



FOOTBALL MANAGER

League Manager C11 – DPOO 2022/2023

Andrea Ballester Griful

Leonardo Rubén Edenak Chouev

Pol Guarch Bosom

Jan Piñol Castuera

Joan Tarragó Pina

Oriol Rebordosa Cots

TABLE OF CONTENTS

1.	RESUM DE LES ESPECIFICACIONS DEL PROJECTE	2
1.	DISSENY DE LA INTERFÍCIE GRÀFICA.....	3
2.	DIAGRAMA DE CLASSES.....	32
3.	METODOLOGIA DE DESENVOLUPAMENT	37
4.	DEDICACIÓ	40
5.	CONCLUSIONS.....	41
6.	BIBLIOGRAFIA CONSULTADA	42

1. RESUM DE LES ESPECIFICACIONS DEL PROJECTE

Football Manager és un programa que permet gestionar la competició de diferents lligues de futbol. En aquest, s'esperen diferents funcionalitats i comportaments que es detallaran a continuació.

Primer de tot, cal saber que existeixen dos tipus d'usuaris diferents que podran iniciar sessió al programa, administradors (*admin*) i jugadors (*players*). En funció de qui accedeixi al sistema, podrà dur a terme unes funcions o unes altres.

L'administrador és el que té el control total del programa, això vol dir que podrà executar les següents funcions:

- ❖ **Crear un nou equip:** l'administrador haurà de carregar al sistema un fitxer JSON amb la informació necessària per a la creació d'un equip. Seguidament, el sistema haurà de validar que les dades introduïdes son vàlides. Un cop les dades siguin acceptades, es procedirà a crear l'equip i seguidament es crearan els comptes per als nous jugadors, generant una contrasenya aleatòria per a cadascun d'ells. Finalment, es mostrarà a l'administrador un llistat amb els nous comptes creats (DNI i contrasenya).
- ❖ **Crear lligues:** durant el procés de creació d'una lliga, l'administrador haurà d'introduir el nom que li donarà juntament amb la data i l'hora d'inici. Després de que el sistema verifiqui que la informació és correcte, es mostrarà a l'administrador un llistat amb tots els equips disponibles per a que seleccioni quins vol incloure a la nova lliga. Un cop creada, el sistema procedirà a crear el calendari.
- ❖ **Gestionar els equips i les lligues:** aquesta funcionalitat permet a l'administrador del programa eliminar els equips i/o lligues que desitgi.
- ❖ **Visualitzar totes les lligues:** accedint a aquesta funcionalitat, l'administrador podrà visualitzar un llistat amb totes les lligues existents. Per cada lliga, es mostrarà el nom, el nombre d'equips participants i la jornada actual d'aquesta. Quan es faci clic sobre el nom d'una lliga, aleshores es veurà a la pantalla una taula amb la informació de tots els equips que pertanyen a la lliga seleccionada. A més, també es podrà visualitzar un gràfic que mostrarà com ha evolucionat la puntuació que han obtingut els equips en les diferents jornades. Finalment, l'usuari podrà visualitzar la informació dels jugadors d'un equip concret fent clic sobre el que l'interessi dels que es troben a la taula. Les dades que veurà seran les personals de cada jugador que conformen l'equip (nom, correu electrònic, dorsal, DNI i número de telèfon).
- ❖ **Visualitzar tots els partits:** permet veure en temps real el desenvolupament de tots els partits. Quan s'accedeixi a aquesta opció, es veurà el llistat de tots els partits en curs i es podrà triar quin es vol veure amb més detall.

A més, l'administrador també disposarà de la opció de tancar sessió.

Els jugadors en canvi, disposen de menys funcionalitats i més limitades.

- ❖ **Veure les lligues:** en el que fa a veure les lligues, les visualitzarà de la mateixa manera que ho fa l'administrador però amb la diferència de que només se li mostraran aquells en les que hi ha un equip que ell hi estigui participant.
- ❖ **Veure els partits:** de la mateixa manera que l'administrador visualitza els partits, ho farà també el jugador amb la limitació de que només tindrà accés a veure aquells de les lligues on ell hi participa.

Igual que l'administrador, l'usuari podrà tancar la seva sessió de la mateixa manera que també tindrà l'opció de canviar la contrasenya i d'eliminar el compte.

2. DISSENY DE LA INTERFÍCIE GRÀFICA

Per al disseny de les pantalles del projecte, hem fet ús del programa *Figma* el qual ens permet dissenyar i crear la interfície gràfica. Un cop ja la teníem tota creada, vam crear pensar quins components SWING i quins layouts necessitaríem per a plasmar el que havíem creat en el *Figma* en el nostre projecte.

Tot seguit, es mostraran totes les pantalles que hem implementat juntament amb un esquema on es poden apreciar les decisions de disseny preses, els components SWING i els layouts utilitzats.

Per a dissenyar la interfície gràfica hem fet ús d'una gran varietat de components que ens ofereix Swing com poden ser; JPanel, JScrollPane, ImageIcon, JButton, JLabel entre d'altres. Però com volíem aconseguir una interfície gràfic molt impactant, hem investigat les variacions i modificacions que ens ofereixen aquests components.

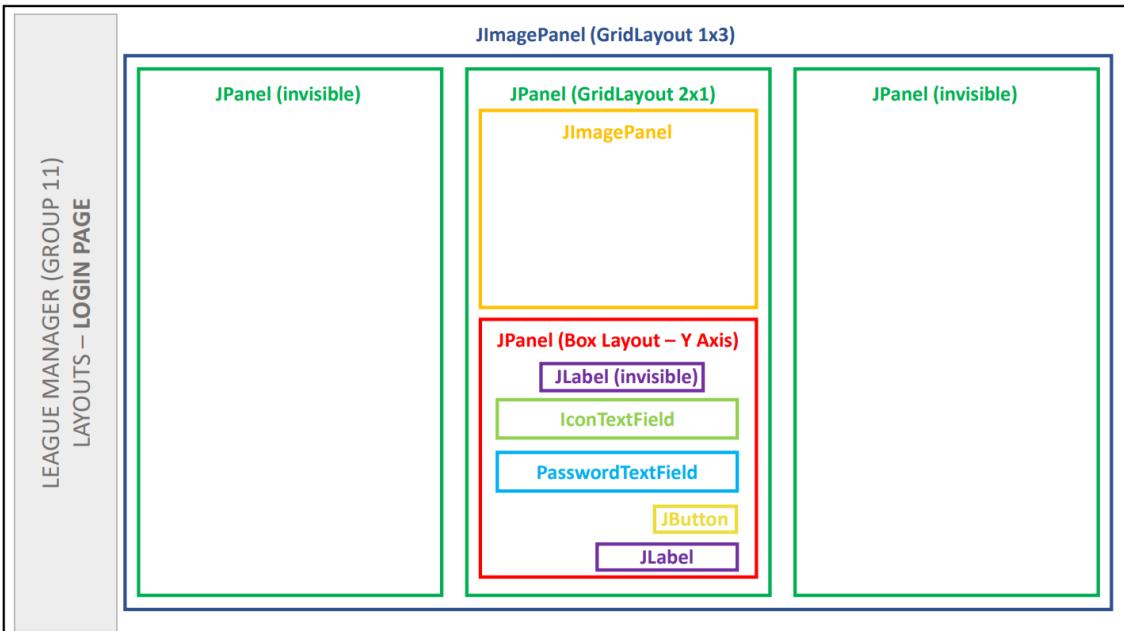
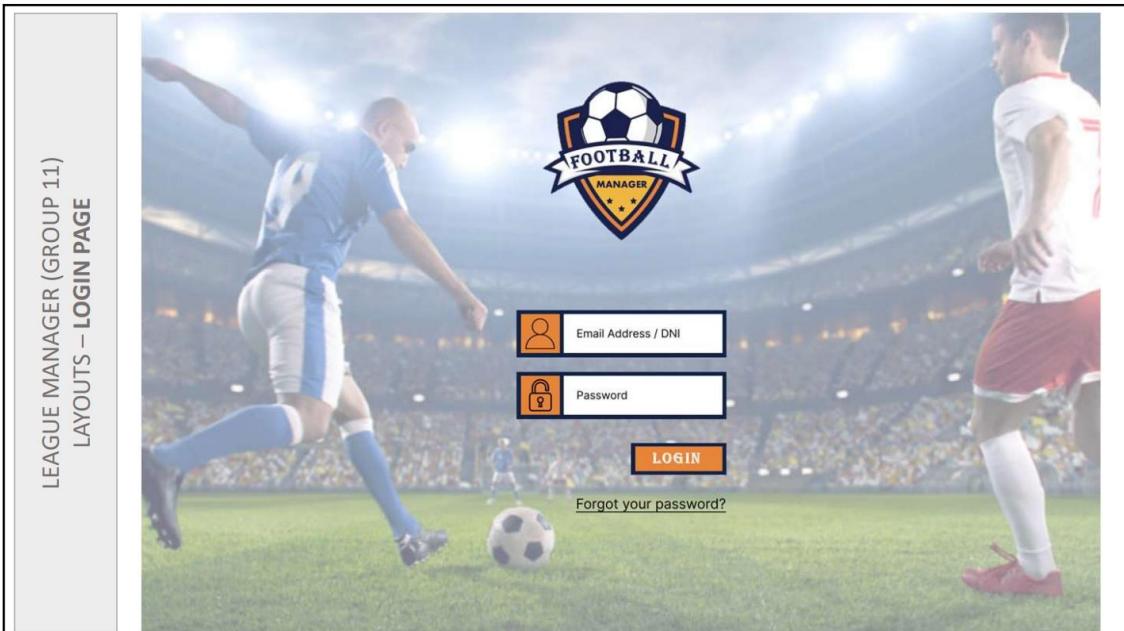
Hem dividit les vistes en 3 grups principals; **Main Pages**, **Admin Pages** i **User Pages**.

MAIN PAGES

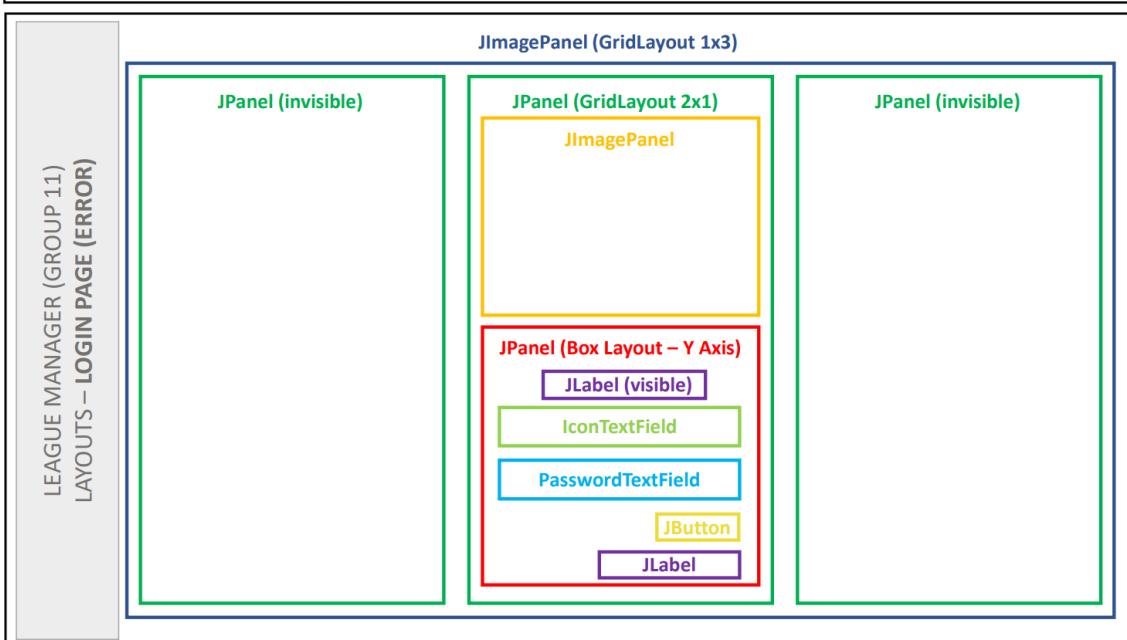
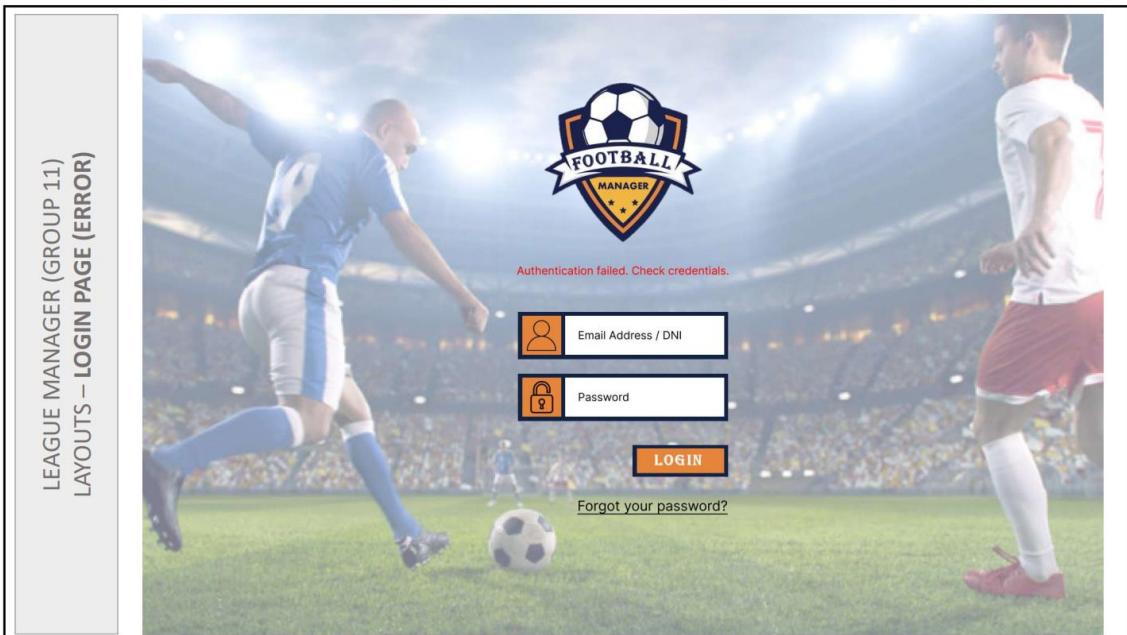
Les Main Pages son les vistes que visualitza l'usuari quan inicia el programa, inicia sessió o reestableix la contrasenya.

Hem creat un cardLayout al centre d'aquesta pantalla per poder anar alternant entre el formulari d'inici de sessió on s'emplen els camps necessaris per iniciar sessió al programa i també mostra en un JLabel els possibles errors que poden sorgir (nom d'usuari no existent, contrasenya invàlida...), i entre la pantalla amb el formulari de restablir la contrasenya, també amb els possibles errors mostrats en pantalla.

