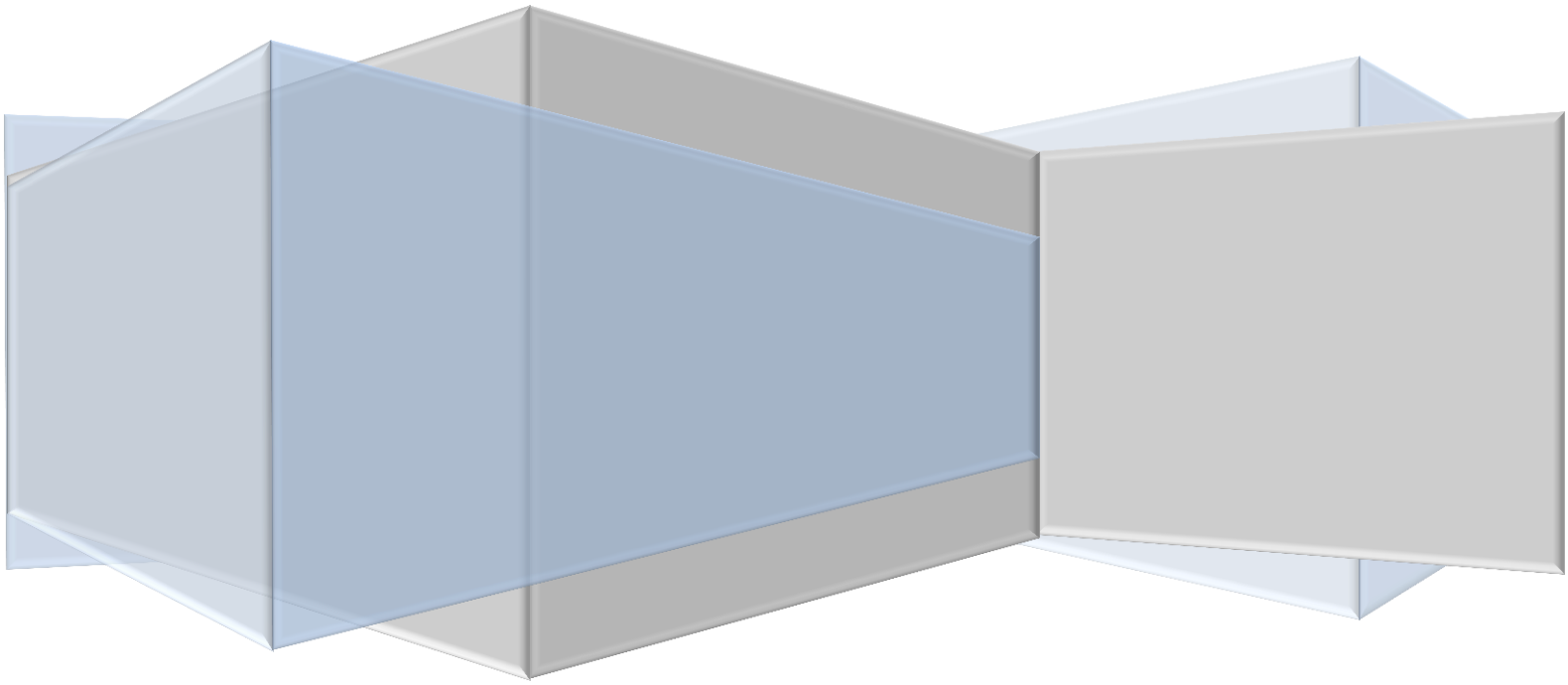
University of Oviedo

Documentation Third Deliverable

Software Architecture

Ana Areces
Raquel Arrojo
David Casado
María González
Álvaro Palanco
Iván Sánchez



Content

Project information

All the documentation files can be found inside the documentation folder in our Github repository.
https://github.com/Arquisoft/Trivial_i1b

This is a direct access to the Readme that contains the general information of the group.
https://github.com/Arquisoft/Trivial_i1b/blob/master/README.md

