Criteria C Development

Project structure

Extensibility

Data structure: arrays, collections… justify choice

Algorithmic: flag, accumulator, early break or return, iteration and conditions

OOP: abstraction, polymorphism,inheritace,encapsulation

Concurrency

Listener/event

Exception Handling

Modularity package, library, modules

Test: integration, unit

Maintenance

Readibility

Performance

Scope and lifetime

Private, public , protected

Static, constant, immutable

DB transaction, SQL, connection , list as return queries, insert/retrieve

# Overview

## Project Structure

We divided our project into the following java packages to get more modularity , facilitate maintenance (debug, test) and decouple the development.

Db for the java database access and the sql scripts

Model for the domain model

Navigation and view for the view model

Service for the services that implements the games

And utils for transversal functionalities

Interfaz de usuario gráfica, Tabla

El contenido generado por IA puede ser incorrecto.

## Unit Testing/Database

On my main function we used argument to three modes that you may pass at the start command:

* Unit test mode, so that the application perform all the unit test
* DB Mockup/Actual DB, so that i could implement the java part without need of the actual database
* Production mode, which is the one used for usual usage and deployment to our clients

Texto

El contenido generado por IA puede ser incorrecto.

## Comments

We took care of commenting all class and methods, and sometimes add comments to the code to make it readable and maintenable:

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

# Java Package Details

## Model, DB

## Service

## View

## Navigation

## Utils

For each of them show the listing of their classes and focus on:

* Model (forgotten before)
  + constructor override
  + private constructor
* // development
  + Mockup
* Inheritance
  + Model DbId
* Abstraction and polymorphisme
  + DbgMe and DbgMeAsStr in Model
  + BaseView and Views
  + DBid Model and game, player..
  + Abstract class AbtractGame
  + Interface DB Mockup
* Main parameters, testability, maintenance
  + Test Mode in Main
* Test
  + Unit test (Unit test by AI / chaptgpt forgotten)
  + Assert function
* Java special complexity:
  + Actionlistener in Views
  + GameEndListener
  + UI buttons and swing library
* Algorithmic
  + List iteration, ArrayList, List, arrays
  + Array direct access
* Mockup
  + List iterate, add ,remove,update
* Comments : readability and understandibility
  + For classes, variables, functions
* Third party class for combo object
* [Navigation and AppStatus](Navigation%20and%20AppStatus)
* Navigation unit test
* Database access classes
* SQL integration

Add some UML class diagram anywhere you can

# Database:

## Mockup

We decided to use a mockup of the database and delay its full implementation at the end of the project.

## modeling

Modeled using sqldbm, we created unique name constraint for game and players, and of course all required primary key

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

## Managing and developing

To easily manage, deploy and test it, we created 3 sql scripts to:

* Create/update the scheme
* Add initial values to our scheme
* Test and implement our sql statements

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.