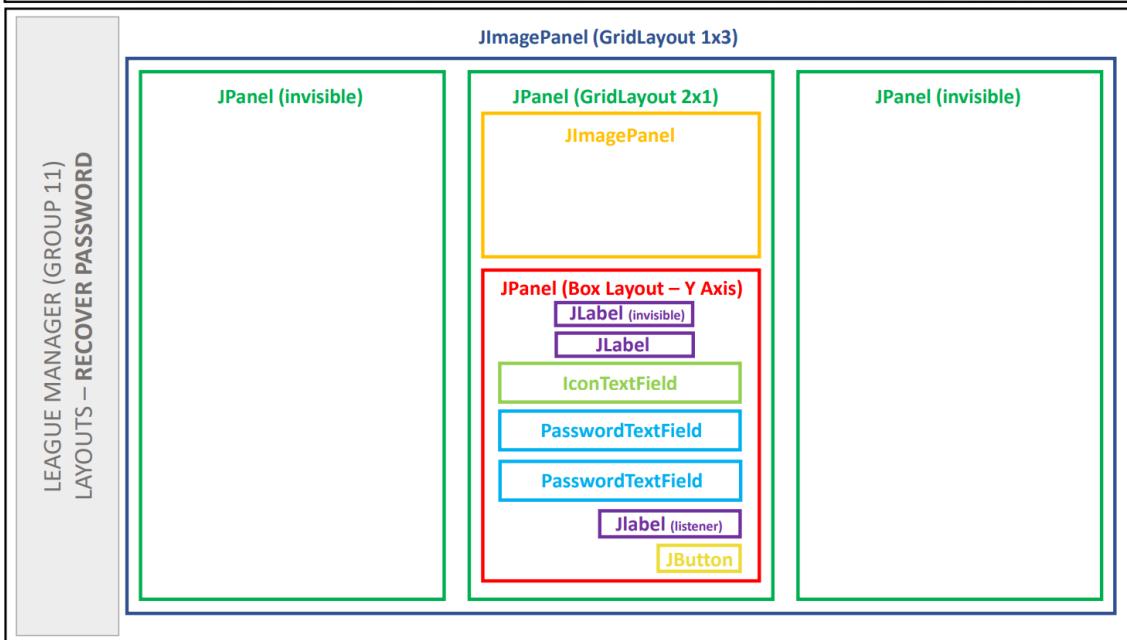
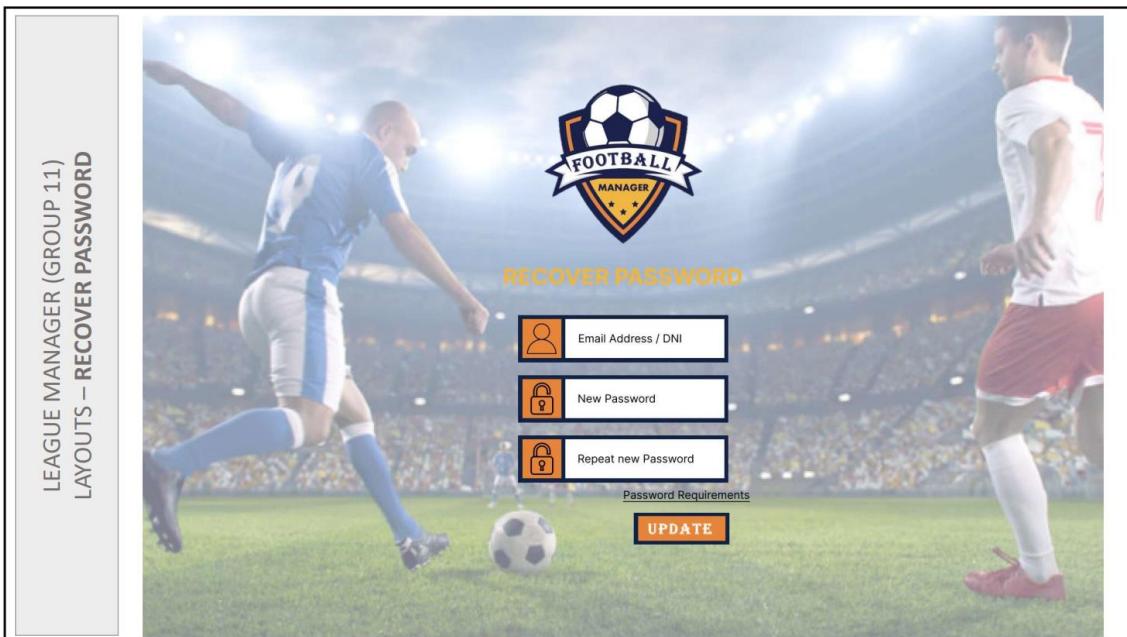
LOGIN PAGE



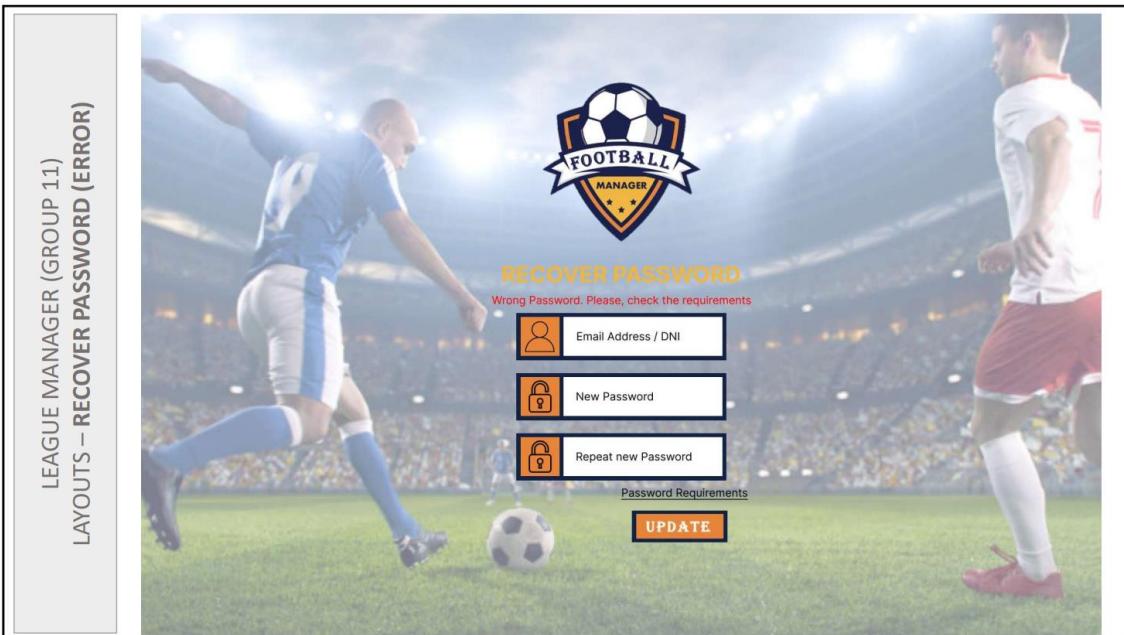
LOGIN PAGE (ERROR)



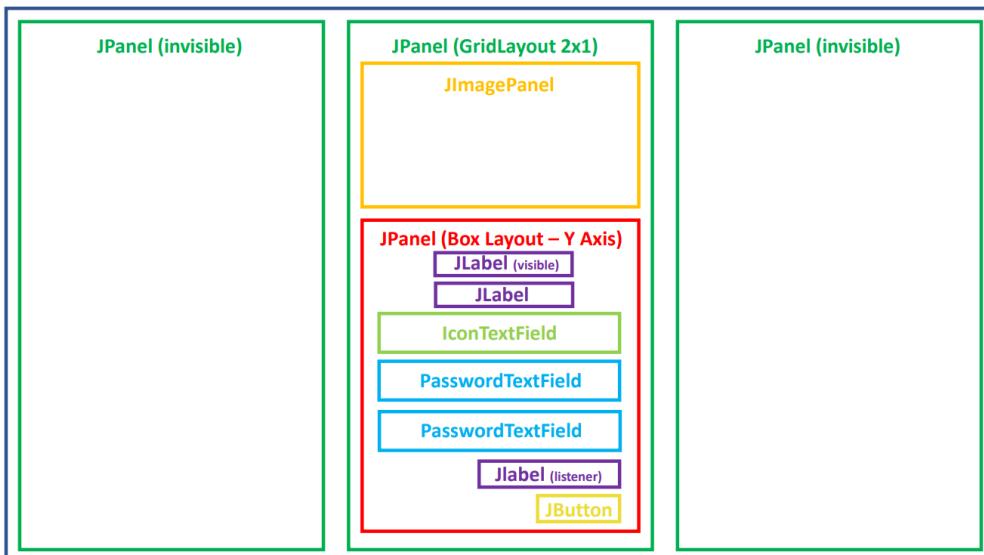
RECOVER PASSWORD



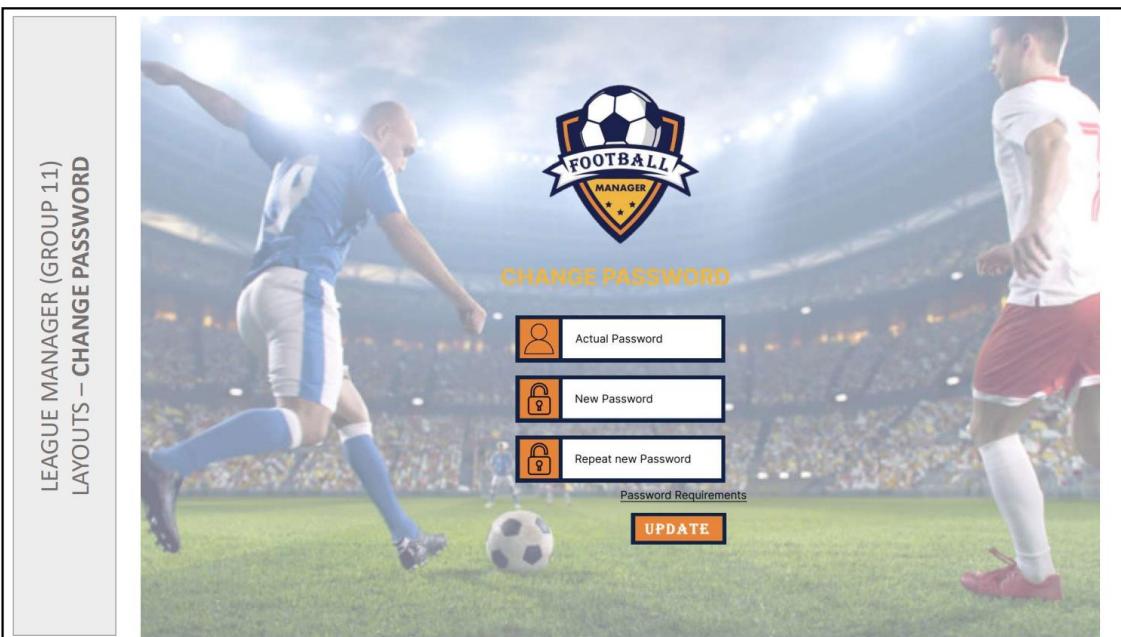
RECOVER PASSWORD (ERROR)



LEAGUE MANAGER (GROUP 11) LAYOUTS – RECOVER PASSWORD (ERROR)

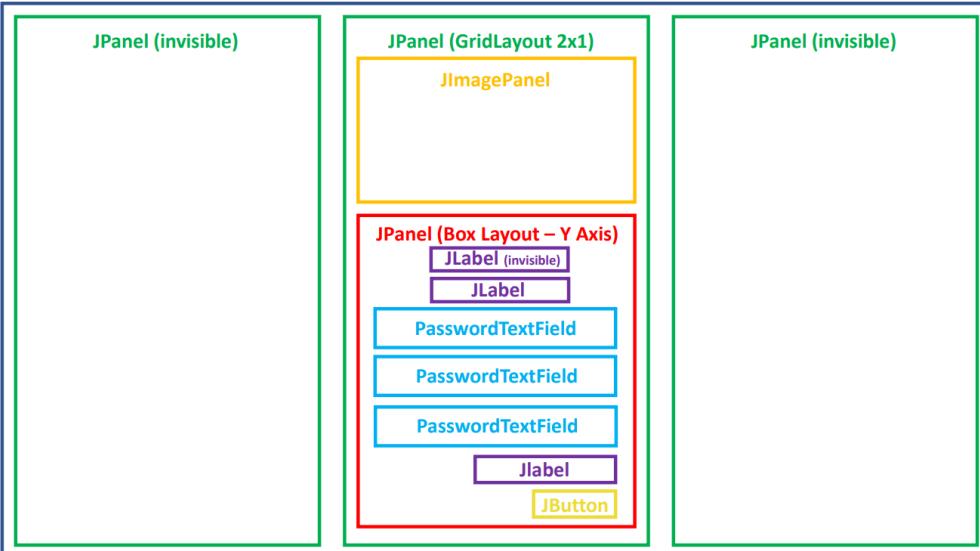


CHANGE PASSWORD

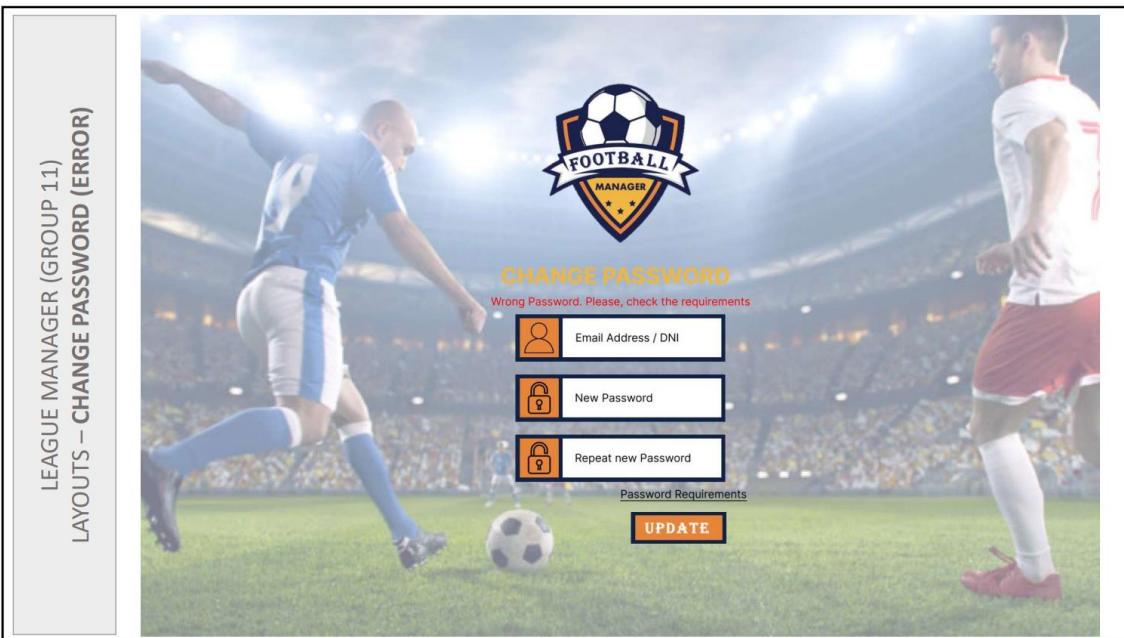


LEAGUE MANAGER (GROUP 11)
LAYOUTS – CHANGE PASSWORD

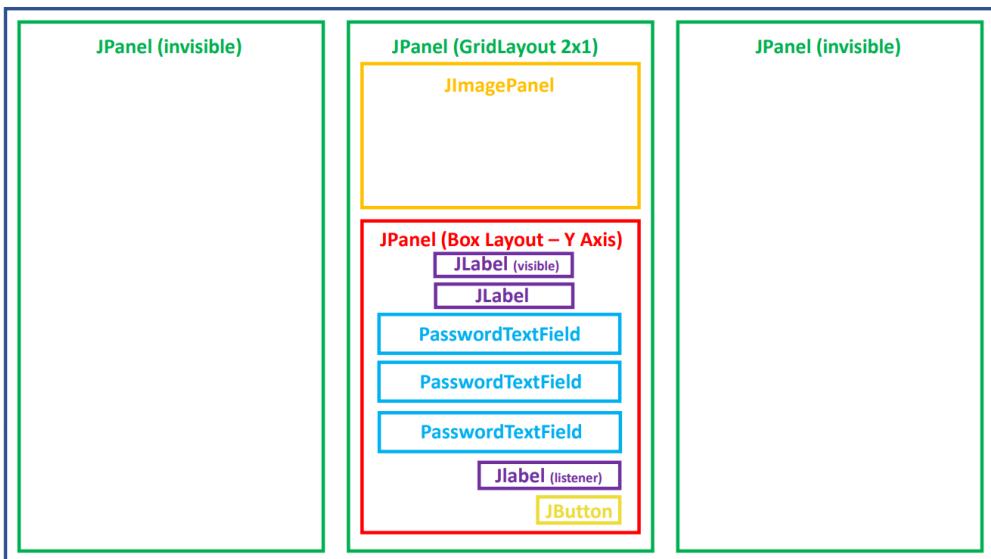
JImagePanel (GridLayout 1x3)



