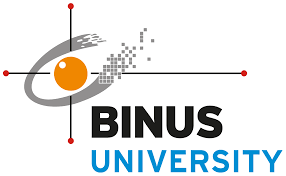
**PROGRAM PENCATATAN AGENDA SEPAKBOLA (PPAS)**

LAPORAN PROYEK AKHIR

MATA KULIAH COMP6364 – OBJECT ORIENTED PROGRAMMING

KELAS BA20



Oleh :

2301885900 – Daniel Alferian

2301879583 – Vincentius Sutanto

2301947376 – I Made Krishna Wirabuana

**DAFTAR ISI**

**HALAMAN JUDUL**

[**DAFTAR ISI**](#_heading=h.gjdgxs) **i**

[**DAFTAR TABEL**](#_heading=h.3dy6vkm) **ii**

[**DAFTAR GAMBAR**](#_heading=h.1t3h5sf) **iii**

[**BAB 1 PENDAHULUAN**](#_heading=h.2s8eyo1) **1**

[1.1](#_heading=h.17dp8vu) Latar Belakang 1

[1.2](#_heading=h.3rdcrjn) Alasan Pemilihan Topik 2

[1.3 Teori yang digu](#_heading=h.26in1rg)nakan 2

[**BAB 2 PEMBUATAN PROGRAM**](#_heading=h.44sinio) **6**

[2.1.](#_heading=h.2jxsxqh) Class Diagram 6

[2.2.](#_heading=h.z337ya) Use Case Diagram 7

[2.3.](#_heading=h.3j2qqm3) Class dan Modul yang digunakan 7

2.4. Tampilan Layar 10

[**BAB 3 PEMBAGIAN TUGAS**](#_heading=h.3whwml4) **16**

[3.1](#_heading=h.2bn6wsx) Pembagian Tugas 16

[**LAMPIRAN**](#_heading=h.36ei31r)

Source Code 17

Referensi 42

**DAFTAR TABEL**

Tabel 1 – Pembagian Tugas 16

**DAFTAR GAMBAR**

Gambar 1 – Class Diagram 6

Gambar 2 – Use Case 7

Gambar 3 – Tampilan Awal 10

Gambar 4 – Main Menu 10

Gambar 5 – Menu Input Data 10

Gambar 6 – Input Data tim dan Pemain 11

Gambar 7 – Input Wasit 11

Gambar 8 – Input Wasit 12

Gambar 9 – Input Wasit 12

Gambar 10 – Input Pertandingan 13

Gambar 11 – Menu View Data 13

Gambar 12 – View Data Tim 14

Gambar 13 – View Data Pemain 14

Gambar 14 – View Data Wasit 14

Gambar 15 – View Data Pertandingan 15

Gambar 16 – Menampilkan Klasemen Sementara 15

Gambar 17 – Mevalidasi Penyimpanan 15

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Seiring berkembangnya jaman, teknologi semakin banyak digunakan oleh manusia. Bahkan, dapat dikatakan bahwa teknologi sudah menjadi kebutuhan bagi manusia pada seluruh sektor kehidupan. Hal ini dikarenakan dengan adanya teknologi, pekerjaan manusia akan menjadi lebih mudah. Sebagai contoh perkembangan teknologi adalah adanya berbagai kemudahan dalam mengakses hiburan yang diinginkan. Sudah tersedia banyak sekali teknologi yang membuat dunia hiburan dapat dilihat, disaksikan, dinikmati bahkan dibuat dimanapun mereka berada. Salah satu contoh hiburan tersebut adalah hiburan sepakbola, masyarakat kini tak hanya dapat menyaksikan sepakbola secara langsung, melainkan mereka juga dapat bermain sebagai pemain pada permainan sepak bola itu (soccer virtual) dalam berbagai *game*.

Sepakbola telah menempati posisi yang spesial di hati masyarakat Indonesia. Mulai dari pertandingan dalam negeri hingga internasional, banyak masyarakat yang sangat menikmati euforia pertandingan sepakbola. Dunia sepakbola sendiri merupakan dunia yang sangat luas dan mencakup banyak hal. Tak hanya permainan itu saja, sepakbola juga berkaitan dengan hubungan antar tim, wasit, pemain, penonton, dan masih banyak lagi.

Dalam suatu pertandingan sepakbola, perilisan pemain untuk masing-masing tim sangat penting. Hal ini bertujuan untuk memberitahukan penonton siapa saja pemain yang akan berpartisipasi dalam suatu pertandingan. Tentu saja, seluruh hal yang berkaitan dengan pertandingan ini perlu dicatat dengan lengkap, mulai dari tim, pemain, wasit, hingga pelanggaran yang terjadi. Pencatatan pertandingan sepakbola sangat berguna bagi banyak pihak, seperti bagi mereka yang tidak dapat menyaksikan pertandingan secara langsung, mereka yang melewatkan suatu pertandingan, atau bahkan bagi orang yang bertugas untuk menganalisa suatu pertandingan.

Maka dari itu, diperlukannya sebuah program yang dapat membantu dan memudahkan banyak pihak. Tak hanya menikmatinya, program ini harus dapat memudahkan dalam pengolahan data pertandingan sepakbola yang memiliki fitur yang cukup lengkap untuk pendataan pertandingan sepakbola. Pendataan tersebut harus mencakup pencatatan tim sepak bola, pemain, wasit, pelanggaran, hasil pertandingan, *standing* seluruh tim, hingga adanya fitur *save* agar data yang telah dicatat dapat tersimpan.

* 1. **Alasan Pemilihan Topik**

Sesuai dengan latar belakang diatas, kelompok kami memilih untuk mengambil topik ini adalah karena bagi kami, perlu adanya sebuah program yang tersusun dengan baik untuk memudahkan pendataan dalam dunia sepak bola. Selain itu, program tersebut harus mudah untuk dimengerti, digunakan, dan dapat bermanfaat. Nama dari program yang kami buat ini merupakan Program Pencatatan Agenda Sepakbola Indonesia (PPAS).

* 1. **Teori yang Digunakan**

Dalam membuat program ini, kami menggunakan beberapa teori yang telah dipelajari pada semester ke-4 ini yaitu mata kuliah Object Oriented Programming (OOP). Ada beberapa teori-teori OOP yang kami gunakan yakni seperti,

**Inheritance**

Inheritance adalah salah satu dari konsep dasar Object Oriented Programming (OOP). Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan. Dengan konsep inheritance, sebuah class dapat mempunyai class turunan. Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class [1].

Ini menyiratkan bahwa derivasi kelas (subclassing) adalah alat pemrograman penting dalam dirinya sendiri. Ini dapat digunakan untuk mendukung pemrograman berorientasi objek, tetapi memiliki kegunaan yang lebih luas. Hal ini terutama benar jika seseorang mengidentifikasi penggunaan pewarisan dalam pemrograman berorientasi objek dengan gagasan bahwa kelas dasar mengekspresikan konsep umum di mana semua kelas turunan adalah spesialisasi. Ide ini hanya menangkap sebagian dari kekuatan ekspresif pewarisan, tetapi sangat didorong oleh bahasa di mana setiap fungsi anggota adalah virtual (atau metode). Mengingat kontrol yang sesuai dari apa yang diwarisi, turunan kelas dapat menjadi alat yang ampuh untuk membuat tipe baru. Diberikan sebuah kelas, derivasi dapat digunakan untuk menambah dan/atau mengurangi fitur. Hubungan kelas yang dihasilkan dengan basisnya tidak selalu dapat sepenuhnya dijelaskan dalam hal spesialisasi. Turunan adalah alat lain di tangan seorang programmer dan tidak ada cara yang sangat mudah untuk memprediksi bagaimana itu akan digunakan – dan terlalu dini untuk mengetahui kegunaan mana yang akan digunakan.