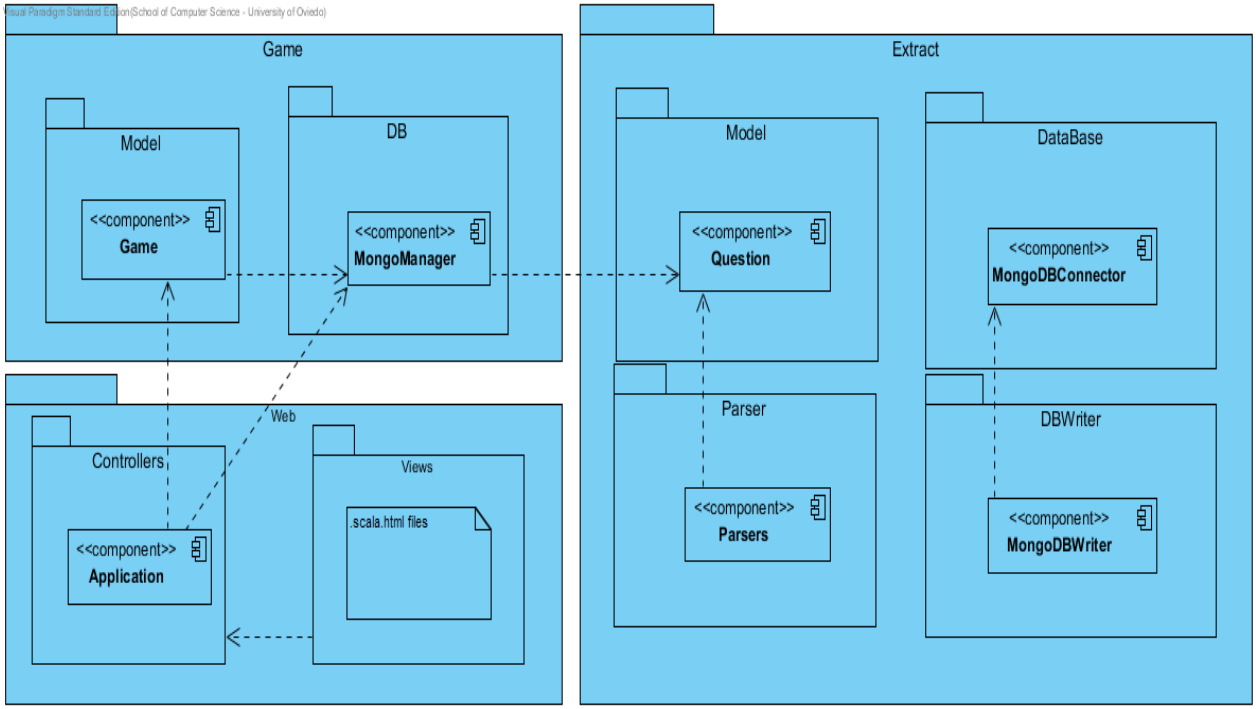
This folder contains the Documentation information like manuals and all the diagrams, also the Visual Paradigm project.
https://github.com/Arquisoft/Trivial_i1b/tree/web/Documentation

The generated documentation.
https://github.com/Arquisoft/Trivial_i1b/blob/web/Documentation/GeneratedVPDocumentation_Deliverable3.pdf

This is the project, where you could find all the implementation.
https://github.com/Arquisoft/Trivial_i1b/tree/web