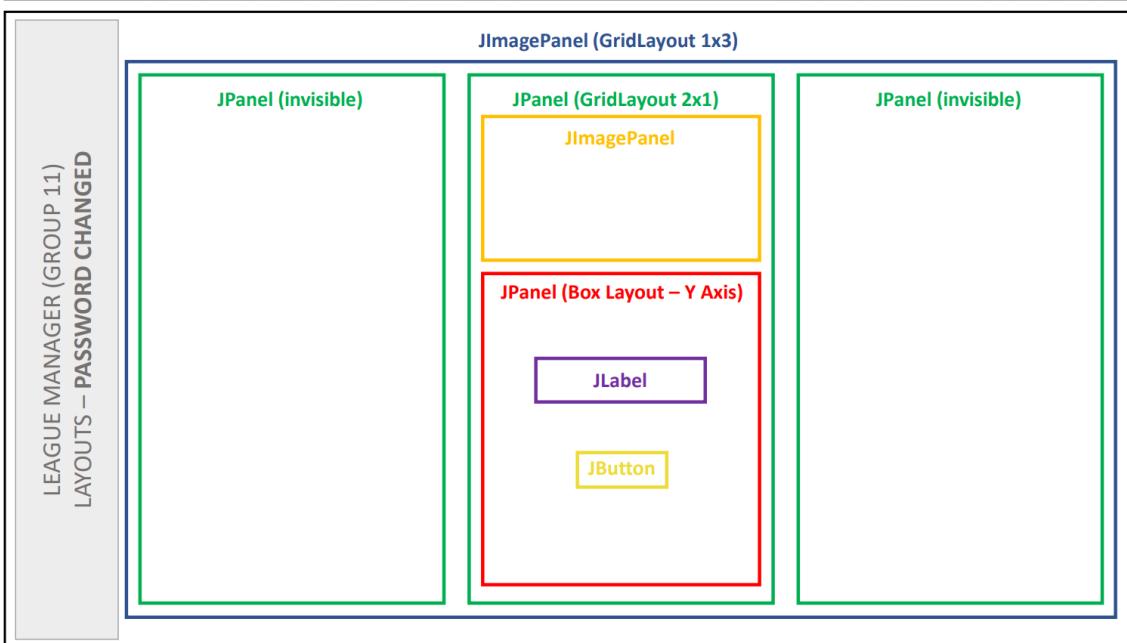
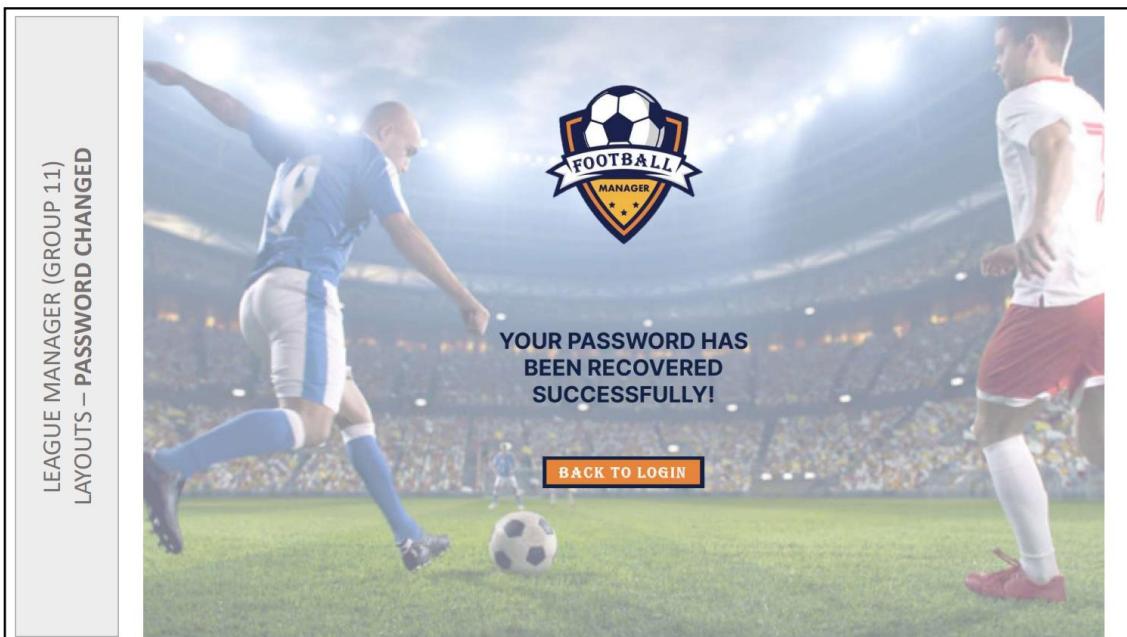
CHANGE PASSWORD (ERROR)



LEAGUE MANAGER (GROUP 11) LAYOUTS – CHANGE PASSWORD (ERROR)



PASSWORD CHANGED



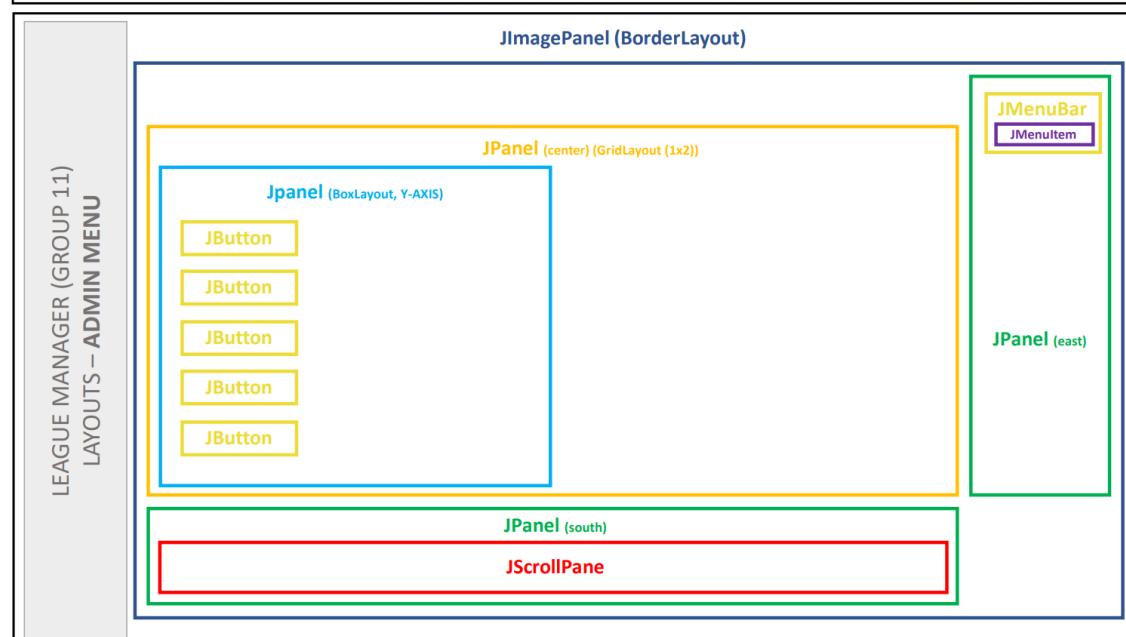
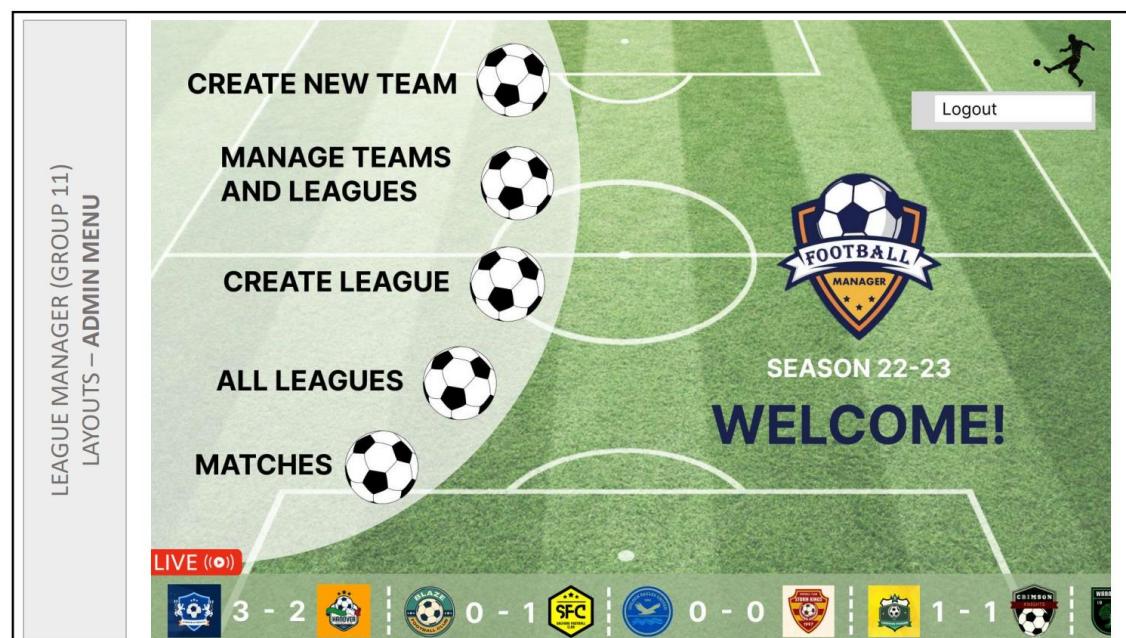
ADMIN PAGES

Aquesta pantalla s'encarrega de anar canviant entre la pantalla del menu principal del admin o bé entre la pantalla d'alguna de les funcions exclusives del admin. També té un CardLayout que permet anar canviant quan es requereixi.

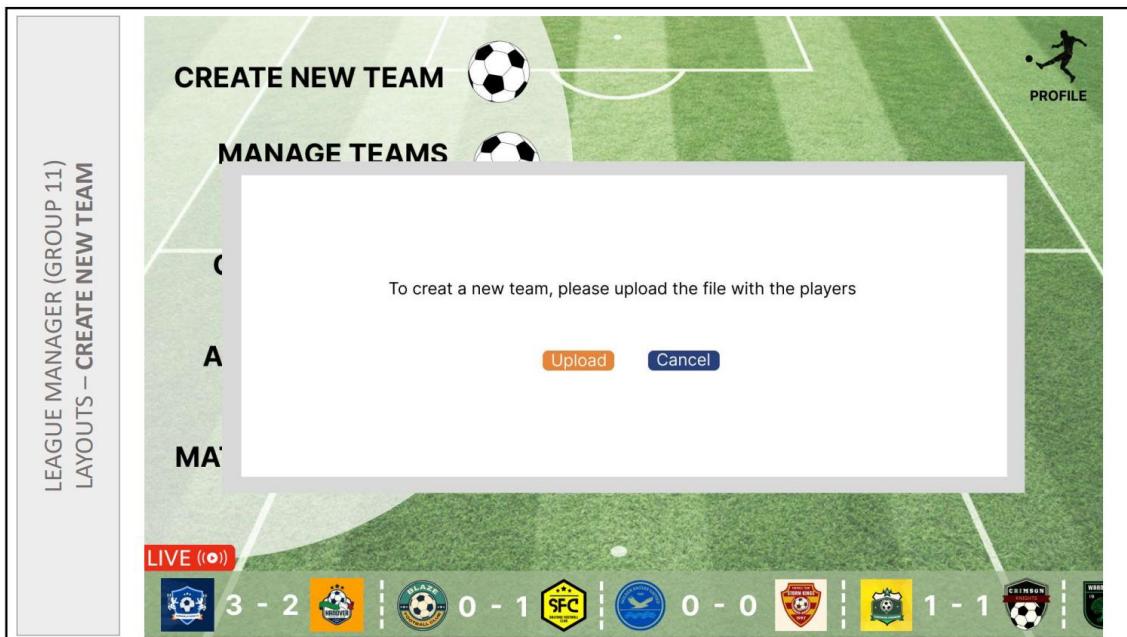
La pantalla del menú és única, excepte alguns moments on es mostra algun JDialog per sobre però queda en segon pla.

La pantalla de les opcions del admin va canviant depenent de quina opció s'ha premut. Per tant aquesta també tindrà un CardLayout per canviar entre les pantalles de manage teams and leagues, crear una lliga, veure totes les lligues, veure les estadístiques, veure un equip, veure els partits i veure un partit en directe.