Deklarasi Inheritance di dalam Java untuk mendeklarasikan suatu class sebagai subclass cukup mudah, yaitu dengan menambahkan kata kunci extends setelah deklarasi nama class, kemudian diikuti dengan nama parent class-nya. Kata kunci extends tersebut memberitahu kompiler Java bahwa kita ingin melakukan perluasan class. Berikut adalah contoh deklarasi inheritance : –

> class B adalah subclass dari class A

public class B extends A {

…

}

Semua class di dalam Java adalah merupakan subclass dari class super induk yang bernama

Object. Misalnya saja kita mempunyai sebuah class sederhana :

public class A{

….

}

Pada saat dikompilasi, Kompiler Java akan membacanya sebagai subclass dari class Object.

public class A extends Object{

….

}

**Array List**

ArrayList adalah Array yang bersifat dinamis sehingga permasalahan terhadap panjang array dapat diatasi. Maksudnya adalah arraylist dapat digunakan tanpa perlu tahu pasti jumlah array yang akan dibuat.

* ArrayList mewarisi kelas AbstractList dan mengimplementasikan antarmuka Daftar.
* ArrayList diinisialisasi dengan ukuran, namun ukurannya bisa bertambah jika koleksi bertambah atau menyusut jika objek dihapus dari koleksi.
* Java ArrayList memungkinkan kita untuk mengakses daftar secara random/acak.
* ArrayList tidak dapat digunakan untuk tipe primitif, seperti int, char,.
* ArrayList di Java dapat dilihat mirip dengan vektor di C++.

**Interface**

Interface adalah tipe referensi di Java, mirip dengan kelas, itu adalah kumpulan metode abstrak. Kelas mengimplementasikan Interface, sehingga mewarisi metode abstrak Interface. Seiring dengan metode abstrak, Interface juga dapat berisi konstanta, metode default, metode statis, dan tipe bersarang. Badan metode hanya ada untuk metode default dan metode statis. Menulis Interface mirip dengan menulis kelas. Tetapi kelas menggambarkan atribut dan perilaku suatu objek. Dan Interface berisi perilaku yang diimplementasikan oleh kelas. Kecuali kelas yang mengimplementasikan Interface abstrak, semua metode Interface perlu didefinisikan di kelas.

**Object dan Class**

Objek perangkat lunak juga memiliki status dan perilaku. Status objek perangkat lunak disimpan dalam bidang dan perilaku ditampilkan melalui metode. Jadi dalam pengembangan perangkat lunak, metode beroperasi pada keadaan internal suatu objek dan komunikasi objek-ke-objek dilakukan melalui metode. sedangkan Kelas dapat didefinisikan sebagai template yang menggambarkan perilaku/status objek yang didukung tipenya.

**Serialization**

Serialization adalah Proses mengubah struktur data atau status objek menjadi format yang dapat disimpan. Dengan menggunakan serialization bisa dicapai dengan menulis kode untuk mengubah objek kelas ke dalam format apa pun yang dapat kita simpan dalam perangkat keras dan hasilkan kembali nanti di lingkungan lain atau yang sama.

**Thread**

Thread memungkinkan satu program untuk menjalankan beberapa bagian dari dirinya sendiri secara bersamaan. Misalnya, satu bagian dari program dapat menampilkan animasi di layar sementara bagian lain membuat animasi berikutnya untuk ditampilkan. Thread tidak membuat salinan dari seluruh proses induk. Sebaliknya, hanya kode yang dibutuhkan yang dijalankan secara paralel. Ini berarti bahwa Thread dapat dimulai dengan cepat karena mereka tidak memiliki semua overhead dari proses yang lengkap. Namun, mereka memiliki akses lengkap ke semua data dari proses induk. Thread dapat membaca dan/atau menulis data yang dapat diakses oleh Thread lainnya.

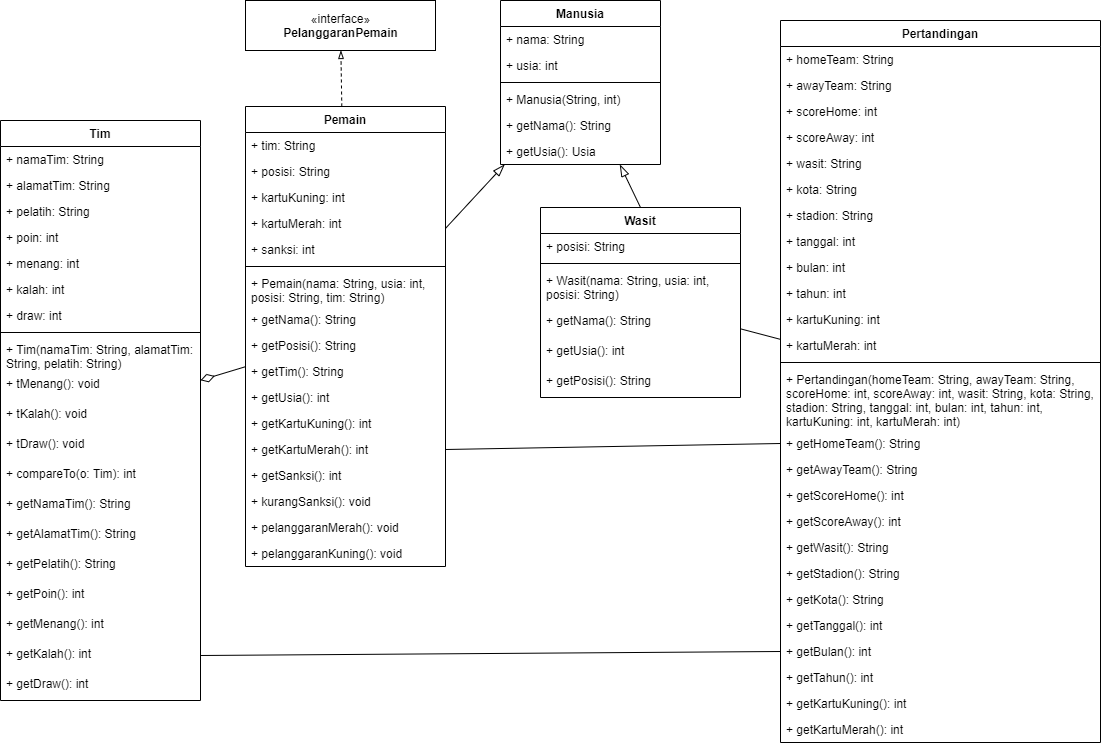
**Encapsulation**

Enkapsulasi adalah salah satu dari empat konsep dasar OOP. Tiga lainnya adalah pewarisan, polimorfisme, dan abstraksi. Enkapsulasi adalah mekanisme membungkus variabel data dan kode yang bekerja pada metode data bersama sebagai satu kesatuan [2]. Dalam enkapsulasi, variabel suatu kelas akan disembunyikan dari kelas lain, dan hanya dapat diakses melalui metode kelas mereka saat ini, oleh karena itu disebut juga sebagai penyembunyian data.

**BAB II**

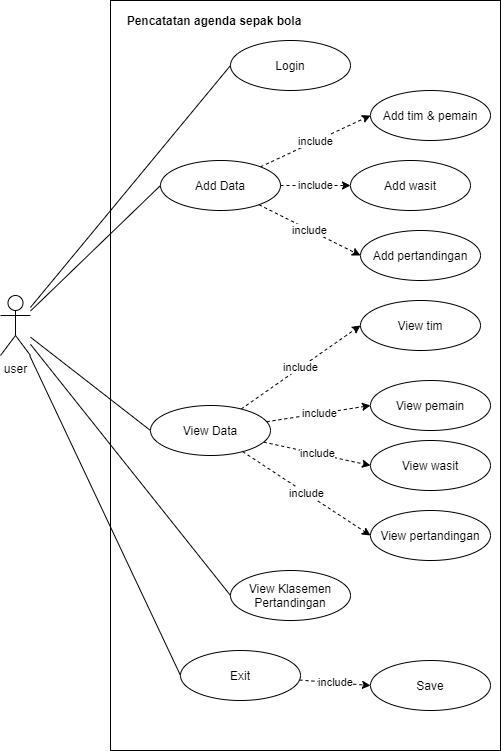
**PEMBUATAN PROGRAM**

**2.1 Class Diagram**



Gambar 1 – Class Diagram

**2.2 Use Case**



Gambar 2 – Use Case

**2.3 Class dan Modul**

**Class Main**

Class ini merupakan class dimana program akan dijalankan. Pada class ini dibuat arraylist Wasit, Pemain, Tim, Pertandingan untuk menyimpan data yang ada sebagai object yang selanjutnya akan disimpan pada akhir berjalannya pogram dan akan selalu di *load* pada awal program dijalankan.

