : charactercards dan skillcards dalam field, menyimpan list charactercard dan skillcard in field yang didapat dari backend, dan tipe charactercards dan skillcards
    - HandController : java.util arraylist dan list, keduanya digunakan untuk menyimpan data kartu di tangan
    - PlayerStatusController : java.io.File untuk mendapatkan file image untuk character image
- Kelas dalam package guicontroller.card menggunakan API:
    - CardController : java.io.File untuk mendapatkan image file
    - CardInHandController : java.util.List dan stream.collectors untuk mendapatkan list of characters in field yang digunakan saat tombol play diklik agar summon dapat dilakukan di indeks terkecil
- Kelas dalam package guicontroller.MainMenu menggunakan API:
    - CardLibraryController : java.util.List untuk menyimpan list kartu dalam database
    - HowToPlayController : java.util.File dan java.io.IOException untuk menyimpan file FXML dan untuk menangkap IOexception bila file tidak ada
    - MainMenuController : java.io.File dan IOException, dan java.net.URL, ketiganya digunakan untuk mendapatkan file controller dan memanipulasinya
- Kelas dalam package guicontroller.popup menggunakan API:
    - AttackPopupLoader : java.util.List untuk menyimpan list of enemy characters yang bisa diserang
    - PlayAuraOrPowerupCardLoader: java.util.list, stream.collectors, dan stream.Stream, yang ketiganya digunakan untuk menyimpan characters dari kedua field yang bisa diselect saat memainkan skill card
    - PlayDestroyCardLoader : java.util.list, alasannya sama dengan AttackPopupLoader
    - PopupLoader : java.io.IOException digunakan untuk menangkap IOException apabila file fxml yang di load (Actionform.fxml ) tidak ditemukan
- Kelas dalam package guicontroller.util menggunakan API:
    - FXMLHandler: java.io.File dan java.io.IOException, digunakan untuk load data dari file FXML, dan menangkap exception apabila file tidak ada
- Kelas dalam package model.player_component menggunakan API:

- Kelas Hand dan FIeld memanfaatkan bantuan Java API Collection yaitu List (bentuk implementasinya berupa ArrayList of Card, CharacterCardInField, dan SkillCardInField).
- Kelas Deck memanfaatkan bantuan Java API Collection yaitu Stack (bentuk implementasinya berupa Stack of Card) serta bantuan method dari Java API Collection berupa shuffle untuk melakukan pengacakan kartu dalam deck .
- Kelas Field memanfaatkan bantuan Java API stream filter, collect, findFirst dan orElse guna melakukan pencarian dan penghapusan kartu bagi kartu yang sudah tidak digunakan dalam field permainan lagi.
- Player : java.util.EmptyStackException digunakan untuk throw exception ketika deck sudah kosong

## 2.7. SOLID

### 2.7.1. Single Responsibility Principle

A. CharacterCardInFieldController

CharacterCardInFieldController memiliki responsibility untuk memanage aksi-aksi kartu character yang sedang dimainkan, yaitu merender aksi yang dilakukan pemain misalnya rotate, attack, dan summon character card dari tangan.

B. HandRenderRequest

Seperti namanya, kelas ini memiliki tanggung jawab untuk mengirimkan permintaan render untuk hand suatu player.

C. AttackEvent

AttackEvent memiliki tanggung jawab untuk manajemen penyerangan CharacterCardInField.

### 2.7.2. Open Closed Principle

A. Request, global request, specific requeust

Request tidak memiliki attribute type, melainkan global/specific request adalah sebuah abstract class yang bisa digunakan untuk mengirim permintaan render suatu komponen dari komponen yang lainnya. Contoh turunan render request adalah HandRenderRequest, GameStatusRenderRequest, dan lainnya. Cara membuat request baru cukup menambahkan implement specific request atau global request, namun tidak perlu mengubah request yang lainnya

B. InvalidOperationException

Cara menambahkan exception suatu operasi bisa dengan cara cukup menambahkan kelas yang extends invalid operation exception. Kelas kelas turunan invalid operation exception tidak perlu disentuh ketika menambahkan exception baru

C. SkillCard

Cara menambahkan skill card dengan menambahkan tipe kartu yang extend SkillCard, sehingga dapat dengan mudah menambahkan kelas, namun pemodifikasian dalam skillcard itu sendiri ditutup.

2.7.3. Liskov Substitution Principle

A. SkillCards

Semua skill yang mengimplementasikan SkillCard bisa menggunakan semua method yang diberikan SkillCard.

B. CardController

Semua controller card dapat menggunakan semua method yang didefinisikan CardController

C. Event

Interface Event memiliki satu method, execute yang dapat dipakai oleh semua member dari interface tersebut

2.7.4. Interface Segregation Principle

A. IEvent

Semua event menggunakan method execute dan validate

B. IPhase

Semua phase memiliki method next() dan getphase()

C. IField

CharacterCardInField dan SkillCardInField memiliki method yang ada pada IField ( getIndex, getCard, getCreatedAtTurn())

2.7.5. Dependency Inversion Principle

A. InvallidOperationException terhadap ExceptionCause

Invalid operation exception menerima argumen operation, dan exception cause yang merupakan sebuah interface, sehingga semua turunan invalid operation exception bisa menerima exception cause apapun

B. BoardController terhadap InvalidOperationException

BoardController menerima InvalidOperationException, yang merupakan abstract class, dan mengeluarkan alert sesuai anak dari invalid operation exception sehingga low level module dan high level module keduanya bergantung terhadap abstraksi

## 2.8. Design Pattern

Terdapat beberapa daftar penggunaan konsep design pattern di dalam aplikasi Avatar Duel, sebagai berikut :

- Singleton

Digunakan design pattern Singleton dengan Game (<path>) sebagai kelas Singleton agar hanya terdapat 1 (satu) instance game selama aplikasi Avatar Duel berjalan. Hal ini juga kami lakukan guna mempermudah proses inisiasi dan pengambilan data terkait state - state yang terjadi dalam permainan selama aplikasi berlangsung.