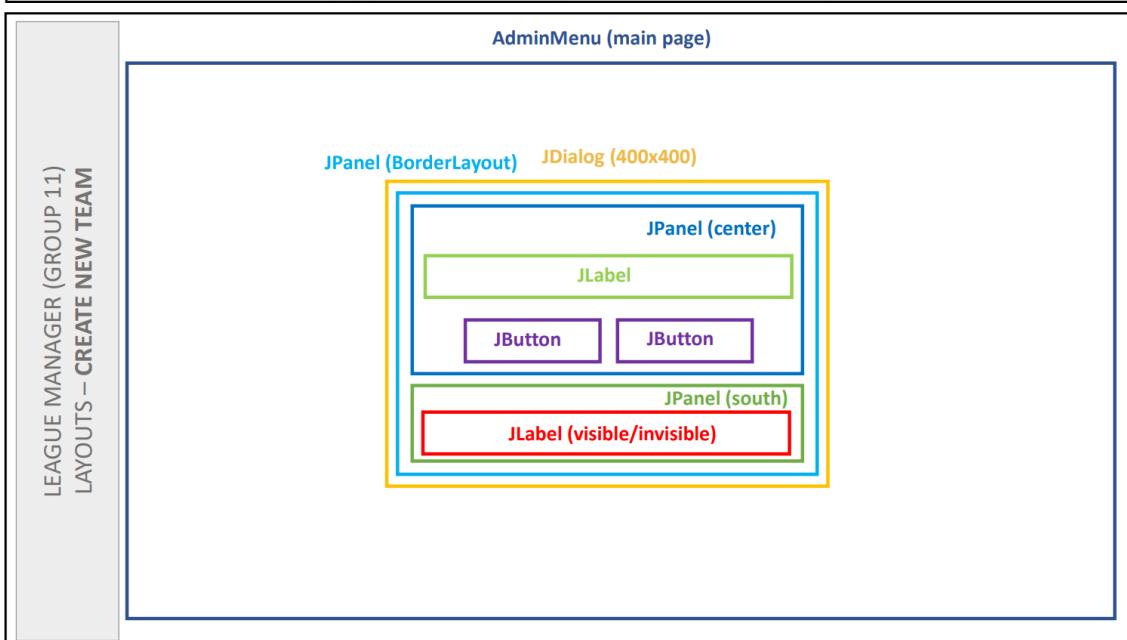
ADMIN MENU



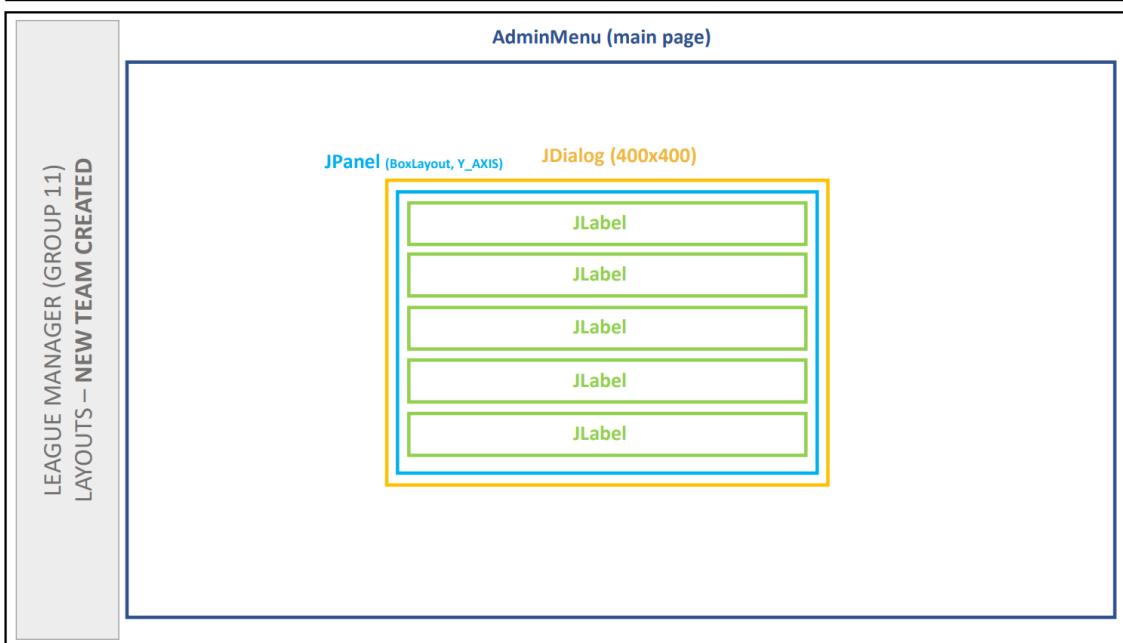
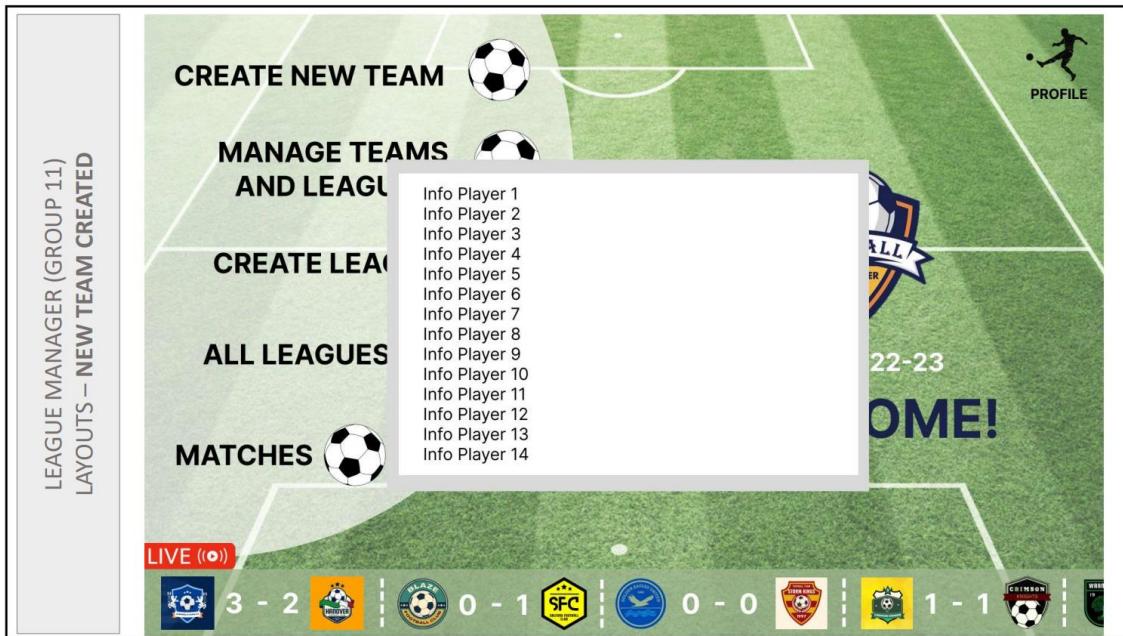
CREATE NEW TEAM



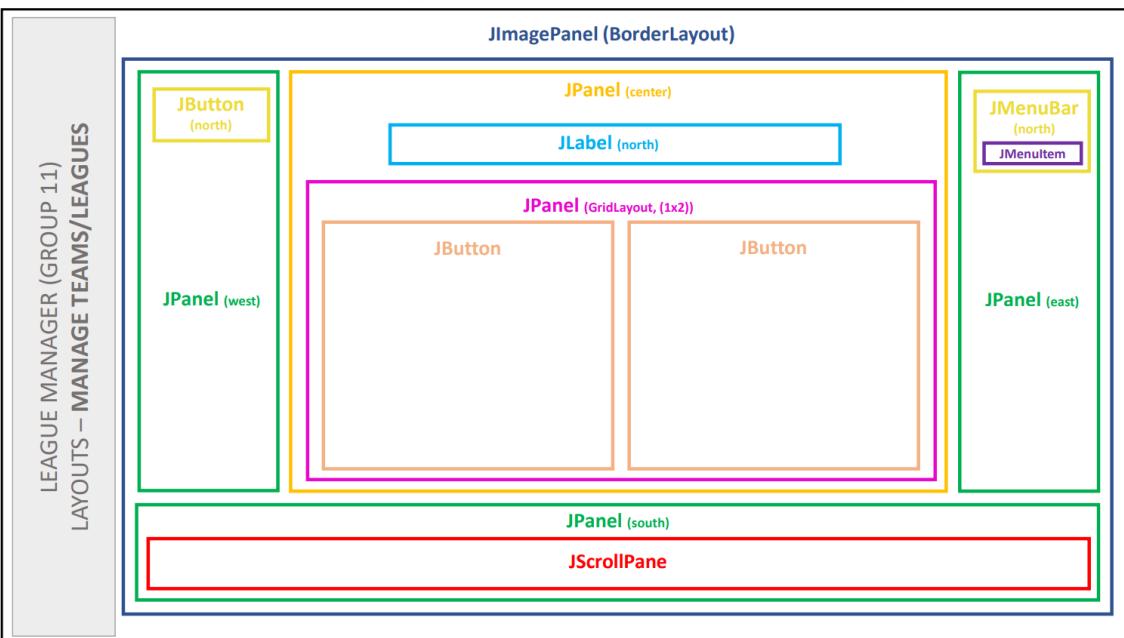
AdminMenu (main page)



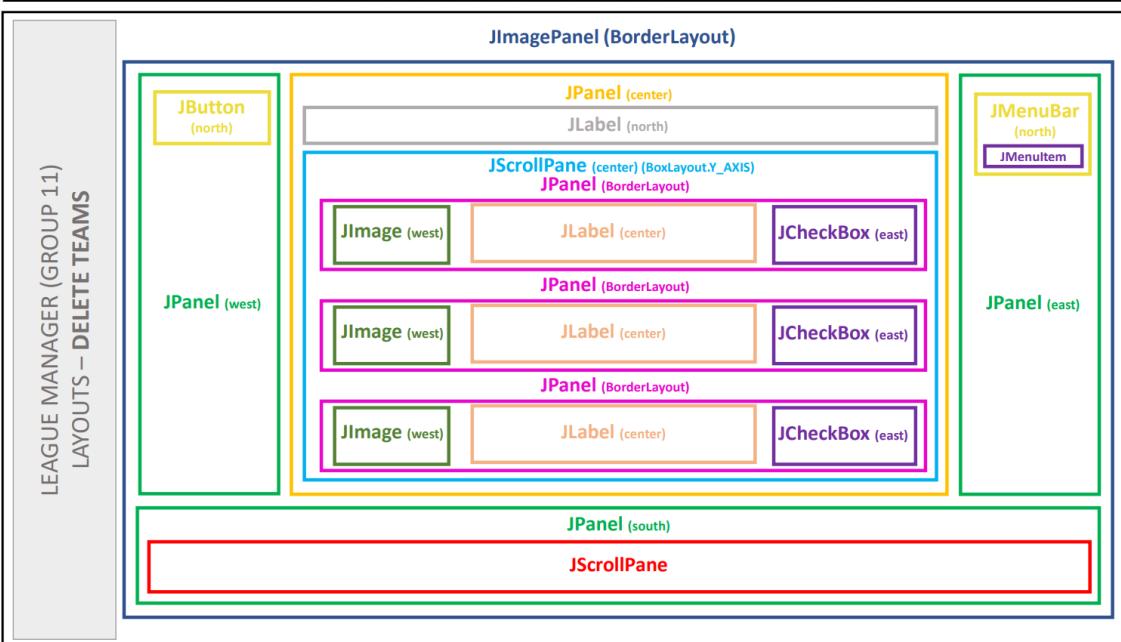
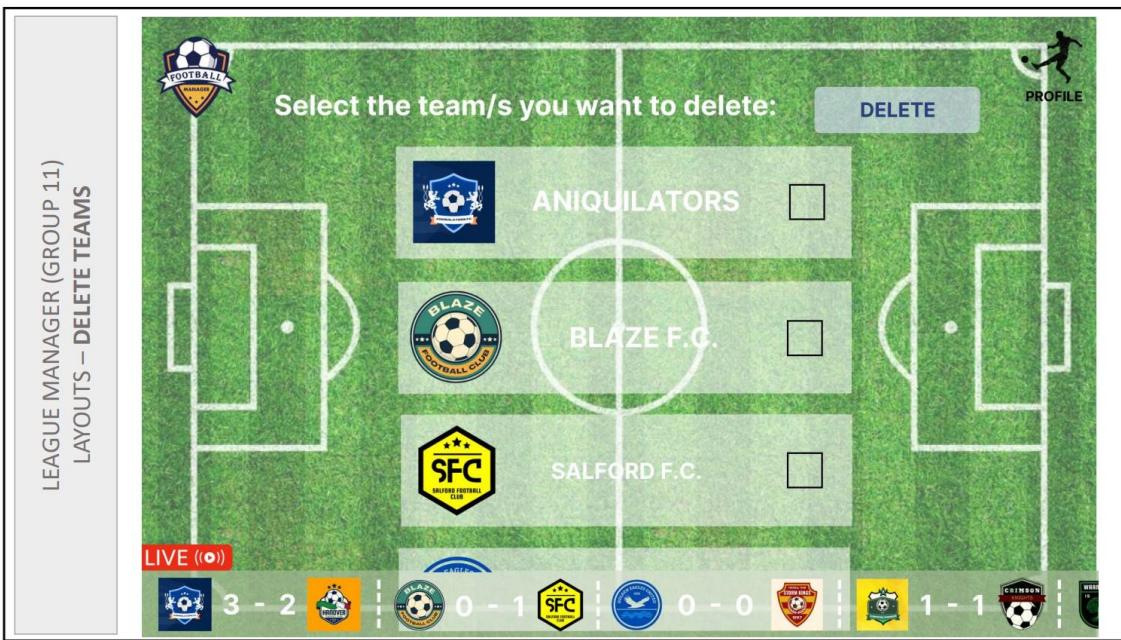
NEW TEAM CREATED



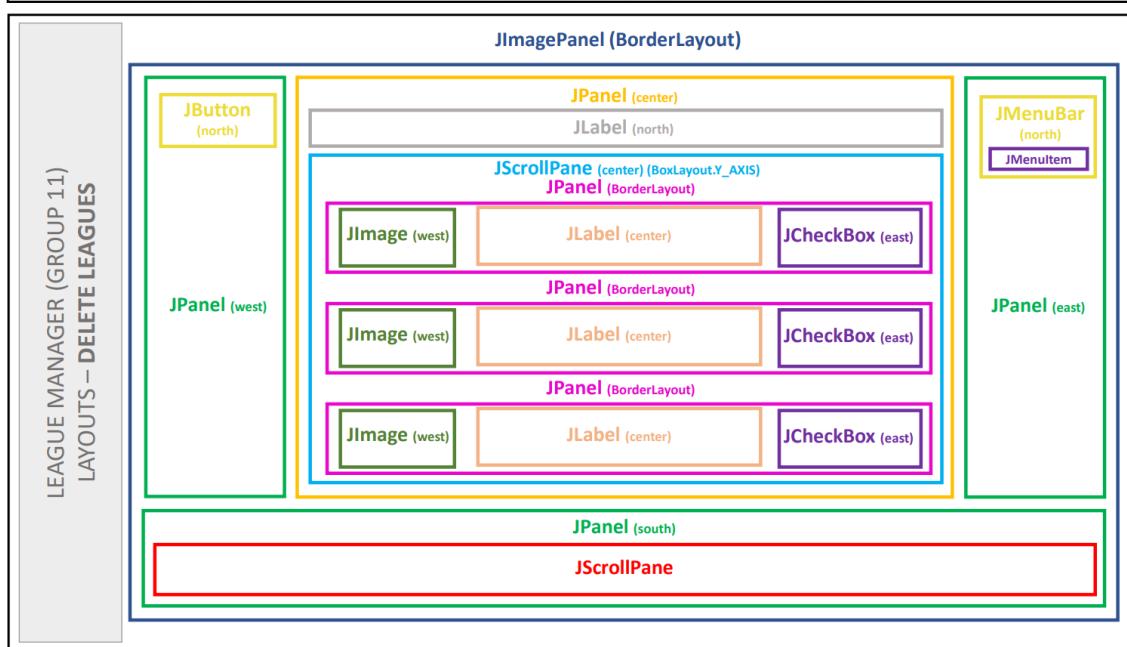
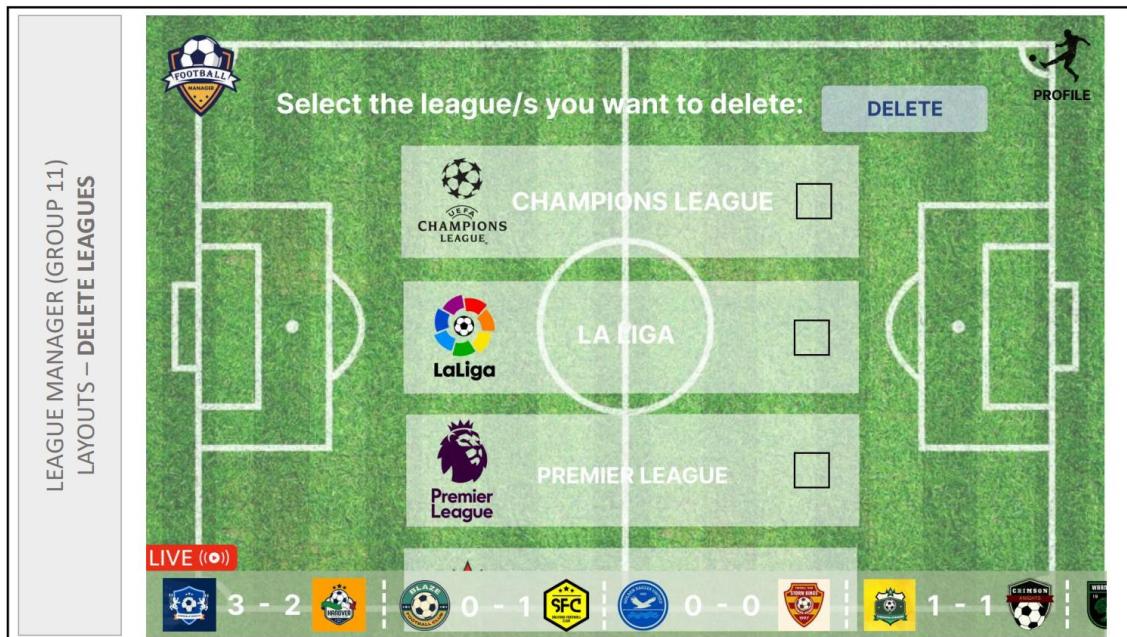
MANAGE TEAMS/LEAGUES