Terdapat 4 menu yaitu menu input data, view data, view klasemen, dan exit atau save. Keempat menu tersebut dapat dipilih oleh user dan akan selalu berulang sampai user memilih exit atau save.

Menu input data dibagi lagi menjadi 3 pilihan, yaitu add tim & pemain, add wasit, dan add pertandingan. Untuk add tim dan pemain maka user akan diminta untuk memasukkan seluruh data dari tim dan pemain yang akan bertanding. Untuk add wasit maka user akan diminta untuk memasukkan seluruh data dari wasit yang akan bertanding. Sedangkan untuk pertandingan akan dapat diakses jika sudah terdapat minimal 2 tim dan 1 wasit. Setelah itu user akan diminta untuk memasukkan data yang dibutuhkan untuk pertandingan dan juga dapat memberikan informasi pemain yang sedang mengalami sanksi dan dapat menambahkan sanksi pada pemain.

Menu view data dibagi lagi menjadi 4 pilihan yaitu, view data tim, view data pemain, view data wasit, dan view data pertandingan. Masing masing dari pilihan tersebut akan menampilkan data masing masing yang diinginkan.

Menu view klasemen akan menampilkan posisi tim diurutkan berdasarkan poin terbesar sampai poin terkecil dari semua tim serta menampilkan jumlah kemenangan, kekalahan dan draw masing masing tim.

Menu exit dan save akan menanyakan user mau menyimpan data atau tidak, jika iya maka data akan disimpan serta program berhenti dan jika tidak maka program akan berhenti tanpa menyimpan data.

**Class Manusia**

Class ini merupakan super class dari class Wasit dan Pemain. Class ini memiliki atribut nama dan usia yang juga dimiliki oleh subclassnya serta memiliki getNama dan getUsia.

**Class Wasit**

Class ini digunakan untuk membuat object wasit. Class ini berisi seluruh data yang memungkinkan dimiliki oleh wasit serta merupakan subclass dari class Manusia. Memiliki constructor untuk memasukkan data dan juga berbagai getter untuk memanggil data.

**Class Pemain**

Class ini merupakan subclass dari class Manusia dan mengimplementasi interface PelanggaranPemain. Karena mengimplementasikan interface PelanggaranPemain maka class ini wajib memiliki method yang ada di interface PelanggaranPemain yaitu pelanggaranMerah dan pelanggaranKuning. Kedua method tersebut digunakan untuk menambahkan jumlah sanksi pada pemain jika terdapat pemain yang terkena pelanggaran kartu merah atau kartu kuning.

Method kurangSanksi pada class Pemain digunakan untuk mengurangi jumlah sanksi pada pemain ketika mereka telah melewati sebuat pertandingan. Pada class ini juga terdapat berbagai getter yang dibutuhkan untuk memanggil data pada class Pemain.

**Class Tim**

Class Tim mengimplement Comparable dan methodnya yaitu compareTo yang merupakan salah satu fitur dari library java yang berguna untuk mengurutkan arraylist object yang kami gunakan untuk melihat klasemen pertandingan.

Class ini juga memiliki method tMenang dan tkalah yang berguna untuk menghitung jumlah kemenangan atau kekalahan tim tersebut serta poin sementara tim. Class Tim juga memiliki berbagai getter untuk memanggil data pada class Pemain.

**Class Pertandingan**

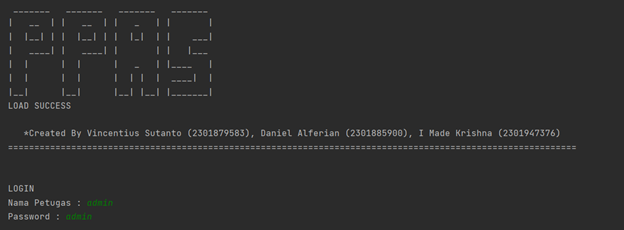
Class Pertandingan digunakan untuk menyimpan seluruh data pertandingan yang dibutuhkan. Seluruh data tersebut akan dimasukkan melalui constractor class Pertandingan. Selain itu class ini juga memiliki berbagai macam getter untuk memanggil data pada class Pertandingan

**Interface PelanggaranPemain**

Interface ini berisikan dua method yaitu method pelanggaranMerah dan pelanggaranKuning yang akan diimplementasikan pada class Pemain.

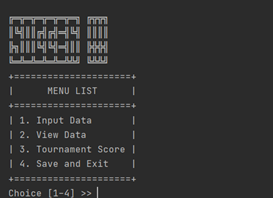
**2.4 Tampilan Layar**

Berikut merupakan tampilan dari program yang telah kami buat, seperti berikut



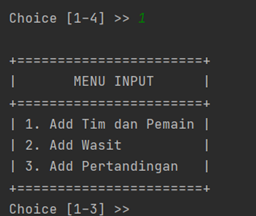
Gambar 3 – Tampilan Awal

Login awal program pencatatan agenda sepakbola



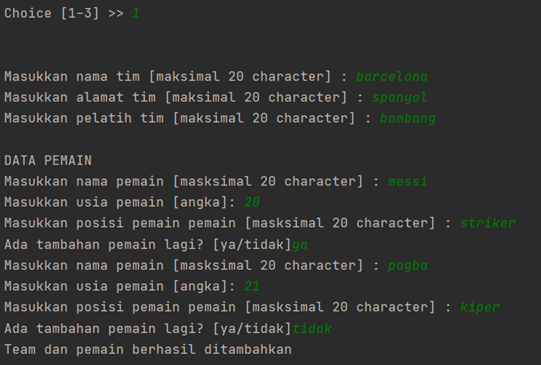
Gambar 4 – Main Menu

MAIN MENU dari program pencatatan agenda sepakbola



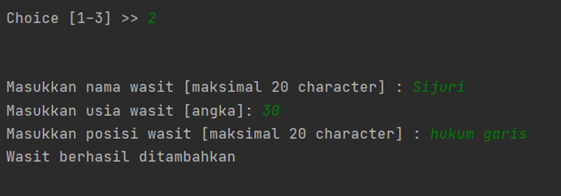
Gambar 5 – Menu Input Data

Menu 1 untuk menu input data, terdapat 3 pilihan yaitu add tim dan pemain, add wasit, dan add pertandingan.



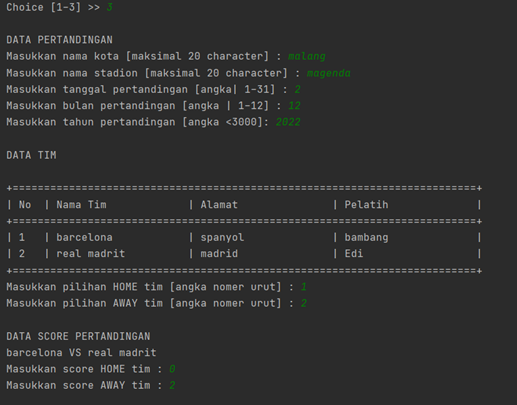
Gambar 6 – Input Data tim dan Pemain

Menu 1.1 add tim dan pemain. Pada menu ini user diminta memasukkan segala informasi tentang tim yang akan bertanding serta data pemain yang ada didalam tim tersebut.