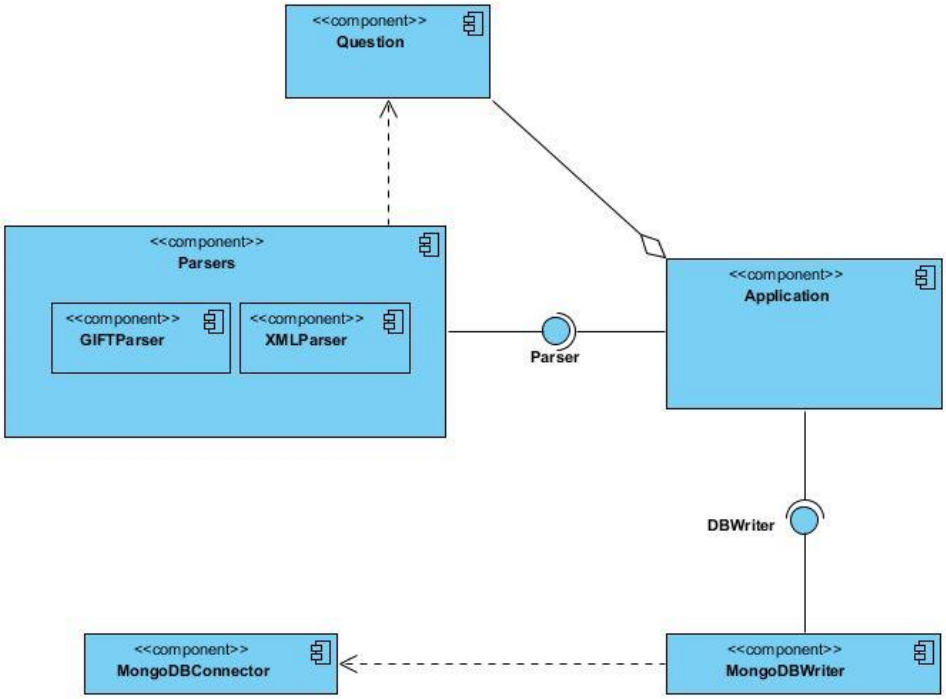
Architecture diagrams

Package view

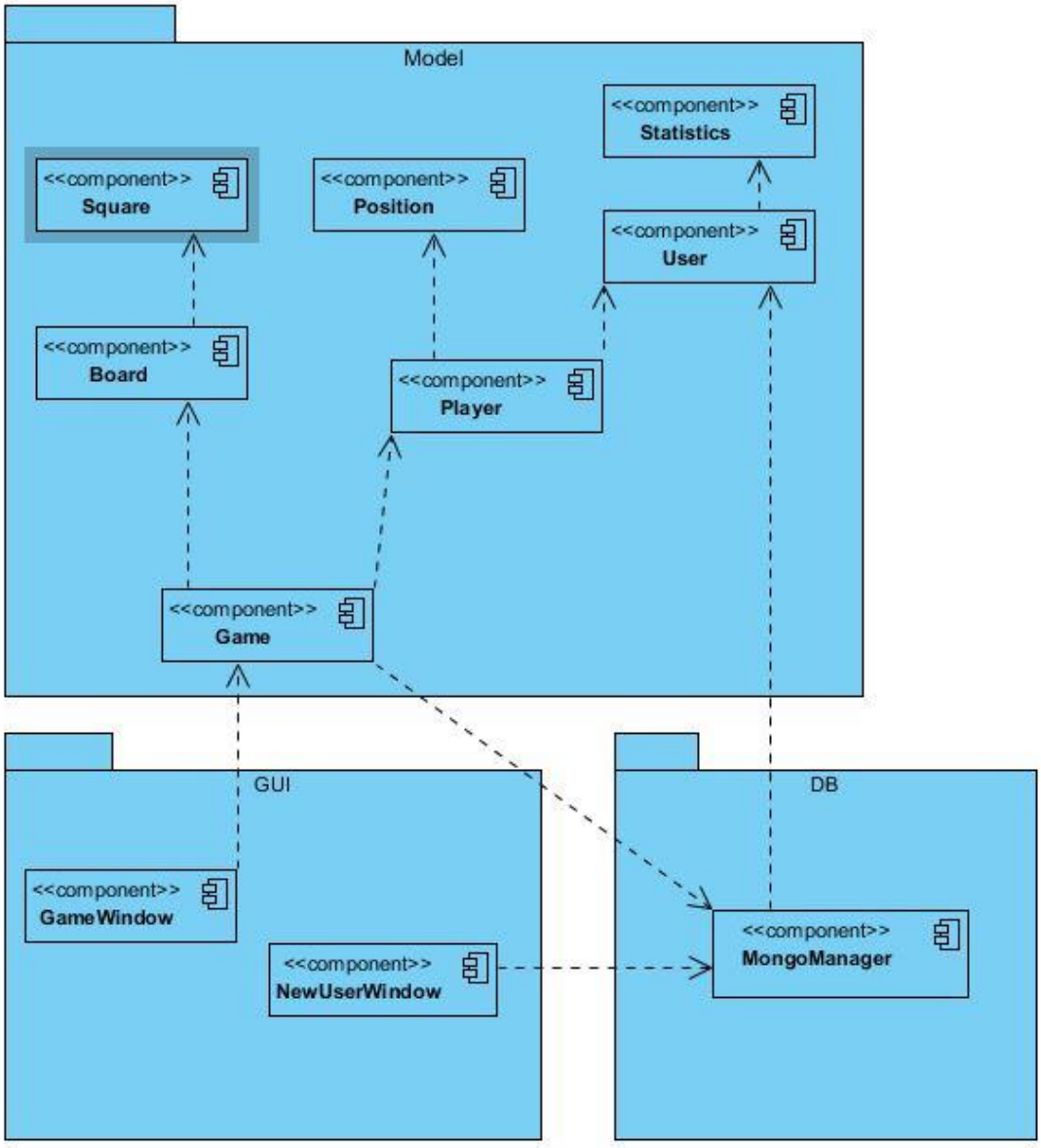


Component views

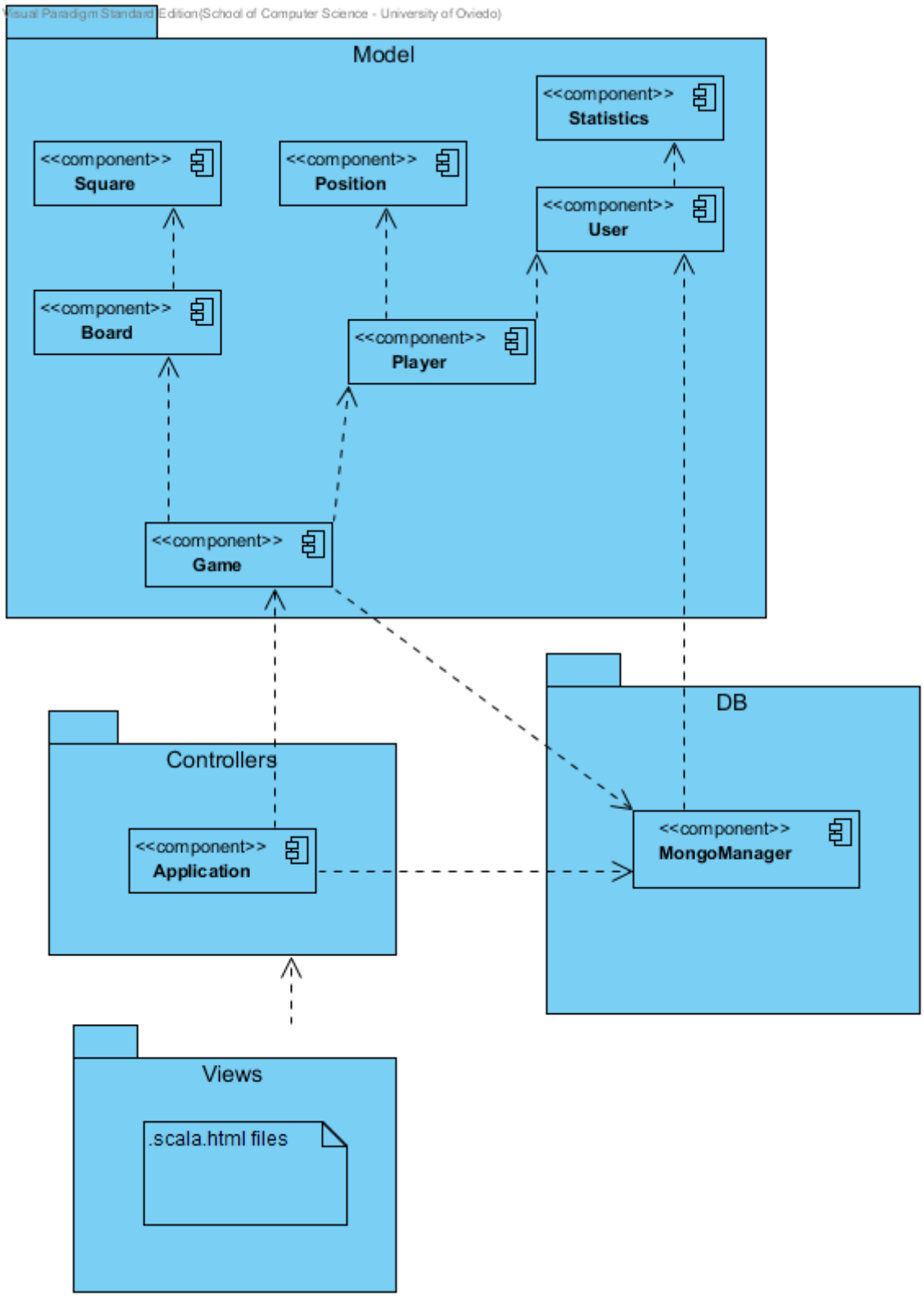