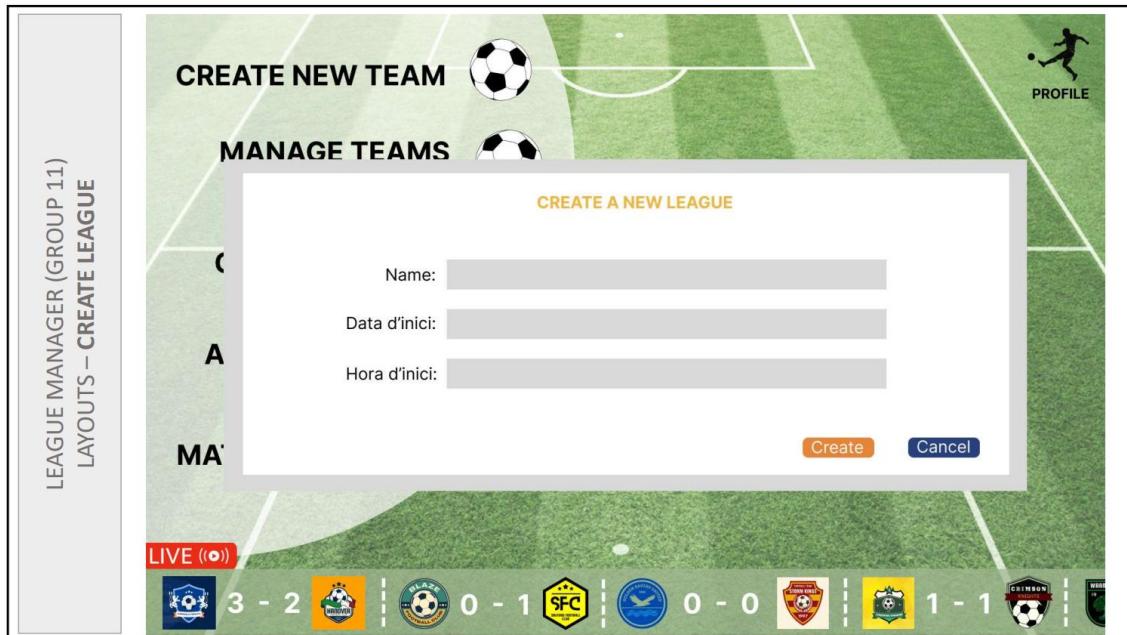
DELETE TEAMS



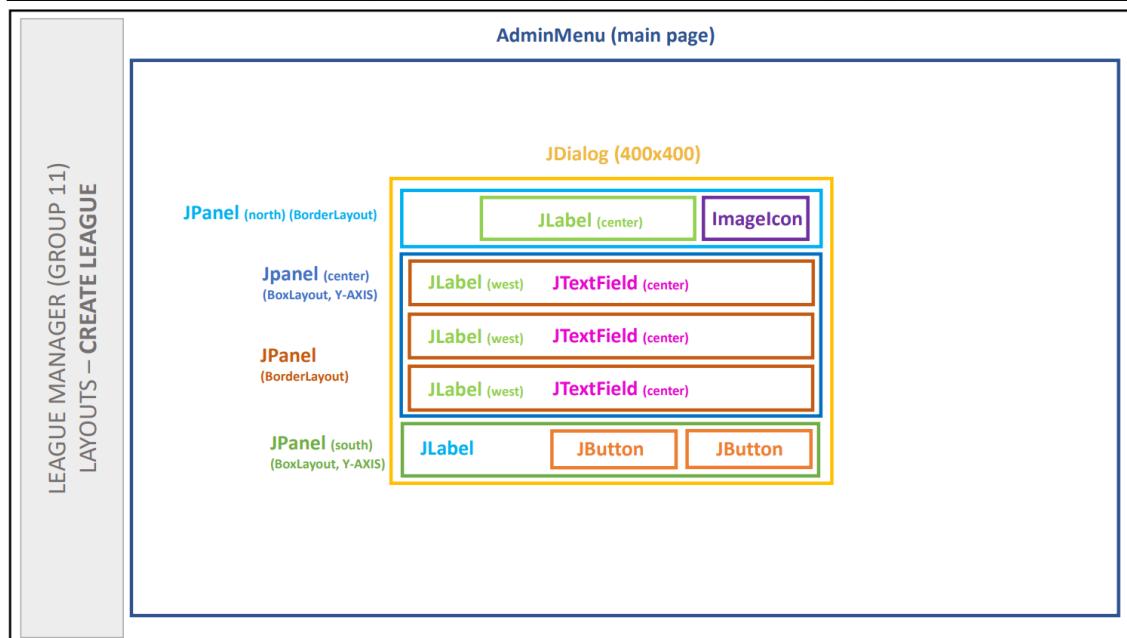
DELETE LEAGUES



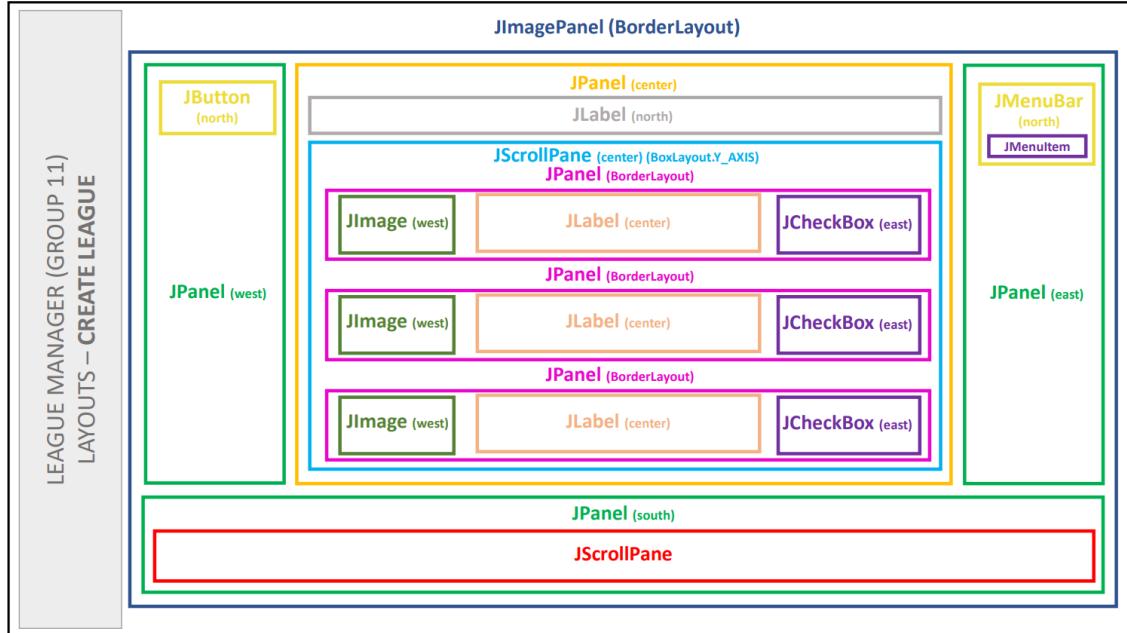
CREATE LEAGUE (FORM)



AdminMenu (main page)



CREATE LEAGUE (SELECTION)



ALL LEAGUES

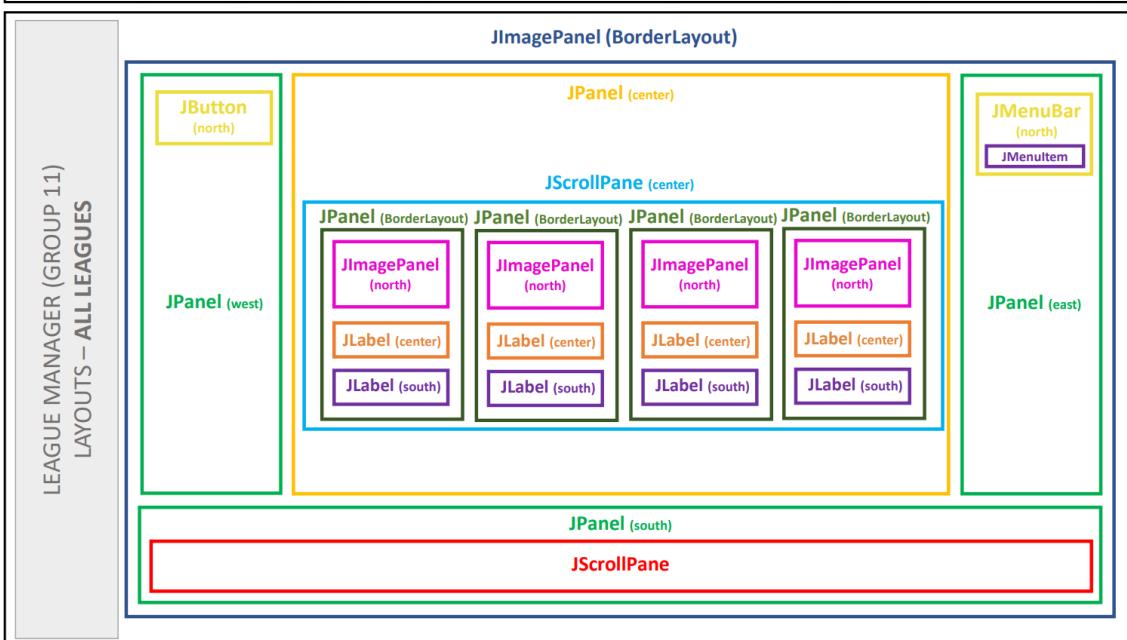


TABLE STATISTICS

LEAGUE MANAGER (GROUP 11)
LAYOUTS – TABLE STATISTICS

LEAGUE TABLE		LEAGUE STATISTICS					
RANK	NAME	NUM. PLAYERS	P	W	D	L	PTS
1	F.C. BARCELONA	31	27	23	2	2	71
2	REAL MADRID	32	27	18	5	4	59
3	ATLÉTICO DE MADRID	28	27	16	6	5	54
4	REAL SOCIEDAD	29	27	14	6	7	48
5	BETIS	25	27	13	6	8	45
6	VILLAREAL	25	27	13	5	9	44
7	ATHLETIC DE BILBAO	27	27	10	7	10	37
8	RAYO VALLECANO	30	27	9	10	8	37

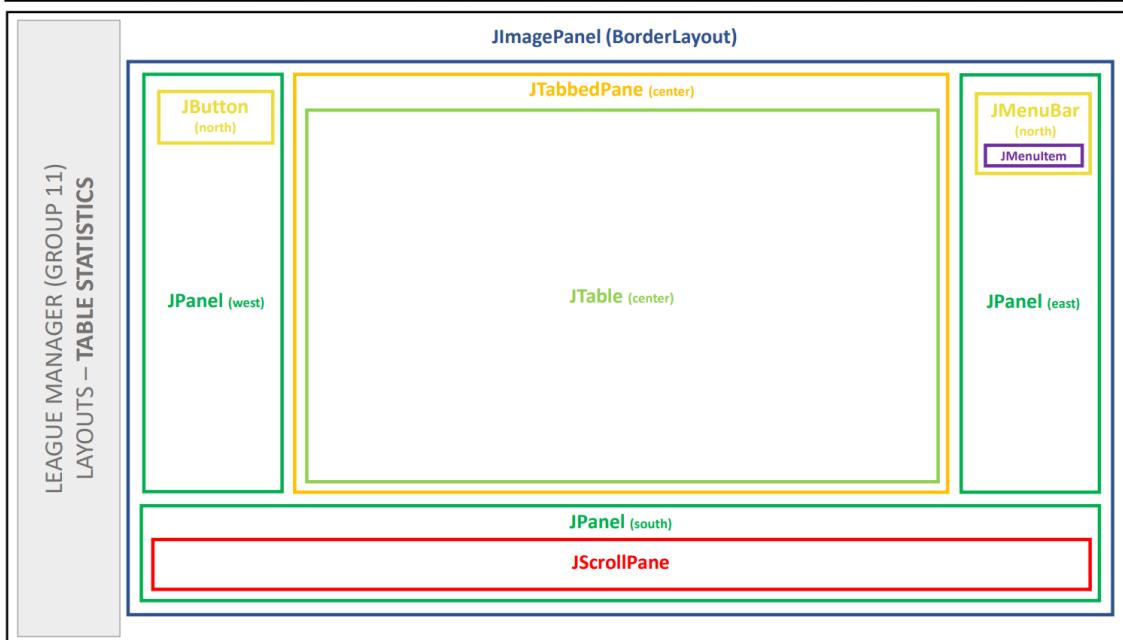
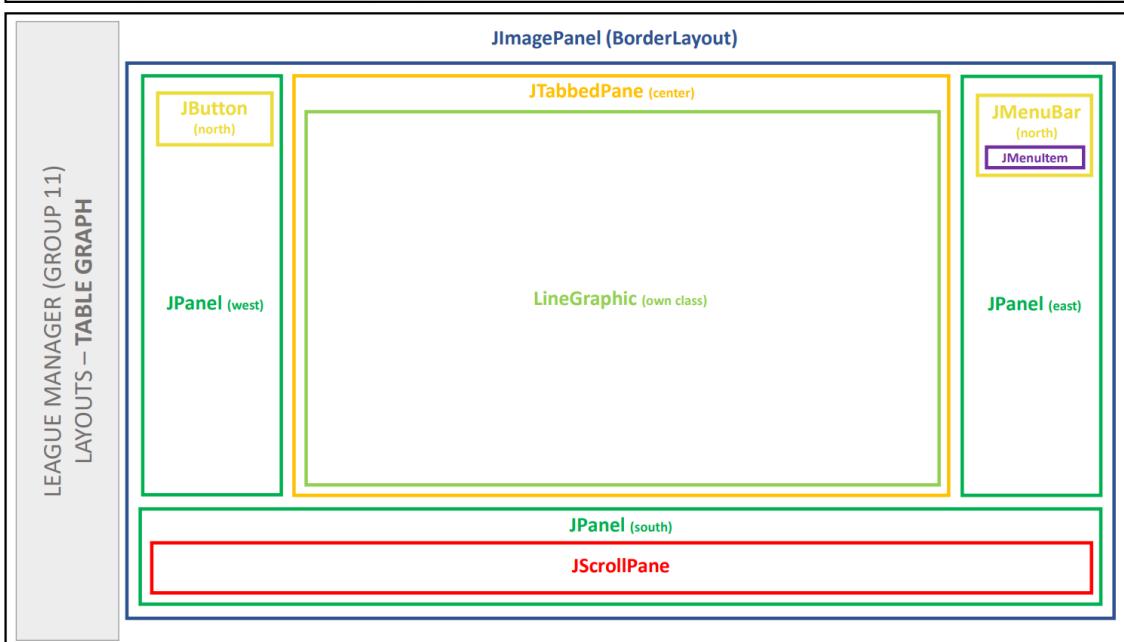
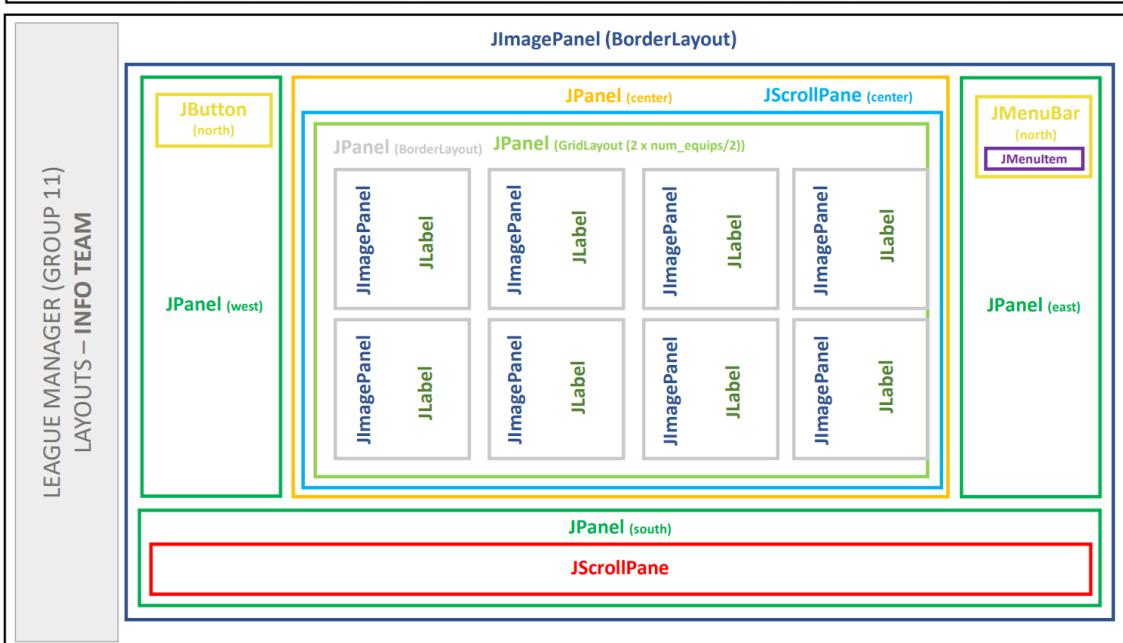