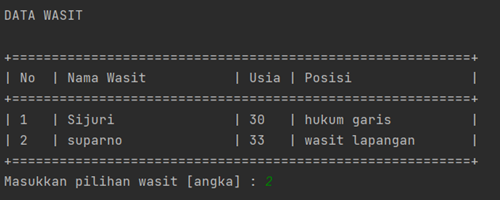


Gambar 7 – Input Wasit

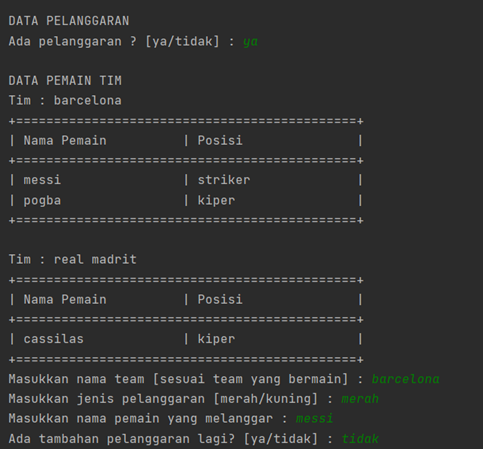
Menu 1.2 add wasit. Pada menu ini user diminta untuk memasukkan segala informasi tentang wasit yang akan menjadi menjadi penengah dalam pertandingan.



Gambar 8 – Input Wasit

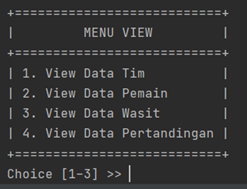


Gambar 9 – Input Wasit



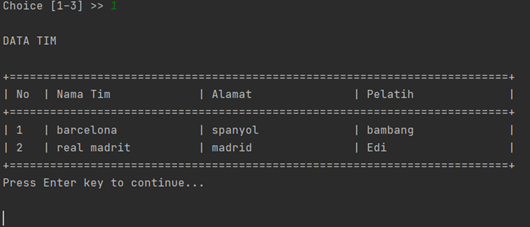
Gambar 10 – Input Pertandingan

Menu 1.3 add pertandingan. Pada menu ini user diminta untuk memasukkan tempat, stadion, dan tanngal pertandingan. Setelah itu user diminta untuk memilih tim yang bertanding dan memasukkan skor kedua tim. User diminta memilih wasit yang dalam pertandingan tersebut. User juga diminta untuk memasukkan data pelanggaran jika terjadi pelanggaran dalam pertandingan.



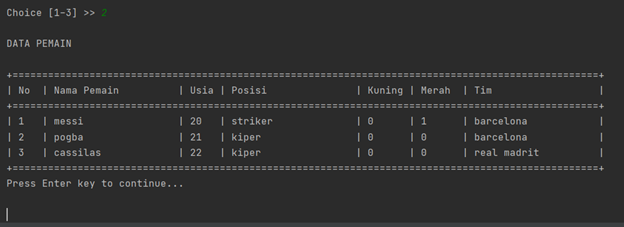
Gambar 11 – Menu View Data

Menu 2 untuk menu view data. Terdapat pilihan untuk view data tim, pemain, wasit, dan pertandingan.



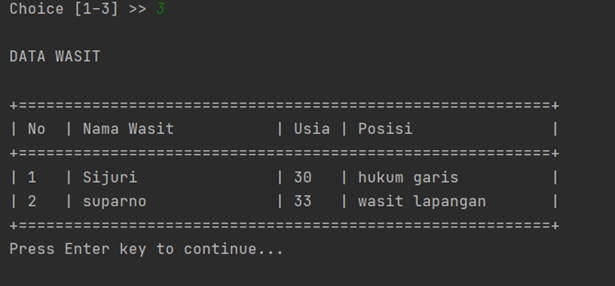
Gambar 12 – View Data Tim

Menu 2.1 view data tim akan menampilkan seluruh data tim yang tersedia.



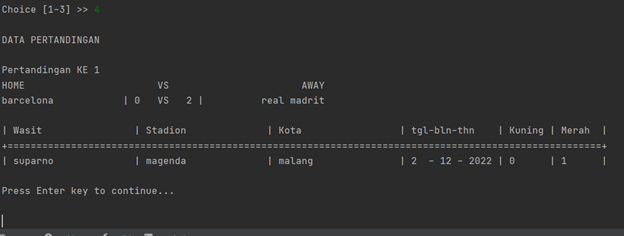
Gambar 13 – View Data Pemain

Menu 2.2 view data pemain akan menampilkan seluruh data pemain yang tersedia



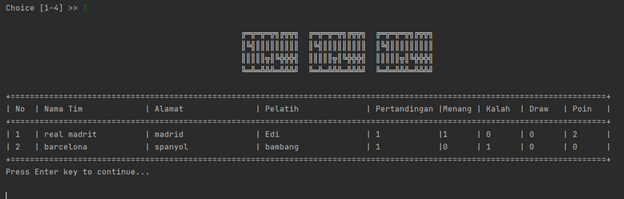
Gambar 14 – View Data Wasit

Menu 2.3 view data wasit akan menampilkan seluruh data wasit yang tersedia.



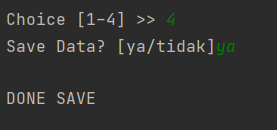
Gambar 15 – View Data Pertandingan

Menu 2.4 view data pertandingan akan menampilkan seluruh data pertandingan yang tersedia



Gambar 16 – Menampilkan Klasemen Sementara

Menu 3 untuk menampilkan klasemen sementara dari setiap tim beserta data pertandingan masing masing tim



Gambar 17 – Mevalidasi Penyimpanan

Menu 4 untuk memvalidasi apakah user akan menyimpan data atau tidak. Setelah itu program akan berhenti.

**BAB III**

**PEMBAGIAN TUGAS**

Berikut merupakan table pembagian tugas masing-masing anggota kami,

|  |  |
| --- | --- |
| Vincentius Sutanto (2301879583) | * Coding JAVA (logical, save, show, tampilan) * Paper (latar belakang, alasan, class diagram) * PPT |
| Daniel Alferian (2301885900) | * Coding JAVA (logical, input, show, save) * Paper (use case, class and modul, screenshot program) |
| I Made Krishna Wirabuana (2301947376) | * Coding JAVA (tampilan) * Paper (latar belakang, alasan, teori, dll) * PPT |

Tabel 1 – Pembagian Tugas

**LAMPIRAN**

**Source Code**

**CLASS MAIN**

package com.company;  
  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.Collections;  
import java.util.Scanner;  
  
public class Main {  
  
 public static void main(String[] args) {  
 *// write your code here* int opt =0;  
 *//save able :* ArrayList<Tim> tims = new ArrayList<>();  
 ArrayList<Pemain> pemains = new ArrayList<>();  
 ArrayList<Wasit> wasits = new ArrayList<>();  
 ArrayList<Pertandingan> pertandingans = new ArrayList<>();  
  
 String fileTims = "tims.txt";  
 String filePemains = "pemains.txt";  
 String fileWasits = "wasits.txt";  
 String filePertansingans = "pertandingans.txt";  
  
 *//load DATAA  
 //load DATAA* System.*out*.println(" \_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_");  
 System.*out*.println("| \_\_ | | \_\_ | | \_ | | |");  
 System.*out*.println("| |\_\_| | | |\_\_| | | |\_| | | \_\_\_|");  
 System.*out*.println("| \_\_\_\_| | \_\_\_\_| | | | |\_\_\_ ");  
 System.*out*.println("| | | | | \_ | |\_\_\_\_ |");  
 System.*out*.println("| | | | | | | | \_\_\_\_| |");  
 System.*out*.println("|\_\_| |\_\_| |\_\_| |\_\_| |\_\_\_\_\_\_\_|");  
  
  
 try {  
 *//teams* ObjectInputStream inTims = new ObjectInputStream(new FileInputStream(fileTims));  
 tims = (ArrayList<Tim>) inTims.readObject();  
 inTims.close();  
  
 *//pemains* ObjectInputStream inPemains = new ObjectInputStream(new FileInputStream(filePemains));  
 pemains = (ArrayList<Pemain>) inPemains.readObject();  
 inPemains.close();  
  
 *//wasits* ObjectInputStream inWasits = new ObjectInputStream(new FileInputStream(fileWasits));  
 wasits = (ArrayList<Wasit>) inWasits.readObject();  
 inWasits.close();  
  