- Factory

  Digunakan design pattern Factory dengan CardFactory (<path>) sebagai kelas factory dan digunakan untuk menghasilkan 5 jenis kelas yang berbeda, yaitu kelas CharacterCard, kelas LandCard, kelas SkillAuraCard, kelas SkillDestroyCard dan kelas SkillPowerUpCard. Selain itu, design pattern Factory ini juga pada CardInField (<path>) sebagai kelas factory untuk menghasilkan 2 jenis kelas yang berbeda, yaitu kelas CharacterCardInField dan SkillCardInField.

- Data Access Object (DAO)

  Digunakan design pattern data access object untuk melakukan pengaksesan data kartu yang disimpan untuk diproses dan digunakan dalam aplikasi. Dalam aplikasi kali ini, kami memanfaatkan interface CardDAO dan menghasilkan CSVCardDAO untuk melakukan pengaksesan data kartu dalam basis data berbentuk format file csv.

- Adapter

  Digunakan design pattern adapter untuk memberikan suatu interface antara kelas kartu biasa (CharacterCard dan SkillCard) dengan kelas kartu dalam permainan / field (CharacterCardInField dan SkillCardInField). Oleh karena itu, dibentuk suatu interface IField untuk memberikan suatu adapter sehingga kedua kelas tersebut dapat saling terhubung.

- State Design Pattern

  Digunakan design pattern state untuk melakukan perubahan tingkah laku aplikasi sesuai dengan state yang ada saat itu. Design pattern ini kami gunakan untuk menandai apa yang perlu dan dapat dilakukan saat phase tertentu dalam 1 (satu) turn seorang pemain. State yang tersedia terkait dengan phase ini adalah DrawPhase, MainPhase, BattlePhase dan EndPhase, yang dimana seluruh kelas tersebut mengimplementasikan interface yang sama yaitu IPhase

## 3. Pembagian Tugas

1. 13518114/ Mario Gunawan
   a. Membuat komponen - komponen tampilan aplikasi AvatarDuel (UI), yakni:
      - resource/com/avatarduel/card
      - resource/com/avatarduel/GUI

- resource/com/avatarduel/music

b. Mendesain dan membuat kelas kelas controller tampilan aplikasi AvatarDuel, yaitu:
   - Package guicontroller.Board
   - Package guicontroller.Card
   - Package guicontroller.MainMenu
   - Package guicontroller.Popup
   - Package guicontroller.Request
   - Package guicontroller.util
c. Mendesain dan membuat kelas exception yang digunakan dalam aplikasi AvatarDuel, yaitu
   - Package exception
d. Membuat dokumentasi untuk javadoc pada seluruh package guicontroller
e. Melakukan testing pada kelas exception
f. Membantu implementasi event bus dalam proses pengiriman event dalam aplikasi Avatar Duel

2. 13518138/ William Ong
   a. Merancang dan mengimplementasi seluruh event yang terjadi dalam game / aplikasi AvatarDuel, yaitu:
      - Package event
   b. Merancang dan mengimplementasi mekanisme bertarung dalam game / aplikasi AvatarDuel
   c. Merancang dan mengimplementasi design pattern singleton untuk membuat suatu Objek Game tunggal dalam aplikasi AvatarDuel. (package dao)
   d. Merancang dan mengimplementasi design pattern data access object untuk melakukan pengaksesan data aplikasi AvatarDuel dari database (csv)
   e. Merancang dan mengimplementasi design pattern state untuk pergantian phase dalam game AvatarDuel, yaitu pada:
      - Package phase
   f. Merancang dan mengimplementasi design pattern factory untuk menghasilkan kartu dalam game AvatarDuel, yaitu pada:
      - Package factory
   g. Merancang dan mengimplementasi seluruh objek model dalam game / aplikasi AvatarDuel, yaitu :
      - Package model.card

    - Package model.player_component
    - Package model.type
  h. Mengimplementasikan library bawaan guava sebagai EventBus untuk menangani event yang diterima dari user / pemain terhadap game.
  i. Membantu mengimplementasikan controller untuk tampilan aplikasi
  j. Melakukan dokumentasi pada package Event, Factory, util
  k. Melakukan *testing* pada test/java/com.avatarduel.dao dan avatarduel.model.player_component
3. 13518090/ Arthur Edgar
  a. Membantu membuat kelas exception dalam aplikasi AvatarDuel
  b. Membantu merancang package Card dan Phase
  c. Merancang dan mengimplementasikan kelas PowerManager
  d. Merancang dan membuat seluruh file CSV yang menjadi komponen data utama dalam aplikasi AvatarDuel, yaitu :
    - Land.csv
    - Character.csv
    - Skill_aura.csv
    - Skill_destroy.csv
    - skill_power_up.csv
  e. Membuat dokumentasi untuk seluruh komponen model dalam aplikasi AvatarDuel
  f. Membuat seluruh README.md
  g. Membuat dokementasi untuk komponen dao
  h. Melakukan testing pada test/java/com.avatarduel.phase
4. 13518108 / Vincent Hasiholan
  a. Tidak melakukan apapun
5. 13518111/ M Mirza Fathan Al Arsyad
  a. *Testing* beberapa method dalam *package* com.avatarduel.model