TABLE GRAPH



INFO TEAM



LIST MATCHES

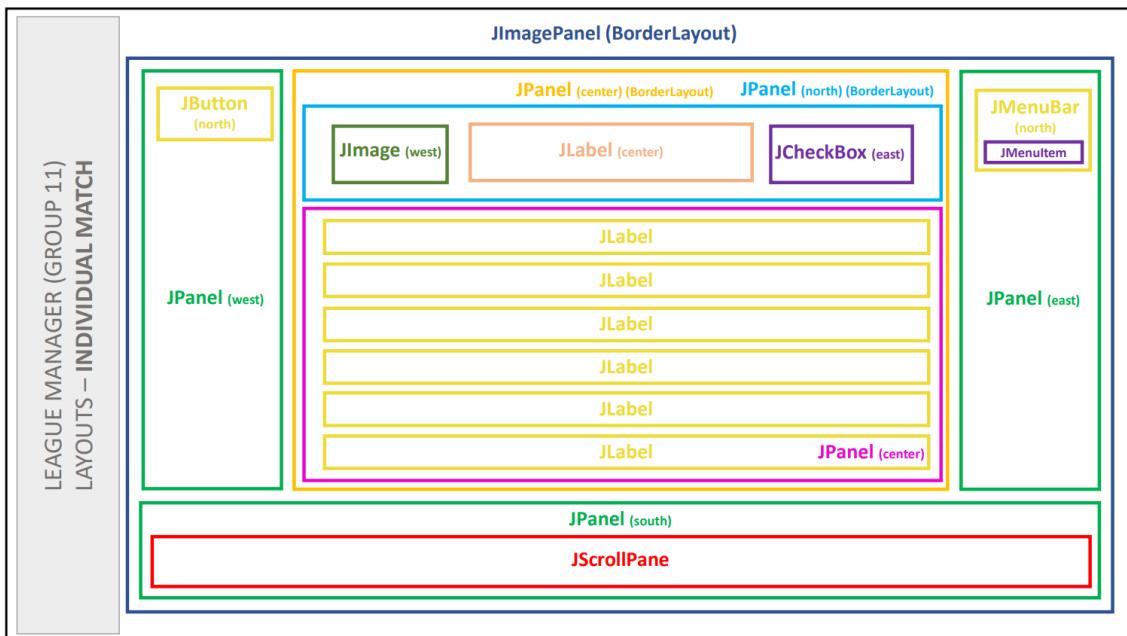
LEAGUE MANAGER (GROUP 11)
LAYOUTS – LIST MATCHES

JImagePanel (BorderLayout)

JPanels and Components:

- North Panel (Yellow border):** JButton (north)
- West Panel (Green border):** JPanel (west)
- Center Panel (Yellow border):**
 - JPanels (center):
 - JScrollPane (center) (BoxLayout.Y_AXIS)
 - JPanels (BorderLayout):
 - JImage (west)
 - JLabel (center)
 - JImage (west)
 - JPanels (BorderLayout):
 - JImage (west)
 - JLabel (center)
 - JImage (west)
 - JPanels (BorderLayout):
 - JImage (west)
 - JLabel (center)
 - JImage (west)
 - JPanels (BorderLayout):
 - JImage (west)
 - JLabel (center)
 - JImage (west)
- East Panel (Green border):** JPanel (east)
- South Panel (Red border):** JPanel (south)
 - JScrollPane
- Other Components:** JMenuBar (north) with JMenuItem, and a small profile icon in the top right corner.

INDIVIDUAL MATCH

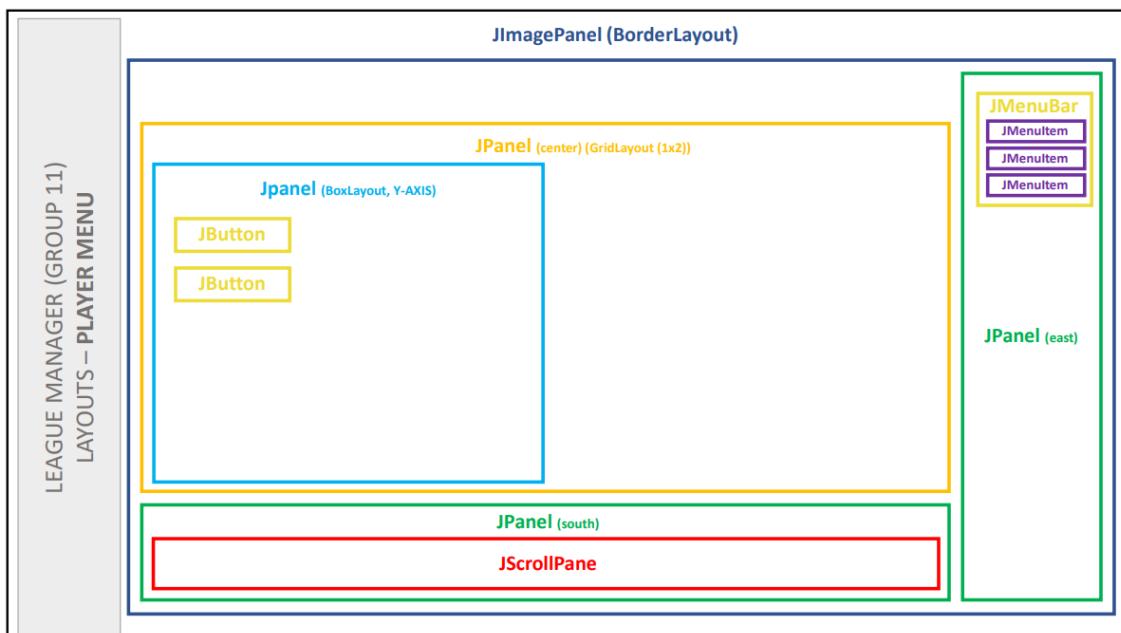


PLAYER PAGES

PLAYER MENU

Aquesta pantalla s'encarrega de anar canviant entre la pantalla del menu principal o bé entre la pantalla d'alguna de les funcions. També té un CardLayout que permet anar canviant entre elles quan es requereixi.

La pantalla del menú és única. La pantalla de les opcions del user va canviant depenent de quina opció s'ha premut. Per tant aquesta també tindrà un CardLayout per canviar entre les pantalles veure les lligues, estadístiques, equips, partits i detall de partit en directe.



VIEW LEAGUES

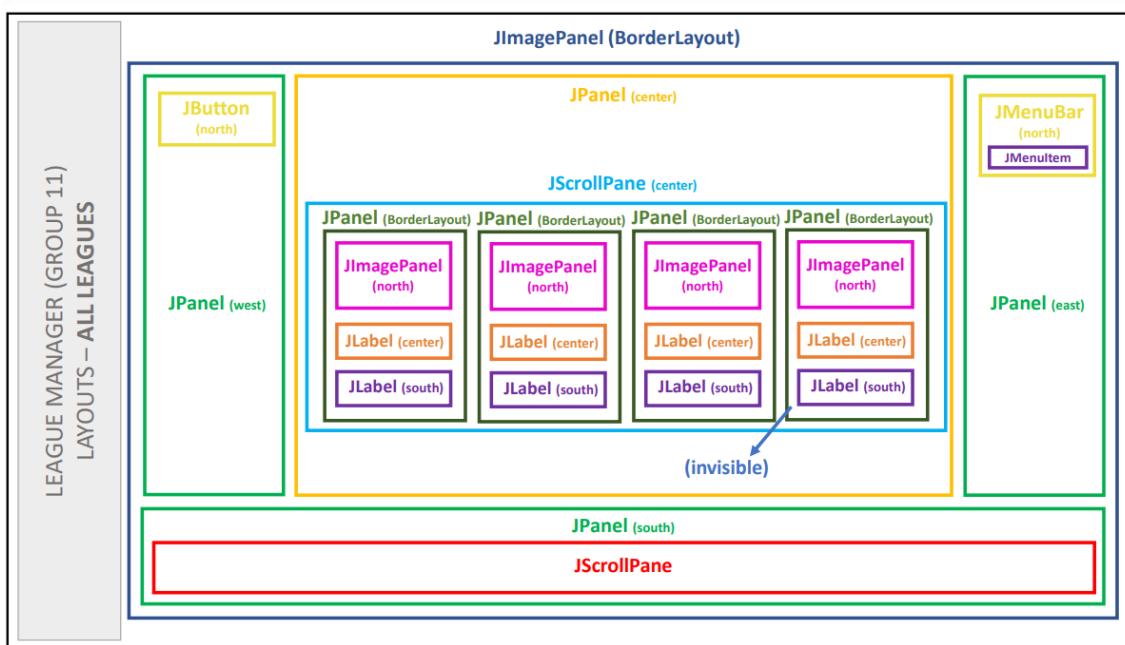


TABLE STATISTICS

LEAGUE MANAGER (GROUP 11)
LAYOUTS – TABLE STATISTICS

RANK	NAME	NUM. PLAYERS	P	W	D	L	PTS
1	F.C. BARCELONA	31	27	23	2	2	71
2	REAL MADRID	32	27	18	5	4	59
3	ATLÉTICO DE MADRID	28	27	16	6	5	54
4	REAL SOCIEDAD	29	27	14	6	7	48
5	BETIS	25	27	13	6	8	45
6	VILLAREAL	25	27	13	5	9	44
7	ATHLETIC DE BILBAO	27	27	10	7	10	37
8	RAYO VALLECANO	30	27	9	10	8	37

LIVE (⌚)

3 - 2 | 0 - 1

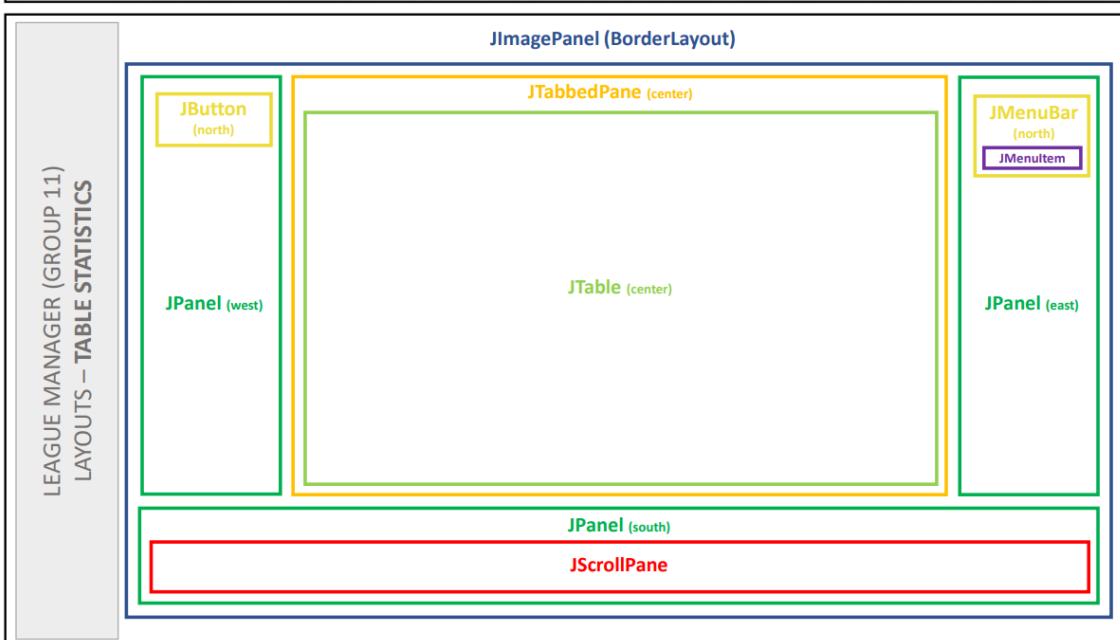
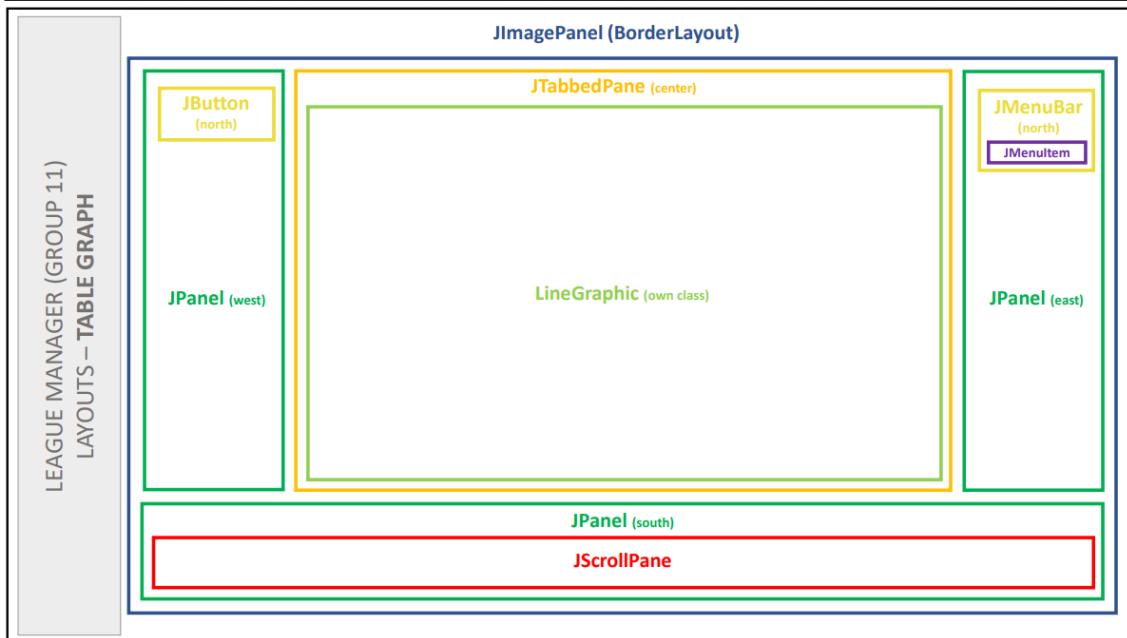
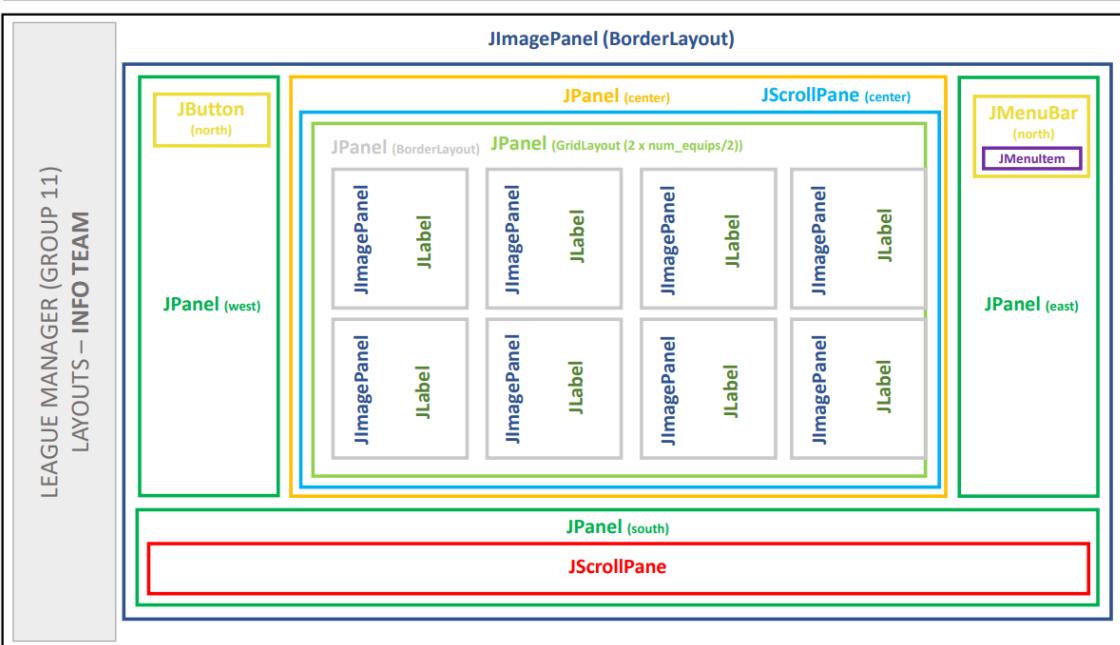