 *//pertandingan* ObjectInputStream inPertandingan = new ObjectInputStream(new FileInputStream(filePertansingans));  
 pertandingans = (ArrayList<Pertandingan>) inPertandingan.readObject();  
 inPertandingan.close();  
 System.*out*.println("LOAD SUCCESS\n");  
  
 } catch(IOException ex) {  
 System.*out*.println("NO DATA\n");  
 } catch(ClassNotFoundException ex) {  
 System.*out*.println("ClassNotFoundException is caught");  
 }  
  
 *//end save able* Scanner read = new Scanner(System.*in*);  
 String namaPetugas, password;  
  
 System.*out*.println(" \*Created By Vincentius Sutanto (2301879583), Daniel Alferian (2301885900), I Made Krishna (2301947376)");  
 System.*out*.println("============================================================================================================\n");  
  
  
 System.*out*.println("\nLOGIN");  
 System.*out*.print("Nama Petugas : ");  
 namaPetugas = read.nextLine();  
 do{  
 System.*out*.print("Password : ");  
 password = read.next(); *//Input password MUST BE "admin" or "Admin"* if(!password.equals("admin") && !password.equals("Admin"))  
 System.*out*.println("Password SALAH");  
 }while(!password.equals("admin") && !password.equals("Admin"));  
 read.nextLine();  
  
 System.*out*.println("\n");  
 *//interface* do{  
 System.*out*.println("╔═╦═╦═╦═╦═╦═╗ ╔╦╦╗");  
 System.*out*.println("║╚╣║║╔╣╔╣═╣╚╣ ║║║║");  
 System.*out*.println("╠╗║║║╚╣╚╣═╣║║ ╠╬╬╣");  
 System.*out*.println("╚═╩═╩═╩═╩═╩╩╝ ╚╩╩╝");  
 System.*out*.println("+=====================+");  
 System.*out*.println("| MENU LIST |");  
 System.*out*.println("+=====================+");  
 System.*out*.println("| 1. Input Data |");  
 System.*out*.println("| 2. View Data |");  
 System.*out*.println("| 3. Tournament Score |");  
 System.*out*.println("| 4. Save and Exit |");  
 System.*out*.println("+=====================+");  
 System.*out*.print("Choice [1-4] >> ");  
 try{  
 opt = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 opt = 0;  
 }  
 String dummy=read.nextLine();  
  
 switch (opt){  
 *//pendataan* case 1 :{  
 *pendataan*(tims, pemains, wasits, pertandingans);  
 *//pendataan* break;  
 }  
 *//view data* case 2 : {  
 *viewMenu*(tims, pemains, wasits, pertandingans);  
 *//view data* break;  
 }  
 *//standing* case 3 : {  
 if (tims.size()>0){  
 *viewStanding*(tims);  
 }else System.*out*.println("\nNO DATA\n");  
 break;  
 }  
 *//exit & save* case 4 : {  
 String pilihan;  
 do{  
 System.*out*.print("Save Data? [ya/tidak]");  
 pilihan = read.nextLine();  
 }while (!pilihan.equals("ya") && !pilihan.equals("tidak"));  
 if (pilihan.equals("ya")){  
 *saveExit*(tims, pemains, wasits, pertandingans, fileTims, filePemains, fileWasits, filePertansingans);  
 }  
 *// exit and save* break;  
 }  
 default : {  
 System.*out*.println("\nPilihan tidak ada\n");  
 break;  
 }  
 }  
 } while (opt != 4);  
 }  
  
 *//save data* private static void saveExit(ArrayList<Tim> tims, ArrayList<Pemain> pemains, ArrayList<Wasit> wasits, ArrayList<Pertandingan> pertandingans, String fileTims, String filePemains, String fileWasits, String filePertansingans) {  
 try  
 {  
 *//Saving of object in a file  
 //tims* ObjectOutputStream outTims = new ObjectOutputStream(new FileOutputStream(fileTims));  
 outTims.writeObject(tims);  
  
 *//pemain* ObjectOutputStream outPemains = new ObjectOutputStream(new FileOutputStream(filePemains));  
 outPemains.writeObject(pemains);  
  
 *//wasit* ObjectOutputStream outWasits = new ObjectOutputStream(new FileOutputStream(fileWasits));  
 outWasits.writeObject(wasits);  
  
 *//pertandingan* ObjectOutputStream outPertandingans = new ObjectOutputStream(new FileOutputStream(filePertansingans));  
 outPertandingans.writeObject(pertandingans);  
  
 System.*out*.println("\nDONE SAVE");  
  
 }  
 catch(IOException ex)  
 {  
 System.*out*.println("Error SAVE DATA");  
 }  
 }  
  
 *//menu1* private static void pendataan(ArrayList<Tim> tims, ArrayList<Pemain> pemains, ArrayList<Wasit> wasits, ArrayList<Pertandingan> pertandingans) {  
 int opt = 0;  
 Scanner read = new Scanner(System.*in*);  
  
 *//interface* System.*out*.println();  
 System.*out*.println("+=======================+");  
 System.*out*.println("| MENU INPUT |");  
 System.*out*.println("+=======================+");  
 System.*out*.println("| 1. Add Tim dan Pemain |");  
 System.*out*.println("| 2. Add Wasit |");  
 System.*out*.println("| 3. Add Pertandingan |");  
 System.*out*.println("+=======================+");  
 System.*out*.print("Choice [1-3] >> ");  
  
 try{  
 opt = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 }  
 String dummy=read.nextLine();  
  
 switch (opt){  
 *//input pemain* case 1 : {  
 *//team dan pemain* String namaTim,alamatTim, pelatih;  
 String pilihan, namaPemain, posisi;  
 int usiaPemain;  
  
 *//interface  
 //nama tim* System.*out*.println("\n");  
 do{  
 System.*out*.print("Masukkan nama tim [maksimal 20 character] : ");  
 namaTim = read.nextLine();  
 }while (namaTim.length()<1 || namaTim.length()>20);  
  
 *//alamat tim* do{  
 System.*out*.print("Masukkan alamat tim [maksimal 20 character] : ");  
 alamatTim = read.nextLine();  
 }while (alamatTim.length()<1 || alamatTim.length()>20);  
  
 *//pelatih tim* do{  
 System.*out*.print("Masukkan pelatih tim [maksimal 20 character] : ");  
 pelatih = read.nextLine();  
 }while (pelatih.length()<1 || pelatih.length()>20);  
  
  
 *//* System.*out*.println("\nDATA PEMAIN");  
 int counter=1;  
 *//pemain tim* do{  
 *//nama pemain* do{  
 System.*out*.print("Masukkan nama pemain [masksimal 20 character] : ");  
 namaPemain = read.nextLine();  
 }while (namaPemain.length()<1 || namaPemain.length()>20);  
  
 *//usia pemain* do{  
 System.*out*.print("Masukkan usia pemain [angka]: ");  
 try{  
 usiaPemain = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 usiaPemain=0;  
 }  
 dummy=read.nextLine();  
 }while (usiaPemain<1);  
  
 *//posisi pemain* do{  
 System.*out*.print("Masukkan posisi pemain pemain [masksimal 20 character] : ");  
 posisi = read.nextLine();  
 }while (posisi.length()<1 || posisi.length()>20);  
  
 do{  
 System.*out*.print("Ada tambahan pemain lagi? [ya/tidak]");  
 pilihan = read.nextLine();  
 }while (!pilihan.equals("ya") && !pilihan.equals("tidak"));  
  
 *//add pada array <> pemain* pemains.add(new Pemain(namaPemain, usiaPemain, posisi, namaTim));  
 counter++;  
 }while (pilihan.equals("ya"));  
  
 *//add pada array <> tim* tims.add(new Tim(namaTim, alamatTim, pelatih));  
  
  
 try {  
 System.*out*.println("\nTeam berhasil ditambahkan");  
 Thread.*sleep*(500);  
 }catch (Exception e) {  
 System.*out*.println("Exception is caught");  
 }  
  
 for (int i=0; i<counter; i++){  
 try {  
 System.*out*.println("Pemain berhasil ditambahkan");  
 Thread.*sleep*(500);  
 } catch (Exception e) {  
 System.*out*.println("Exception is caught");  
 }  
 }  
 break;  
 }  
 *//input wasit* case 2 : {  
 *//wasit* String namaWasit, posisiWasit;  
 int usiawasit;  
  