Extract



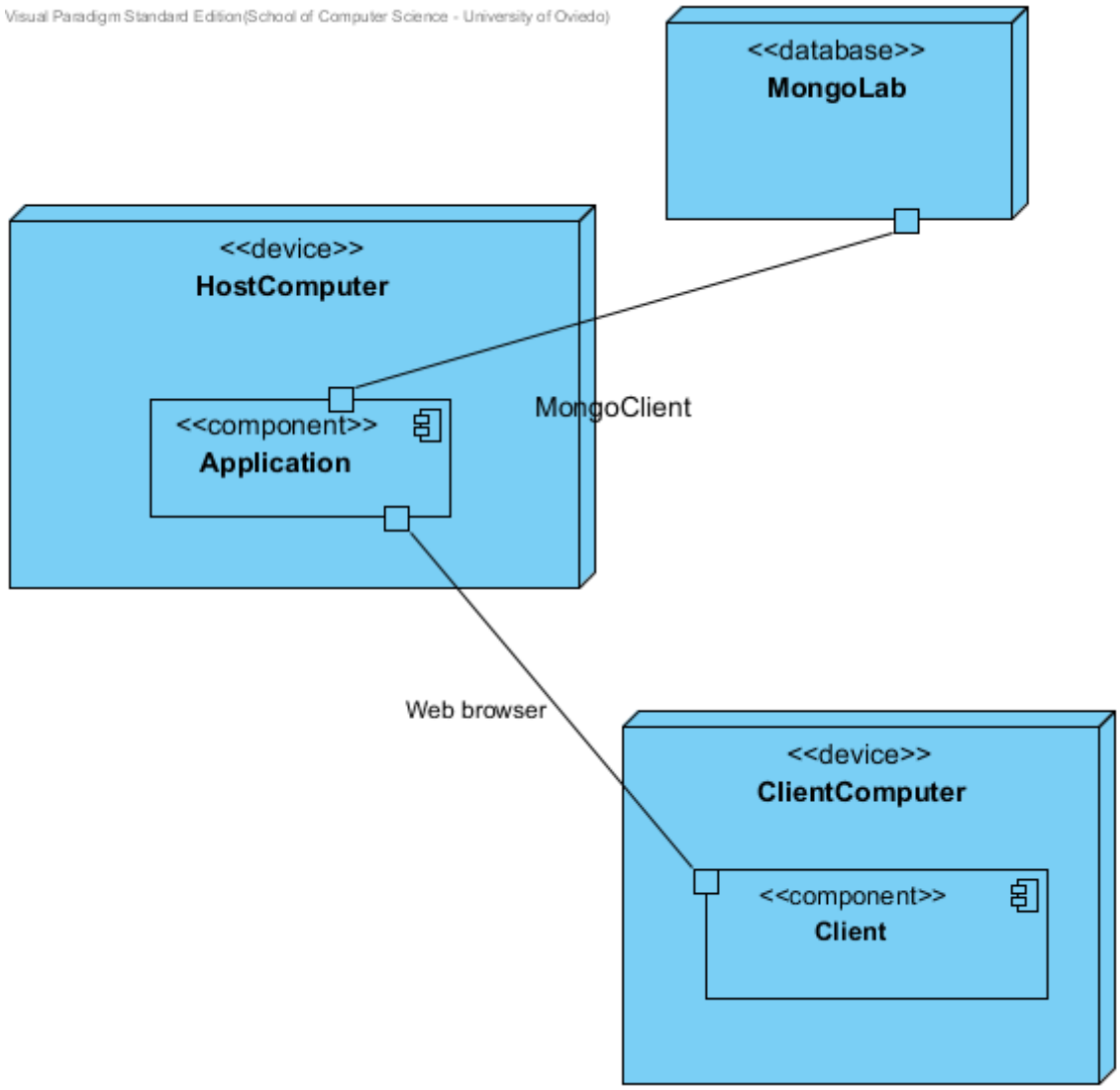
Game


Model
<<component>>
Statistics
<<component>>
Square
<<component>>
Position
<<component>>
User
<<component>>
Board
<<component>>
Player
<<component>>
Game
GUI
DB
<<component>>
GameWindow
<<component>>
NewUserWindow
<<component>>
MongoManager