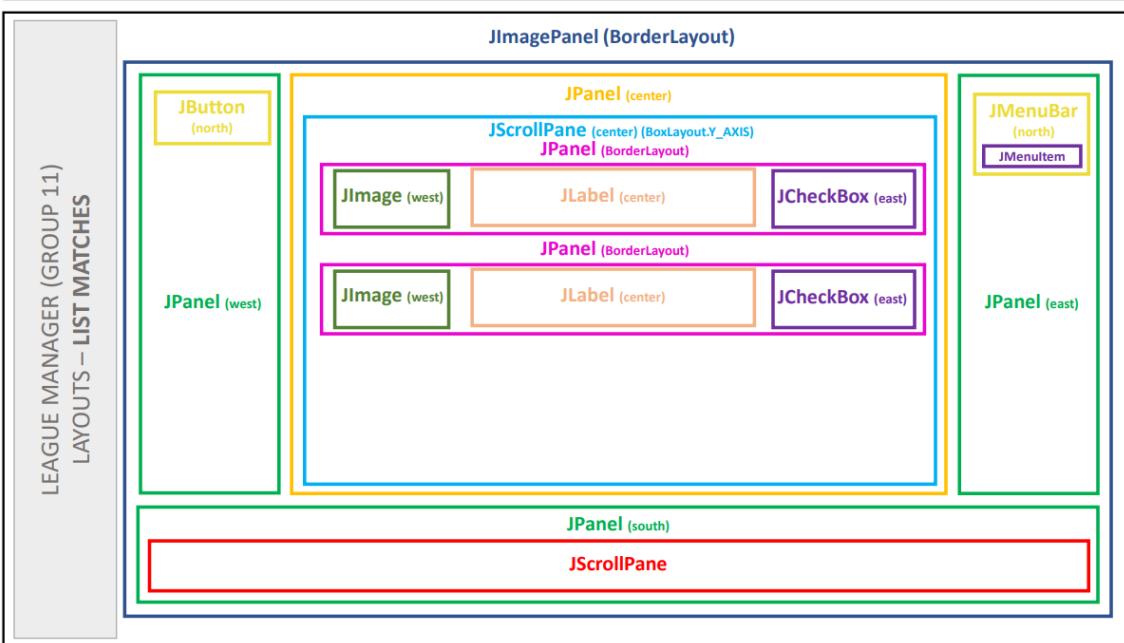
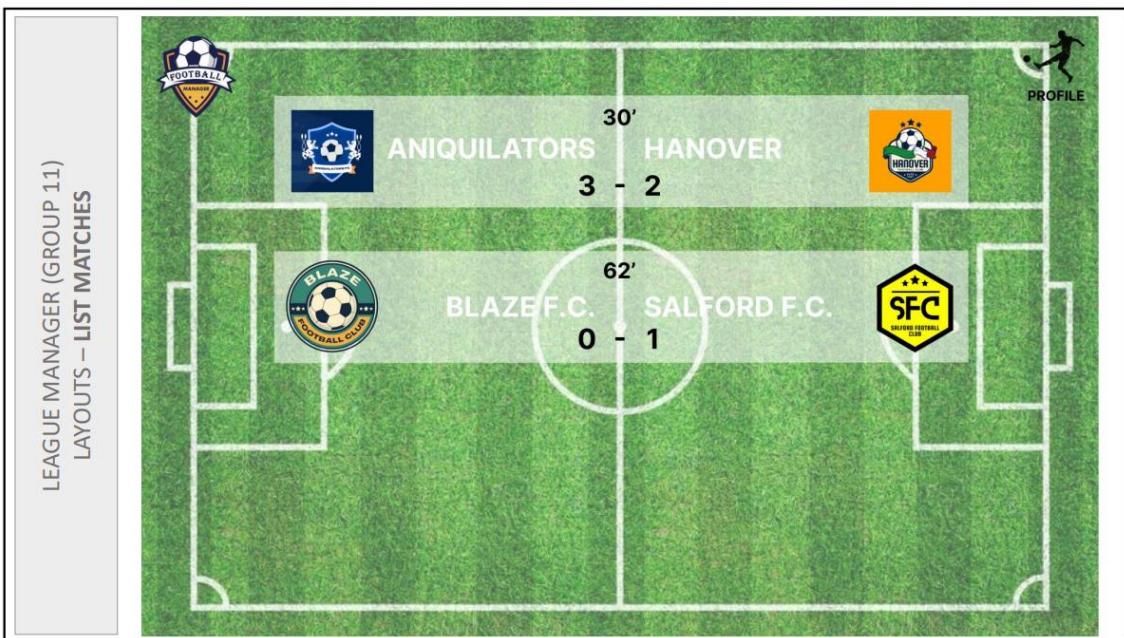
TABLE GRAPH



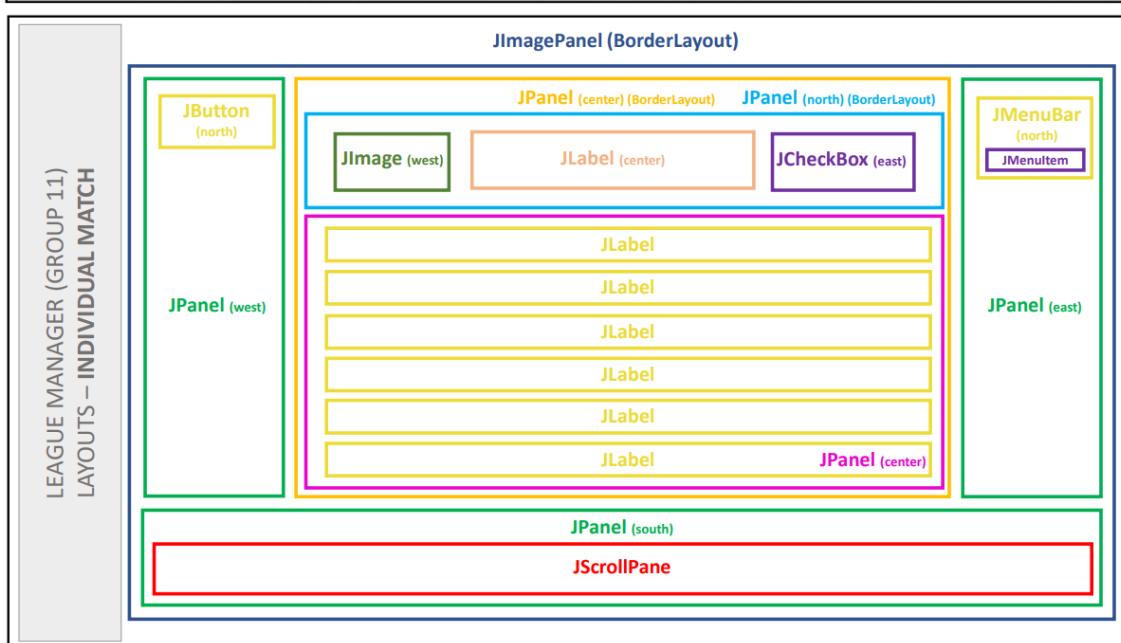
INFO TEAM



LIST MATCHES

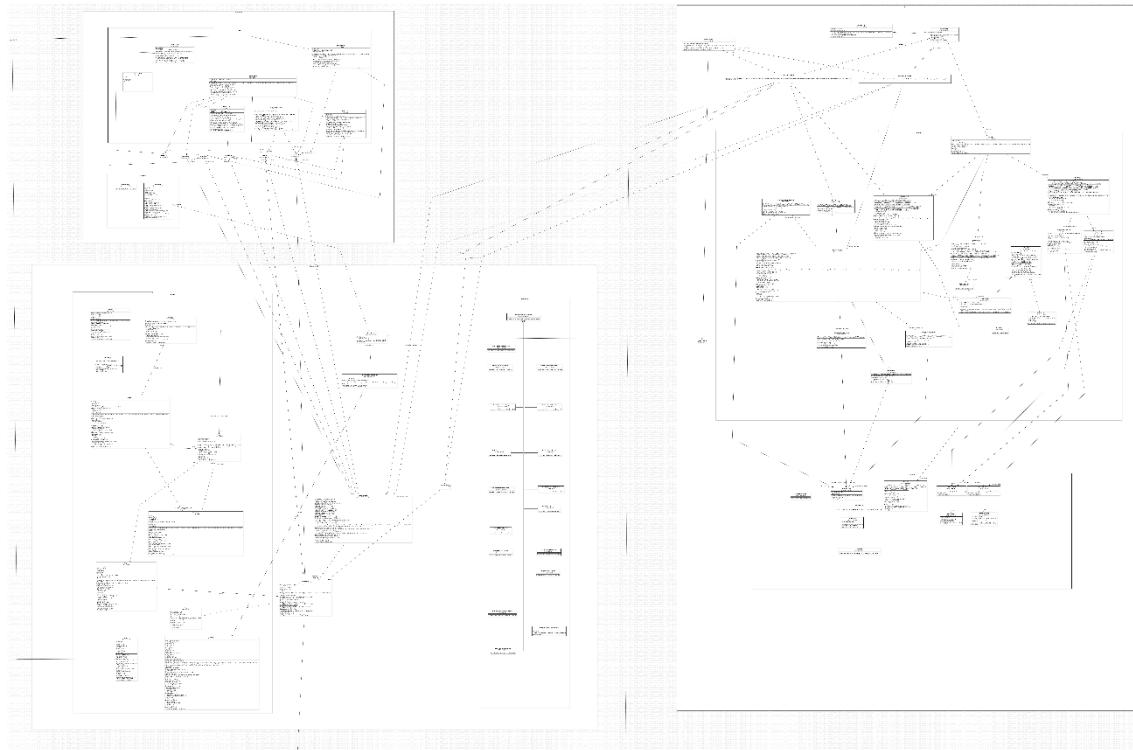


INDIVIDUAL MATCH



3. DIAGRAMA DE CLASSES

El nostre diagrama de classes és el que es mostra a continuació, sabem que es massa gran i que en la captura general no es podrà veure tot, però per la explicació la farem per carpetes amb imatges més enfocades al punt on volem explicar.



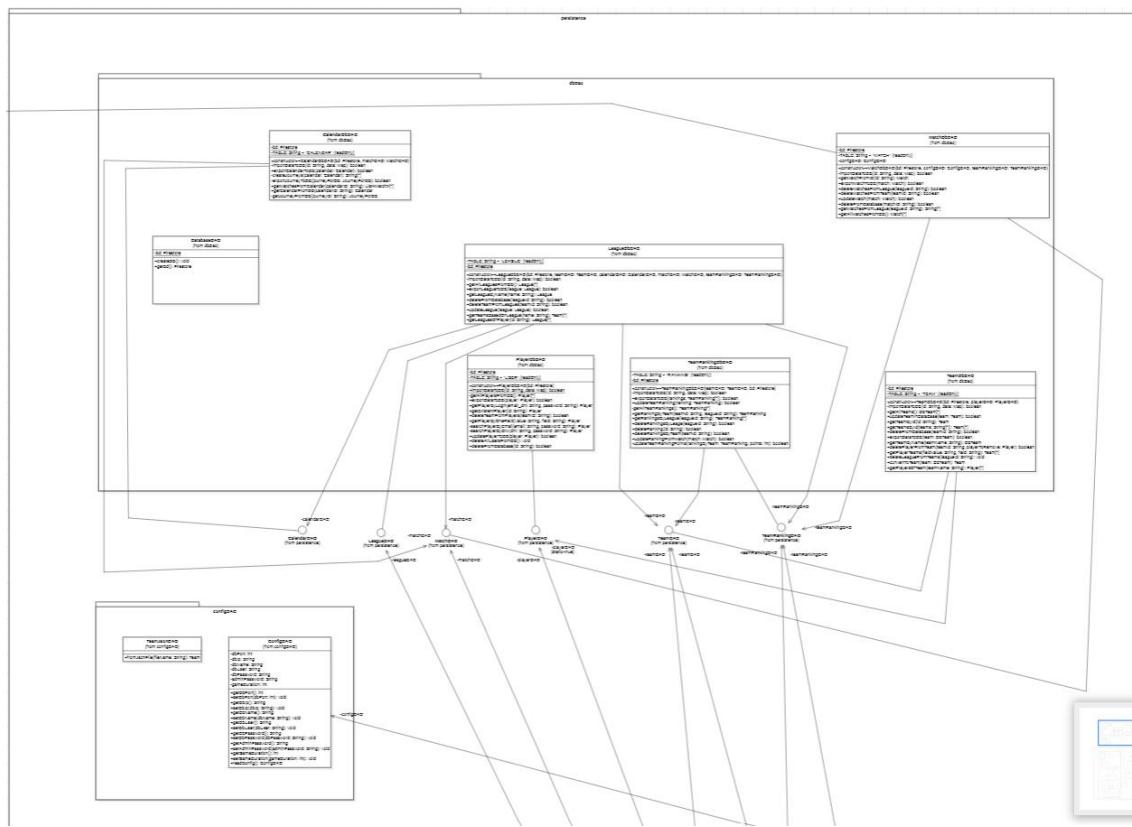
Per què hem necessitat un diagrama de classes tant gran? El programa que hem dissenyat és molt complert i per poder complir tots els requeriments del projecte necessitàvem poder contar amb classes dedicades a mostrar per pantalla, controlar l'execució, i guardar en persistència en el moment indicat per no col·lapsar la base de dades que hi ha darrera de tot el projecte.

Quin model d'estructura hem utilitzat? Per dur a terme el programa s'ha utilitzat una barreja entre el model: presentation -> business -> persistence, i el model: model -> controller -> view.

Quina lògica utilitzen les classes del nostre projecte? Utilitzen una lògica “tell don't ask” que permet a la classe encapsular totes les operacions que es requereixin sense que ho hagin de fer tot els “controllers”.