 *//interface* System.*out*.println("\n");  
 do{  
 System.*out*.print("Masukkan nama wasit [maksimal 20 character] : ");  
 namaWasit = read.nextLine();  
 }while (namaWasit.length()<1 || namaWasit.length()>20);  
  
 do{  
 System.*out*.print("Masukkan usia wasit [angka]: ");  
 try{  
 usiawasit = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 usiawasit=0;  
 }  
 dummy=read.nextLine();  
 }while (usiawasit<1);  
  
 do{  
 System.*out*.print("Masukkan posisi wasit [maksimal 20 character] : ");  
 posisiWasit = read.nextLine();  
 }while (posisiWasit.length()<1 || posisiWasit.length()>20);  
  
 *//add pada array<> wasit* wasits.add(new Wasit(namaWasit, usiawasit, posisiWasit));  
  
 System.*out*.println("Wasit berhasil ditambahkan\n\n");  
 break;  
 }  
 *//input pertandingan* case 3 : {  
 *//menghindari pertandingan tanpa peserta dan wasit* if (tims.size()>1 && !wasits.isEmpty()){  
 int tanggal, bulan, tahun;  
 String namaKota, stadion;  
 String homeTim, awayTim, wasit;  
 int xHometim, yAwayTim;  
 int xwasit;  
 String pilihan;  
  
 System.*out*.println("\nDATA PERTANDINGAN");  
 *//nama kota* do{  
 System.*out*.print("Masukkan nama kota [maksimal 20 character] : ");  
 namaKota = read.nextLine();  
 }while (namaKota.length()<1 || namaKota.length()>20);  
  
 *//stadion* do{  
 System.*out*.print("Masukkan nama stadion [maksimal 20 character] : ");  
 stadion = read.nextLine();  
 }while (stadion.length()<1 || stadion.length()>20);  
  
 *//tanggal* do{  
 System.*out*.print("Masukkan tanggal pertandingan [angka| 1-31] : ");  
 try{  
 tanggal = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input\n");  
 tanggal=0;  
 }  
 dummy=read.nextLine();  
 }while (tanggal<1 || tanggal>31);  
  
 *//bulan* do{  
 System.*out*.print("Masukkan bulan pertandingan [angka | 1-12] : ");  
 try{  
 bulan = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input\n");  
 bulan=0;  
 }  
 dummy=read.nextLine();  
 }while (bulan<1 || bulan>12);  
  
 *//tahun* do{  
 System.*out*.print("Masukkan tahun pertandingan [angka <3000]: ");  
 try{  
 tahun = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input\n");  
 tahun=0;  
 }  
 dummy=read.nextLine();  
 }while (tahun<1 || tahun>3000);  
  
 *tamplianTim*(tims);  
 *//memilih home team* do{  
 System.*out*.print("Masukkan pilihan HOME tim [angka nomer urut] : ");  
 try{  
 xHometim = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 xHometim=0;  
 }  
 dummy=read.nextLine();  
 }while (xHometim<1 || xHometim>tims.size());  
  
 *//memilih away team* do{  
 System.*out*.print("Masukkan pilihan AWAY tim [angka nomer urut] : ");  
 try{  
 yAwayTim = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 yAwayTim=0;  
 }  
 dummy=read.nextLine();  
 }while (yAwayTim<1 || yAwayTim>tims.size() || yAwayTim==xHometim);  
 *//  
 //get nama team dari array team  
 //* xHometim=xHometim-1;  
 yAwayTim=yAwayTim-1;  
 *//home team* Tim xx = tims.get(xHometim);  
 homeTim = xx.getNamaTim();  
 *//away team* Tim yy = tims.get(yAwayTim);  
 awayTim = yy.getNamaTim();  
  
  
 *//pemain yang sedang pelanggaran* for (Pemain p : pemains){  
 if (p.getTim().equals(homeTim) || p.getTim().equals(awayTim)){  
 if (p.getSanksi()>0){  
 System.*out*.println("Pemain dengan nama " +p.getNama() + " Sedang dalam pelanggaran dan tidak dapat bermain");  
 p.kurangSanksi();  
 }  
 }  
 }  
  
 *//pendataan score :* int scoreHome, scoreAway;  
 System.*out*.println("\nDATA SCORE PERTANDINGAN");  
 System.*out*.println(homeTim + " VS " + awayTim);  
 do{  
 System.*out*.print("Masukkan score HOME tim : ");  
 try{  
 scoreHome = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input\n");  
 scoreHome=-1;  
 }  
 dummy=read.nextLine();  
 }while (scoreHome<0);  
  
 do{  
 System.*out*.print("Masukkan score AWAY tim : ");  
 try{  
 scoreAway = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input\n");  
 scoreAway=-1;  
 }  
 dummy=read.nextLine();  
 }while (scoreAway<0);  
  
  
  
 *//update score team* for (Tim t : tims){  
 *//home team update* if (t.getNamaTim().equals(homeTim)){  
 t.tambahPertandingan();  
  
 if (scoreHome>scoreAway){  
 t.tMenang();  
 } else if (scoreHome<scoreAway){  
 t.tKalah();  
 } else {  
 t.tDraw();  
 }  
 }  
 if (t.getNamaTim().equals(awayTim)){  
 t.tambahPertandingan();  
 if (scoreHome>scoreAway){  
 t.tKalah();  
 }else if (scoreHome<scoreAway){  
 t.tMenang();  
 }else {  
 t.tDraw();  
 }  
 }  
 }  
  
  
  
 *//memilih wasit  
 tamplianWasit*(wasits);  
 do{  
 System.*out*.print("Masukkan pilihan wasit [angka] : ");  
 try{  
 xwasit = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 xwasit=0;  
 }  
 dummy=read.nextLine();  
 }while (xwasit<1 || xwasit>wasits.size());  
  
 xwasit = xwasit-1;  
 *//get nama team dari array wasit* Wasit xw = wasits.get(xwasit);  
 wasit = xw.getNama();  
  
  
 String timPelanggar, pilihanPelanggaran, namaPelanggar;  
 boolean namaBenar;  
 int jumlahKuning=0, jumlahMerah=0;  
  
  
 *//pendataan pelanggaran* System.*out*.println("\nDATA PELANGGARAN");  
 do{  
 System.*out*.print("Ada pelanggaran ? [ya/tidak] : ");  
 pilihan = read.nextLine();  
 }while (!pilihan.equals("ya") && !pilihan.equals("tidak"));  
  
 while (pilihan.equals("ya")){  
 namaBenar = false;  
 *//memilih pemain yang dapet sanksi  
 //show pemain yang teamnya bermain  
  
 pemainPelanggar*(pemains, homeTim, awayTim);  
  
 do {  
 System.*out*.print("Masukkan nama team [sesuai team yang bermain] : ");  
 timPelanggar = read.nextLine();  
 }while (!timPelanggar.equals(homeTim) && !timPelanggar.equals(awayTim));  
  
 do {  
 System.*out*.print("Masukkan jenis pelanggaran [merah/kuning] : ");  
 pilihanPelanggaran = read.nextLine();  
 }while (!pilihanPelanggaran.equals("merah") && !pilihanPelanggaran.equals("kuning"));  
  
 do {  
 System.*out*.print("Masukkan nama pemain yang melanggar : ");  
 namaPelanggar = read.nextLine();  
  
 for (Pemain ppp : pemains){  
 if (ppp.getTim().equals(timPelanggar)){  
 if (ppp.getNama().equals(namaPelanggar)){  
 if (pilihanPelanggaran.equals("merah")){  
 ppp.pelanggaranMerah();  
 jumlahMerah = jumlahMerah+1;  
 }  
 if (pilihanPelanggaran.equals("kuning")){  
 ppp.pelanggaranKuning();  
 jumlahKuning = jumlahKuning+1;  
 }  
 namaBenar = true;  
 }  
 }  
 }  
  