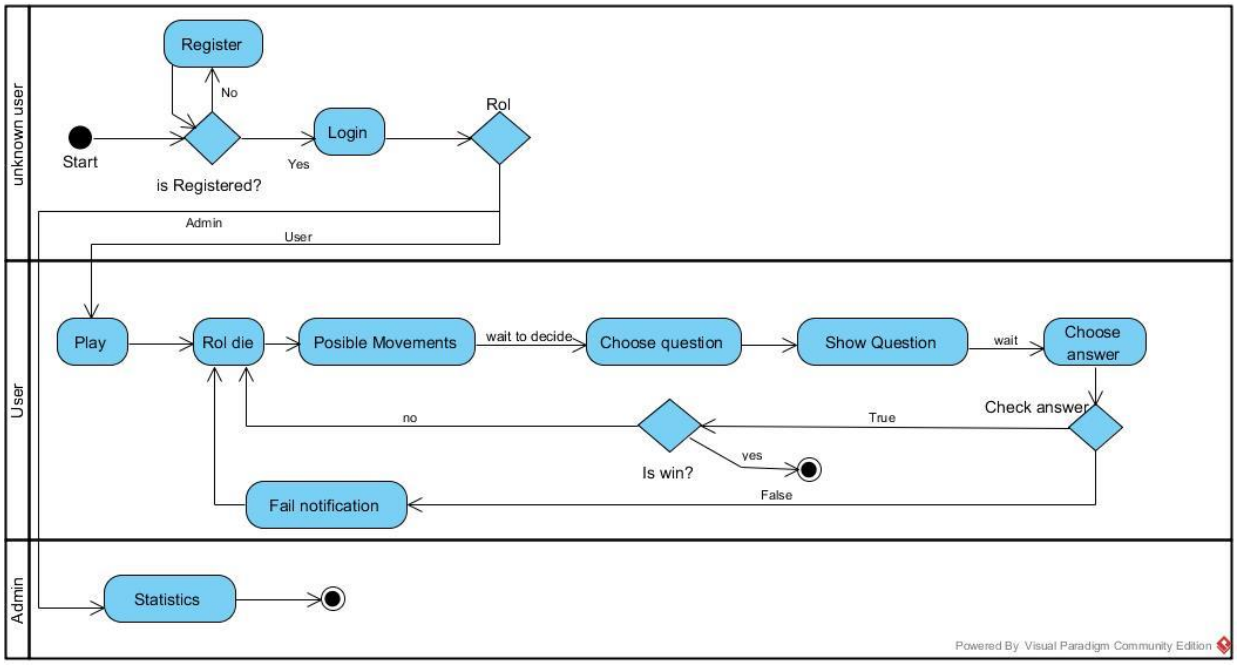
Web


Visual Paradigm Standard Edition(School of Computer Science - University of Oviedo)
Model
<<component>>
Statistics
<<component>>
Square
<<component>>
Position
<<component>>
User
<<component>>
Board
<<component>>
Player
<<component>>
Game
DB
Controllers
<<component>>
Application
<<component>>
MongoManager
Views
.scala.html files

Deployment view



BPM


unknown user
Register
No
Start
is Registered?
Yes
Login
Rol
Admin
User
User
Play
Rol die
Posible Movements
wait to decide
Choose question
Show Question
wait
Choose answer
Check answer
no
True
Is win?
yes
False
Fail notification
Admin
Statistics
Powered By Visual Paradigm Community Edition

User manual

This is a web application that works locally but can be uploaded to a server; since it is based on the original Trivial game everybody will be able to learn how to use it in almost no time.

The most important thing the user must know before starting to use or application is that it is composed by 5 different pages:

Initial page

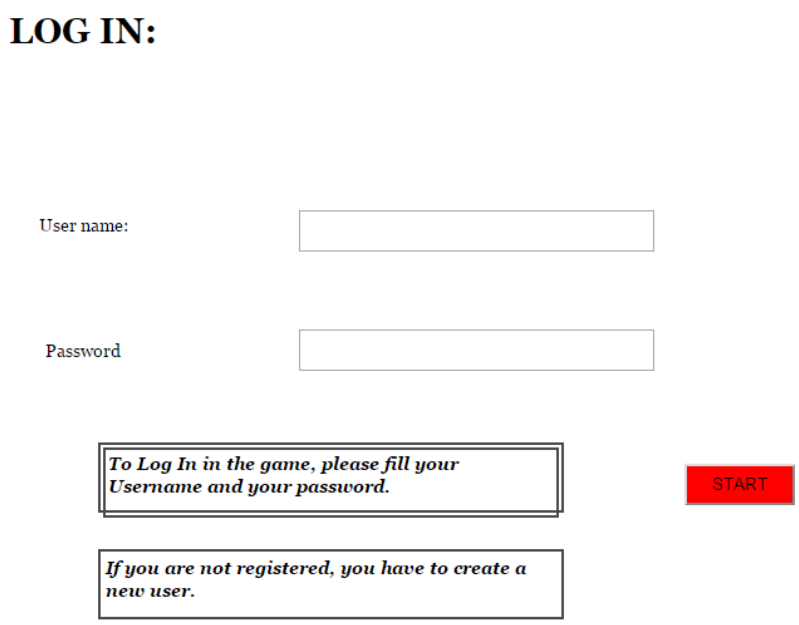
The first window that appears when the application is ran. In this window, you can change the background color with the buttons on the left, go to the page where you can register as a user or go to the login page.