 }while (!namaBenar);  
  
 do{  
 System.*out*.print("Ada tambahan pelanggaran lagi? [ya/tidak] : ");  
 pilihan = read.nextLine();  
 }while (!pilihan.equals("ya") && !pilihan.equals("tidak"));  
 }  
  
 *//array pertandingan* pertandingans.add(new Pertandingan(homeTim, awayTim, scoreHome, scoreAway, wasit, namaKota, stadion, tanggal, bulan, tahun, jumlahKuning, jumlahMerah));  
 *//bisa* } else {  
 System.*out*.println("\nMasukkan minimal 2 tim dan 1 wasit terlebih dahulu\n");  
 }  
 *//pertandingan* break;  
 }  
  
 default:{  
 System.*out*.println("Pilihan tidak ada\n\n");  
 break;  
 }  
 }  
 }  
  
 *//sub menu 1* private static void pemainPelanggar(ArrayList<Pemain> pemains, String homeTim, String awayTim) {  
 System.*out*.println("\nDATA PEMAIN TIM");  
  
 System.*out*.println("Tim : " + homeTim);  
 System.*out*.println("+=============================================+");  
 System.*out*.println("| Nama Pemain | Posisi |");  
 System.*out*.println("+=============================================+");  
 for (Pemain pp : pemains){  
 if (pp.getTim().equals(homeTim)){  
  
 System.*out*.printf("| %-20s | %-20s |\n", pp.getNama(), pp.getPosisi());  
 }  
 }  
 System.*out*.println("+=============================================+");  
  
 System.*out*.println("\nTim : " + awayTim);  
 System.*out*.println("+=============================================+");  
 System.*out*.println("| Nama Pemain | Posisi |");  
 System.*out*.println("+=============================================+");  
 for (Pemain pp : pemains){  
 if (pp.getTim().equals(awayTim)){  
 System.*out*.printf("| %-20s | %-20s |\n", pp.getNama(), pp.getPosisi());  
 }  
 }  
 System.*out*.println("+=============================================+");  
 }  
  
 *//menu3* private static void viewStanding(ArrayList<Tim> tims) {  
 int no =1;  
 System.*out*.println("\n ╔═╦═╦═╦╗╔╦╦╗ ╔═╦═╦═╦╗╔╦╦╗ ╔═╦═╦═╦╗╔╦╦╗");  
 System.*out*.println(" ║╚╣║║║║║║║║║ ║╚╣║║║║║║║║║ ║╚╣║║║║║║║║║");  
 System.*out*.println(" ║║║║║╦║╚╬╬╬╣ ║║║║║╦║╚╬╬╬╣ ║║║║║╦║╚╬╬╬╣");  
 System.*out*.println(" ╚═╩═╩╩╩═╩╩╩╝ ╚═╩═╩╩╩═╩╩╩╝ ╚═╩═╩╩╩═╩╩╩╝\n");  
  
 System.*out*.println("+============================================================================================================================+");  
 System.*out*.println("| No | Nama Tim | Alamat | Pelatih | Pertandingan |Menang | Kalah | Draw | Poin |");  
 System.*out*.println("+============================================================================================================================+");  
  
 Collections.*sort*(tims);  
 for (Tim t : tims){  
 System.*out*.printf( "| %-3d | %-20s | %-20s | %-20s | %-12d |%-6d | %-6d | %-6d | %-6d |\n", no,t.getNamaTim(), t.getAlamatTim(), t.getPelatih(), t.getJumlahPertandingan(),t.getMenang(), t.getKalah(), t.getDraw(), t.getPoin());  
 no++;  
 }  
 System.*out*.println("+============================================================================================================================+");  
  
 System.*out*.println("Press Enter key to continue...\n");  
 try  
 {  
 System.*in*.read();  
 }  
 catch(Exception e)  
 {}  
 *//menampilkan seluruh tim* }  
  
 *//menu2* private static void viewMenu(ArrayList<Tim> tims, ArrayList<Pemain> pemains, ArrayList<Wasit> wasits, ArrayList<Pertandingan> pertandingans) {  
 int x=0;  
 Scanner read = new Scanner(System.*in*);  
  
 System.*out*.println("\n+===========================+");  
 System.*out*.println("| MENU VIEW |");  
 System.*out*.println("+===========================+");  
 System.*out*.println("| 1. View Data Tim |");  
 System.*out*.println("| 2. View Data Pemain |");  
 System.*out*.println("| 3. View Data Wasit |");  
 System.*out*.println("| 4. View Data Pertandingan |");  
 System.*out*.println("+===========================+");  
 System.*out*.print("Choice [1-3] >> ");  
  
 try{  
 x = read.nextInt();  
 }catch (Exception e){  
 System.*out*.print("wrong input");  
 x = 0;  
 }  
 String dummy=read.nextLine();  
  
 switch (x){  
 case 1 : {  
 if (tims.size()>0){  
 *tamplianTim*(tims);  
 System.*out*.println("Press Enter key to continue...\n");  
 try  
 {  
 System.*in*.read();  
 }  
 catch(Exception e)  
 {}  
 }else System.*out*.println("\n[ There is No Team Registered ]\n");  
 break;  
 }  
  
 case 2 : {  
 if (pemains.size()>0){  
 *tampilanPemain*(pemains);  
 System.*out*.println("Press Enter key to continue...\n");  
 try  
 {  
 System.*in*.read();  
 }  
 catch(Exception e)  
 {}  
 }else System.*out*.println("\n[ There is No Player Registered ]\n");  
 break;  
 }  
  
 case 3 : {  
 if (wasits.size()>0){  
 *tamplianWasit*(wasits);  
 System.*out*.println("Press Enter key to continue...\n");  
 try  
 {  
 System.*in*.read();  
 }  
 catch(Exception e)  
 {}  
 }else System.*out*.println("\n[ There is No Referee Registered ]\n");  
 break;  
 }  
  
 case 4 : {  
 if (pertandingans.size()>0){  
 *tampilanPertandingan*(pertandingans);  
 System.*out*.println("Press Enter key to continue...\n");  
 try  
 {  
 System.*in*.read();  
 }  
 catch(Exception e)  
 {}  
 }else System.*out*.println("\n[ There is No Game Registered ]\n");  
 break;  
 }  
  
 default:{  
 System.*out*.println("\nPilihan tidak ada\n");  
 break;  
 }  
 }  
   
 }  
  
 *//sub menu 2* private static void tampilanPertandingan(ArrayList<Pertandingan> pertandingans) {  
 int no =1;  
 System.*out*.println("\nDATA PERTANDINGAN");  
  
 for (Pertandingan p : pertandingans){  
 System.*out*.println("\nPertandingan KE " + no);  
 System.*out*.println("HOME VS AWAY");  
 System.*out*.printf("%-20s | %-3d VS %3d | %20s\n\n", p.getHomeTeam(), p.getScoreHome(), p.getScoreAway(), p.getAwayTeam());  
 System.*out*.println("| Wasit | Stadion | Kota | tgl-bln-thn | Kuning | Merah |");  
 System.*out*.println("+=======================================================================================================+");  
 System.*out*.printf("| %-20s | %-20s | %-20s | %-2d - %-2d - %-4d | %-6d | %-6d |\n\n", p.getWasit(), p.getStadion(), p.getKota(), p.getTanggal(), p.getBulan(), p.getTahun(), p.getKartuKuning(), p.getKartuMerah());  
 no++;  
 }  
 }  
  
 *//sub menu 2* private static void tampilanPemain(ArrayList<Pemain> pemains) {  
 int no =1;  
 System.*out*.println("\nDATA PEMAIN\n");  
 System.*out*.println("+===================================================================================================+");  
 System.*out*.println("| No | Nama Pemain | Usia | Posisi | Kuning | Merah | Tim |");  
 System.*out*.println("+===================================================================================================+");  
  
 for (Pemain p : pemains){  
 System.*out*.printf( "| %-3d | %-20s | %-4d | %-20s | %-6d | %-6d | %-20s |\n", no,p.getNama(), p.getUsia(), p.getPosisi(), p.getKartuKuning(), p.getKartuMerah(), p.getTim());  
 no++;  
 }  
 System.*out*.println("+===================================================================================================+");  
 }  
  
 *//sub menu 2* private static void tamplianTim(ArrayList<Tim> tims) {  
 int no =1;  
 System.*out*.println("\nDATA TIM\n");  
  
 System.*out*.println("+==========================================================================+");  
 System.*out*.println("| No | Nama Tim | Alamat | Pelatih |");  
 System.*out*.println("+==========================================================================+");  
  
 for (Tim t : tims){  
 System.*out*.printf( "| %-3d | %-20s | %-20s | %-20s |\n", no,t.getNamaTim(), t.getAlamatTim(), t.getPelatih());  
 no++;  
 }  
 System.*out*.println("+==========================================================================+");  
 }  
  
 *//sub menu 2* private static void tamplianWasit(ArrayList<Wasit> wasits) {  
 int no =1;  
 System.*out*.println("\nDATA WASIT\n");  
 System.*out*.println("+==========================================================+");  
 System.*out*.println("| No | Nama Wasit | Usia | Posisi |");  
 System.*out*.println("+==========================================================+");  
  
 for (Wasit w : wasits){  
 System.*out*.printf("| %-3d | %-20s | %-4d | %-20s |\n", no,w.getNama(), w.getUsia(), w.getPosisi());  
 no++;  
 }  
 System.*out*.println("+==========================================================+");  
 *//menampilkan seluruh wasit* }  
  
}

**Class Manusia**

package com.company;  
  
import java.io.Serializable;  
  
public class Manusia implements Serializable {  
  
 private String nama;  
 private int usia;  
  
 public Manusia(String nama, int usia) {  
 this.nama = nama;  
 this.usia = usia;  
 }  
  
 *//getter* public String getNama() {  
 return nama;  
 }  
  
 public int getUsia() {  
 return usia;  
 }  
}

**Class Wasit**

package com.company;  
  
  
public class Wasit extends Manusia {  
  
 private String posisi;  
  
 public Wasit(String nama, int usia, String posisi) {  
 super(nama, usia);  
 this.posisi = posisi;  
 }  
  
 *//getter* @Override  
 public String getNama() {  
 return super.getNama();  
 }  
  
 @Override  
 public int getUsia() {  
 return super.getUsia();  
 }  
  
 public String getPosisi() {  
 return posisi;  
 }  
}

**Class Pemain**

package com.company;  
  
  
public class Pemain extends Manusia implements PelanggaranPemain{  
  
 private String tim;  
 private String posisi;  
 private int kartuKuning;  
 private int kartuMerah;  
 int sanksi=0;  
  
 public Pemain(String nama, int usia, String posisi, String tim) {  
 super(nama, usia);  
 this.posisi = posisi;  
 this.tim = tim;  
 }  
  
 *//getter* @Override  
 public String getNama() {  
 return super.getNama();  
 }  
  
 public String getPosisi() {  
 return posisi;  
 }  
  
 public String getTim() {  
 return tim;  
 }  
  
 @Override  
 public int getUsia() {  
 return super.getUsia();  
 }  
  
 public int getKartuKuning() {  
 return kartuKuning;  
 }  
  
 public int getKartuMerah() {  
 return kartuMerah;  
 }  
  
 public int getSanksi() {  
 return sanksi;  
 }  
  
 *//method* public void kurangSanksi(){  
 sanksi = sanksi -1;  
 }  
  
 *//Interface pelanggaran untuk menjelaskan pelanggaran* @Override  
 public void pelanggaranMerah() {  
 sanksi= sanksi+2;  
 kartuMerah = kartuMerah+1;  
 }  
  
 @Override  
 public void pelanggaranKuning() {  
 sanksi = sanksi+1;  
 kartuKuning = kartuKuning+1;  
 }  
}

**Class Tim**

package com.company;  
  
  
import java.io.Serializable;  
  
public class Tim implements Comparable<Tim>, Serializable {  
  
 private String namaTim;  
 private String alamatTim;  
 private String pelatih;  
 private int poin =0;  
 private int menang;  
 private int kalah;  
 private int draw;  
 private int jumlahPertandingan= 0;  
  
 public Tim(String namaTim, String alamatTim, String pelatih) {  
 this.namaTim = namaTim;  
 this.alamatTim = alamatTim;  
 this.pelatih = pelatih;  
 }  
  
 *// method* public void tMenang(){  
 poin = poin+2;  
 menang = menang+1;  
 }  
  
 public void tKalah(){  
 kalah = kalah+1;  
 }  
  
 public void tDraw(){  
 poin = poin+1;  
 draw = draw+1;  
 }  
  
 public void tambahPertandingan(){  
 jumlahPertandingan = jumlahPertandingan+1;  
 }  
 *//setter  
  
 //compare* @Override  
 public int compareTo(Tim o) {  
 return o.poin - this.poin;  
 }  
  
  
 *//getter* public String getNamaTim() {  
 return namaTim;  
 }  
  
 public String getAlamatTim() {  
 return alamatTim;  
 }  
  
 public String getPelatih() {  
 return pelatih;  
 }  
  
 public int getPoin() {  
 return poin;  
 }  
  
 public int getMenang() {  
 return menang;  
 }  
  
 public int getKalah() {  
 return kalah;  
 }  
  
 public int getDraw() {  
 return draw;  
 }  
  
 public int getJumlahPertandingan() {  
 return jumlahPertandingan;  
 }  
}

**Class Pertandingan**

package com.company;  
  
import java.io.Serializable;  
  
public class Pertandingan implements Serializable {  
 private String homeTeam;  
 private String awayTeam;  
 private int scoreHome;  
 private int scoreAway;  
 private String wasit;  
 private String kota;  
 private String stadion;  
 private int tanggal;  
 private int bulan;  
 private int tahun;  
 private int kartuKuning;  
 private int kartuMerah;  
  
 public Pertandingan(String homeTeam, String awayTeam,int scoreHome, int scoreAway ,String wasit ,String kota, String stadion, int tanggal, int bulan, int tahun, int kartuKuning, int kartuMerah){  
 this.homeTeam = homeTeam;  
 this.awayTeam = awayTeam;  
 this.scoreHome = scoreHome;  
 this.scoreAway = scoreAway;  
 this.wasit = wasit;  
 this.kota = kota;  
 this. stadion = stadion;  
 this.tanggal = tanggal;  
 this.bulan = bulan;  
 this.tahun = tahun;  
 this.kartuKuning = kartuKuning;  
 this.kartuMerah = kartuMerah;  
 }  
  
 *//getter* public String getHomeTeam() {  
 return homeTeam;  
 }  
  
 public String getAwayTeam() {  
 return awayTeam;  
 }  
  
 public int getScoreHome() {  
 return scoreHome;  
 }  
  
 public int getScoreAway() {  
 return scoreAway;  
 }  
  
 public String getWasit() {  
 return wasit;  
 }  
  
 public String getStadion() {  
 return stadion;  
 }  
  
 public String getKota() {  
 return kota;  
 }  
  
 public int getTanggal() {  
 return tanggal;  
 }  
  
 public int getBulan() {  
 return bulan;  
 }  
  
 public int getTahun() {  
 return tahun;  
 }  
  
 public int getKartuKuning() {  
 return kartuKuning;  
 }  
  
 public int getKartuMerah() {  
 return kartuMerah;  
 }  
}

**Interface PelanggaranPemain**

package com.company;  
  
public interface PelanggaranPemain {  
 public void pelanggaranMerah();  
 public void pelanggaranKuning();  
}

# **Referensi**

|  |  |
| --- | --- |
| [1] | B. Stroustrup, "What is ‘‘Object-Oriented Programming’’?," pp. 10-20, 1988. |
| [2] | M. Skoglund, "Practical Use of Encapsulation in Object-Oriented Programming," 2003.  3. https://www.w3schools.com/java/java\_arraylist.asp  4. https://www.w3schools.com/java/java\_interface.asp  5. https://www.geeksforgeeks.org/java-class-and-object/ |
|  |  |