Create a new user

If you want to create a new user, you must provide some personal data like your username, your email and a password. All this information is required to create the user and will be saved to the database after you click in the register button.

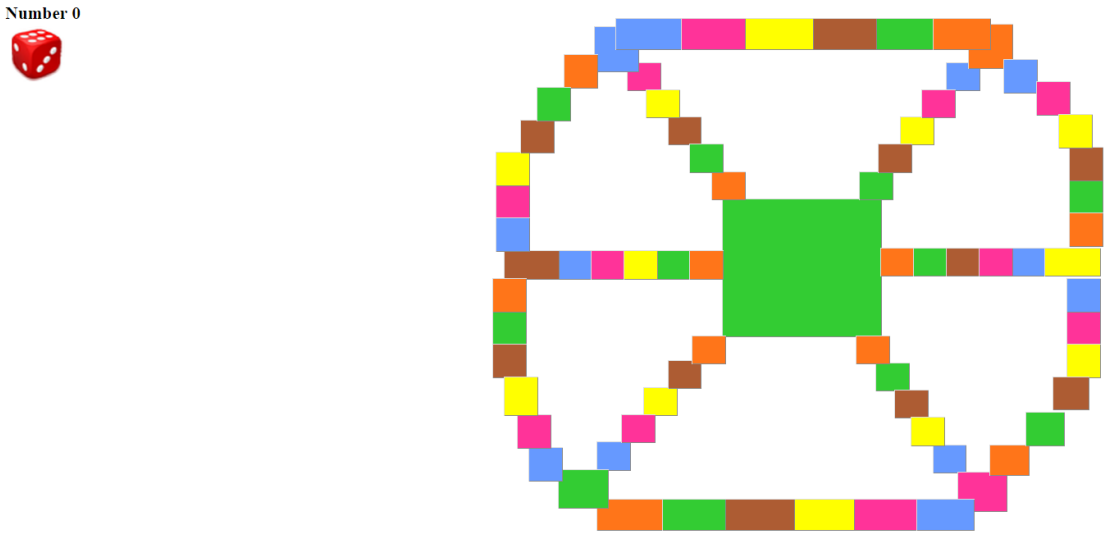
Login

In this window you can login with your username and password you previously created in the register page. This information is going to be checked when you click in the start button, after that if the information is correct the play page is going to be shown.



Play

This is the main window of the application. Here is the board that should show where the player is. To start the game, the dice has to be pressed and the player has to decide which category he/she prefers. Depending on the category selected, one question or another will be retrieved from the database and asked to the player.



Statistics

In this page the user will be able to see a small piece of information as a table. The information shown is the username, number of games played, number of answers and number of correct ones for all players.

See statistics

User name	Games played	Questions Answered	Questions Correct
user1	5	100	24
user2	2	15	7
user3	4	50	8

System Manual

System requirements

In order the application to work properly it is necessary to have installed a Java version greater than or equal to Java 5.

Project development information

This project is based on the MVC architecture. For this reason you will be able to find the following folders:

Model

This package contains all the model of the game, since it has not suffered any change with respect the previous implementation all the information about it can be consulted in the page written below, accessing to the "*Logic*" section.

https://github.com/Arquisoft/Trivial_i1b/blob/web/Documentation/Documentation_Triviali1b_V2.0.pdf

Views

This package the different pages can be accessed in our application:

- **Board**
  HTML page that will show the board of the game. The board is constructed by using buttons to represent each cell.
- **Error**
  This page is the one used in case some error occurs while logging or creating a new user. It only contains an error message.
- **Initial**
  Window that will be shown the first. It provides access to the *login* and the *newUser* window. There are also 4 buttons in order to change the color of the screen, the implementation of this behavior us made by using a JavaScript function that changes the background depending on the id of the button clicked.
- **Login**
  This window is used by the player to tell the game who is the user that will play. In case the user and password entered are correct the board, otherwise the error page is shown.
- **NewUser**
  This page allows creating a new user by asking for a name, email address and password.
- **Statistics**
  It is only for the admin, who can access by means of the login window. This page shows the statistics of the players looking in the database for the needed information.

Controllers

Here you will be able to find all the classes necessary for coordinate the model and the views. Inside it, the class Application is stored, which takes the role of being the main controller of the application.

About the database, in this deliverable, it has been decided to move it to the MongoLab server instead of using in the host machine.

Not completed tasks

Because of some failures during the implementation of the deliverable there are some parts of the project that are not working.

The representation of the game status is not working, although you can see the board and throw the die there is not any piece indicating the position of the player or allowing him to select where to go.

Another thing that we have not been able to implement is the representation of the statistics in the window built for that purpose.

On the other hand you can log into the database and create a new user without any problem and the database s working properly.

For this deliverable we have two representations of the board, one is made with buttons and it would be use in case there were future upgrades, but as it is too big we have already made another one with a image of the board in order to show it how it would be.

Future upgrades

The application is a prototype, meaning that there are some things yet to be implemented.

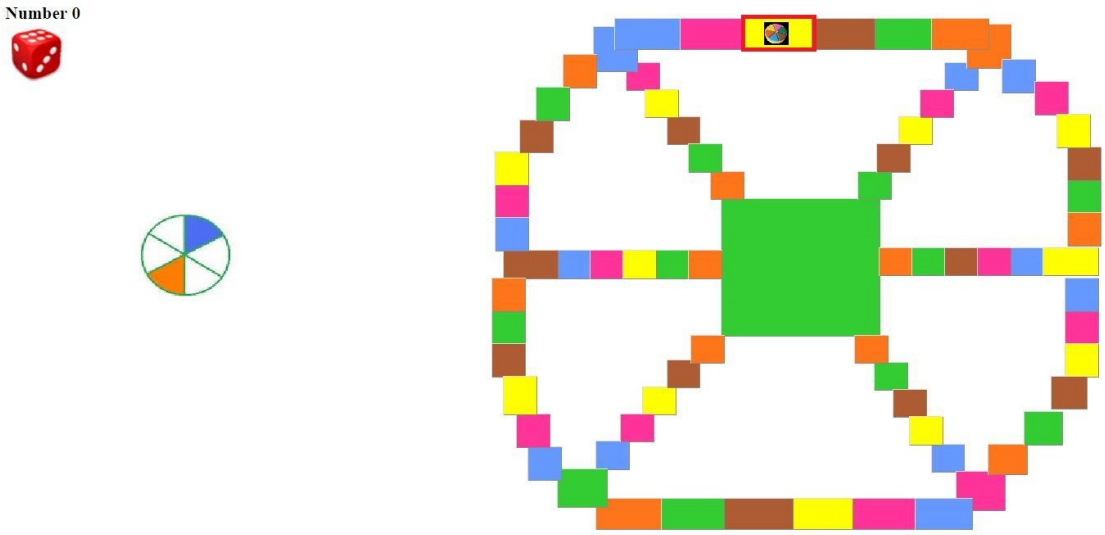
Some of the ideas we have for the future of the application include:

Completely joining the views with the API

Make all the views use correctly the methods from the API. To do this we would have to solve some problems that arise when using Play, and also learn JavaScript more in-depth in order to represent correctly the information on the views.

Improving the view for playing with the board

Improving the view which represents the board so it represents more information and allows the user to have a better experience playing the game (position of the player, wedges, etc.)



Multiplayer mode

This could be implemented with JMS (*Java Message Service*) using queues (peer to peer communication), or using topics (publish/subscribe). The latter would probably be the best option according to the needs of our application.

This system would allow different players to publish or subscribe to different channels, we would make the different clients check for updates on the channel with a certain frequency (or even better with an event-based update system) and update the view to represent the information of all players.

This system is not only very convenient for our needs but also performs very well as it can use asynchronous messages to communicate the different clients.