(Per veure el disseny UML amb més definició obrir el projecte “LeagueMDJ” o la imatge adjuntada en la carpeta de la memòria)

Part de Persistència:



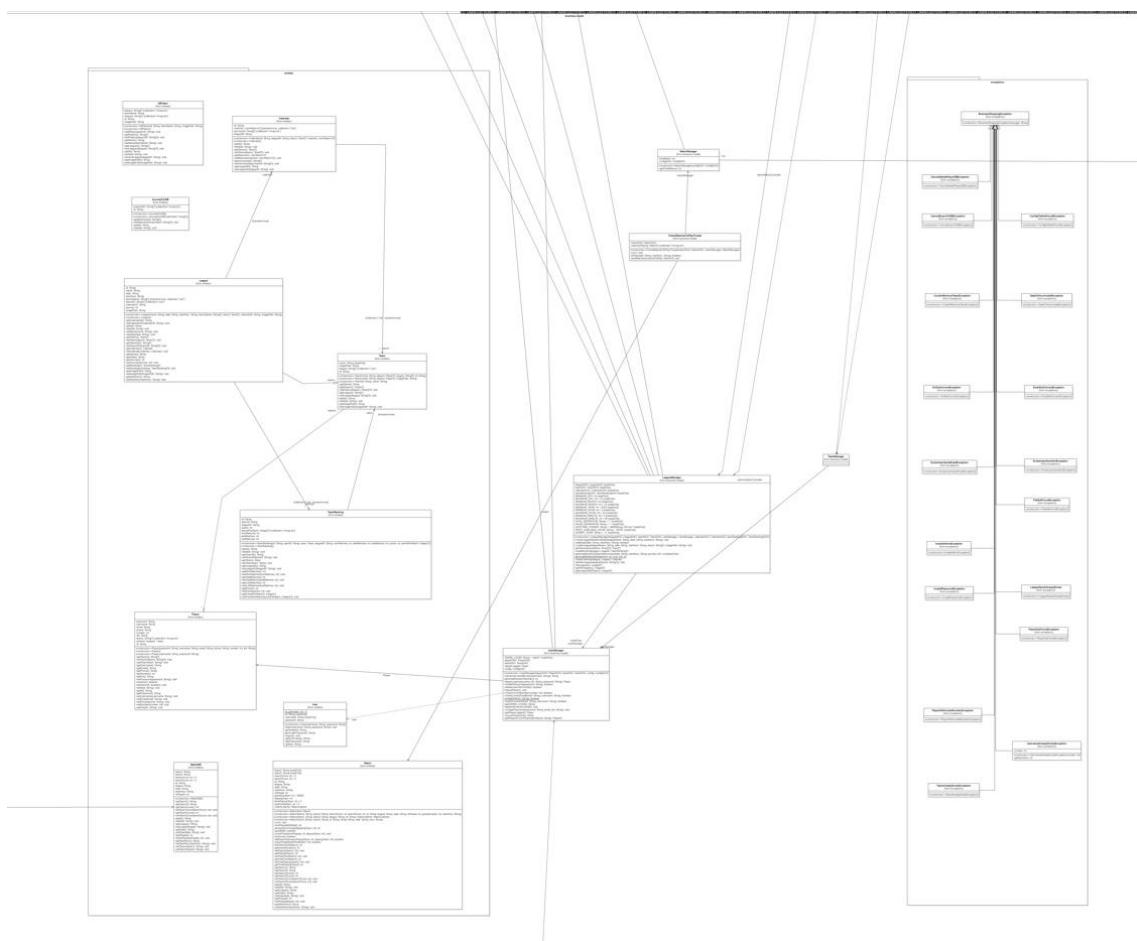
En aquesta carpeta de persistència el primer que ens salta a la vista és que hi ha dues carpetes internes, una que és “dbDAO” on guardem tots els mètodes per poder interactuar amb la base de dades online. Ens podem trobar amb una altre carpeta que és “config” que s’encarregarà d’obrir el programa amb una configuració inicial parametrizada cada cop que aquest sigui inicialitzat.

Dins de la carpeta “dbDAO” ens trobem classes com ara “MatchDbDAO”, “LeagueDbDao”... Que bàsicament representen cada taula guardada a la base de dades NoSQL firebase, de Google, que hem utilitzat pel projecte.

Dins de la carpeta “config” ens trobem la classe “ConfigDAO” i “TeamsJsonDAO” que ens permeten llegir i recaptar informació dels fitxers configuració.

A persistència, també ens trobem tots les interfícies que ens permeten comunicar les classes que guardaran a la base de dades amb els mètodes que han de tenir sí o sí perquè la lògica del programa funcioni correctament.

Part de Business o Model:



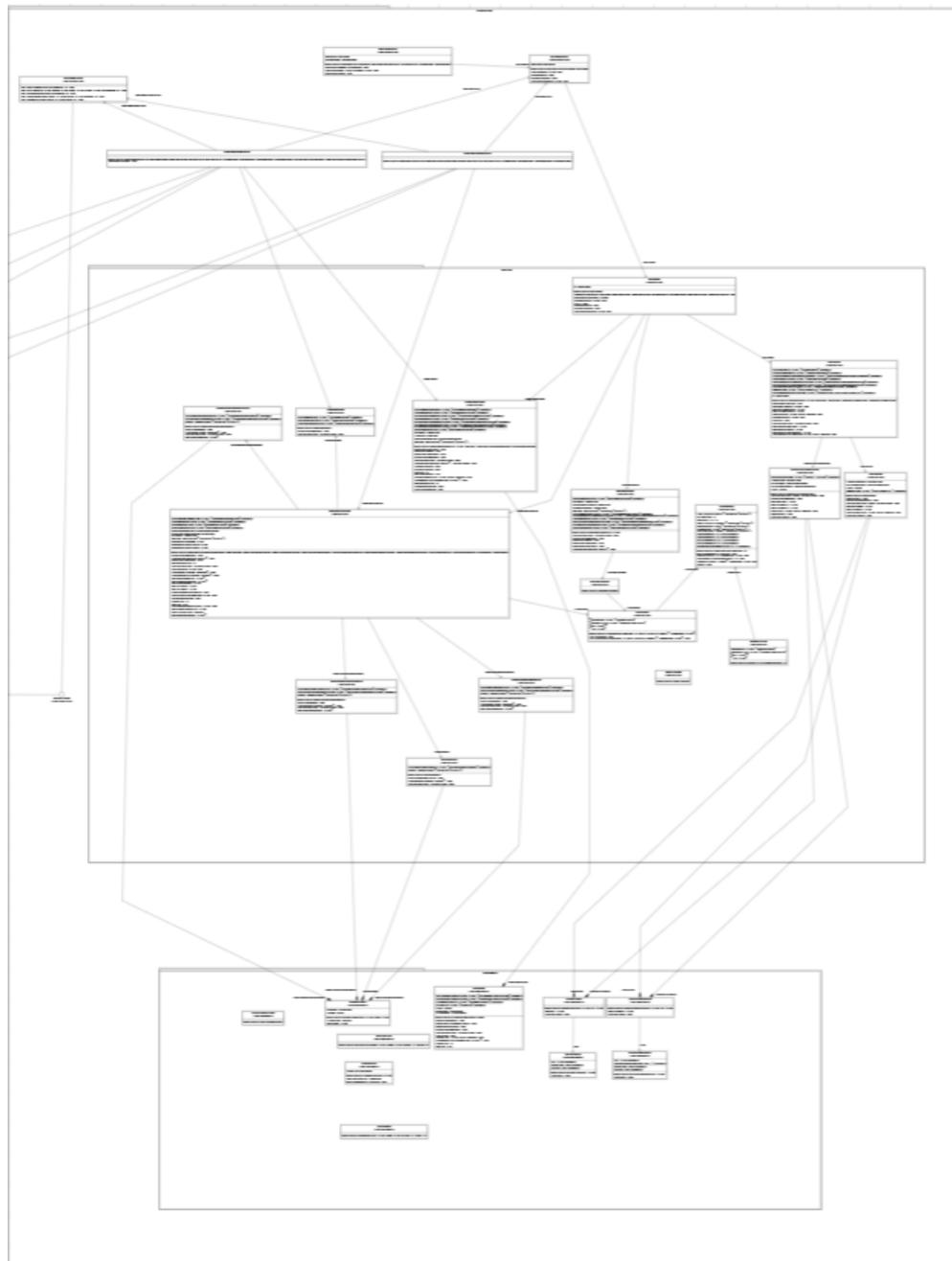
En aquesta part tenim totes les classes que ens proporcionen la lògica del programa. Totes les classes tenen connexions amb els manager que són les classes que contenen els mètodes per atorgar la informació en la manera pertinent a les classes controladores del flux del programa.

Les dues carpetes contingudes en business: “entities” i “exceptions” contenen classes que utilitza aquesta classe per transmetre la informació en manera d’Objectes cap a la carpeta de presentation.

Dins de la carpeta “entities” hi ha totes les classes objecte que ens guarden la informació de el que s’ha considerat que és un partit, jugador, temporada... També hi ha dues classes que no tenen relacions perquè són classes que només ens serveixen per transformar els partits a algun objecte que es pugui guardar a la base de dades.

Dins de la carpeta “exceptions” hi ha totes les excepcions que poden succeir a l’hora d’executar el nostre programa, si ens dona el cas d’alguna excepció l’usuari és avisat amb el missatge pertinent

Part de presentació:



En aquesta carpeta de presentació ens trobem tot el que l'usuari veurà per pantalla, és a dir, la UI. Aquesta és controlada pels diferents “controllers” que controlen el flux del programa segons les decisions que va prenent el usuari en la GUI dedicada d'aquest projecte.

La carpeta presentació està dividida en dues subcarpetes: “pantalles” i “ui_elements”. Aquestes carpetes representen la part visual de la UI, i contenen el format de les pantalles que s'ha dissenyat per a donar una bona experiència d'usuari.

Dins la carpeta “pantalles” ens trobem ja les pantalles complertes amb els seus pertinents “listeners” per poder relacionar pantalla amb el controlador d'aquesta i que aquest ens torni el resultat de la tria.

Dins de la carpeta “ui_elements” ens trobem amb “JPanels”, “JButtons” modificats per a que tinguin un comportament adaptat a les necessitats del projecte i es vegin de la millor manera possible.

Cal destacar també dins de la carpeta de presentació hi ha un “listener” que es dedica a rebre informació constant del manager “MatchManager” i transmetre-la de manera constant al “MatchController”, per actualitzar la vista a temps real.

4. METODOLOGIA DE DESENVOLUPAMENT

Per a desenvolupar un projecte de grans dimensions com ha sigut aquest, ens hem hagut d'organitzat molt bé i repartir-nos les tasques al llarg de la duració del projecte entre tots els membres de l'equip de treball.

El projecte ha estat dividit en 4 *sprints* d'una duració aproximada de 2-3 setmanes. Abans de començar-ne cap, repartíem les tasques que havíem decidit entre els membres del grup i seguidament ens posàvem a treballar. Tot i que cadascú tenia la seva feina individual, això no volia dir que cadascú es centrés únicament exclusivament en el que li tocava sinó que també era important anar veient com progressaven els altres.

Per a organitzar-nos els *sprints*, hem fet ús d'una eina que ens facilita la universitat que s'anomena *Jira* i que és un producte desenvolupat per Atlassian que permet la gestió àgil de projectes i el seguiment d'errors.

SPRINT 1:

- 1) Configuration file
- 2) Class diagram
- 3) UI mockups
- 4) Database design
- 5) Database setup

SPRINT 2:

- 6) UI mockups
- 7) Logout and account deletion GUI
- 8) Team/User creation GUI
- 9) Class diagram
- 10) Database design
- 11) Database setup
- 12) User login GUI
- 13) Admin login
- 14) Regular user login
- 15) Logout
- 16) Delete user account
- 17) Main Menu UI: Admin
- 18) Team/User creation

SPRINT 3:

- 1) UI mockups
- 2) Logout and account deletion GUI
- 3) Main Menu UI: User
- 4) Team/User creation GUI
- 5) Database information access
- 6) League creation UI
- 7) League creation
- 8) List / delete leagues
- 9) Show league details
- 10) Simulating Matches
- 11) Visualizing Matches

SPRINT 4:

- 1) Password management

- 2) League creation
- 3) List / delete leagues
- 4) Show league details
- 5) Statistics chart
- 6) Project report
- 7) Database information access
- 8) Removing teams
- 9) League creation UI
- 10) Simulating Matches
- 11) Visualizing Matches

Si mirem cadascuna de les tasques que s'han dut a terme en cada Sprint, veiem que hi ha algunes que es troben repetides en diferents Sprints, això vol dir que no es van tancar al final de l'Sprint. Aquest fet és degut a que ho bé no vam tenir temps de finalitzar la tasca o bé que la tasca la realitzava una persona sola i no era viable tancar-la en 2 setmanes d'Sprint.

Com en tot projecte, cal repartir els rols o les funcions entre les persones que conformen aquest, en el nostre cas, ens ho hem repartit de la següent manera:

Andrea: S'ha centrat principalment en crear la interfície de l'usuari. Primer va crear els layouts en una aplicació de disseny (*Figma*), i després va pensar en els layouts a utilitzar per a després programar-ho tot amb més facilitat. Quan això ja estava fet, juntament amb altres companys del grup, es va encarregar de comunicar les vistes entre elles. Gran part de la memòria, ha estat realitzada per ella.

Joan: Va formar part de l'equip que a l'inici del projecte treballava en el diagrama de classes. També es va encarregar de les funcions que s'encarreguen del login de l'usuari. El gràfic de les estadístiques el va dissenyar ell amb els càlculs pertanyents per a generar-lo a partir de les dades donades. Finalment, també es va dedicar a acabar el diagrama de classes abans de l'entrega.

Pol: La seva funció principal ha sigut la d'unir les vistes amb la lògica. Tot i que també s'ha encarregat d'anar ajudant allà on algú necessitava ajuda. Ha estat passant vistes del Figma a Swing juntament amb l'Andrea, alhora que connectant totes les pantalles, afegint els listeners on calien, demanant dades a la base de dades per poder mostrar-les en la vista en temps real. També comprovava que les dades que s'introduïrien mitjançant la vista arribessin on calia amb el format correcte, així com mostrant errors sempre que no anés com calia.

Oriol: Ha sigut el principal encarregat de la base de dades. L'ha dissenyada i ha creat tota la lògica d'inserció de dades i la eliminació d'aquestes també. Ha dut ha terme les següents tasques; serialització i deserialització d'objectes, mètodes cruds (create, read, update, delete) en els DAO, creació dels DAOs, la lògica de crear teams i totes les seves implicacions (comprovar si el team ja existeix, si l'usuari existeix, llegir el json i validar el json). S'ha encarregat també de la funcionalitat que té l'usuari per recuperar la contrasenya i també d'implementar els mètodes d'obtenció de dades en els managers.

Leo: S'ha encarregat de les següents tasques; login de l'administrador (diferenciar si està iniciant sessió un jugador o l'administrador), comprovar les dades al voler crear una lliga i si aquestes són correctes, crear una lliga nova. Relacionat també amb les lligues, ha creat la funció de llistar-les juntament amb tots els detalls necessaris i la funció d'eliminar-les qual l'administrador això ho desitja.

Jan: La seva feina ha estat dur a terme la simulació del partit i implementar-la en un Thread. Calia fer un partit amb una heurística realista i que tingui sentit, per tant s'ha encarregat que tingui un mínim de sentit en la implementació. També ha fet la connexió entre els managers i controllers amb la simulació del partit per tal de facilitar-ho juntament amb la vista.

Per a desenvolupar el projecte, a part de l'ús del Jira per a la gestió d'aquest, hem utilitzat també el Bitbucket, un servei de repositoris de Git que permet a cada membre del grup poder anar penjant la feina que realitza així tothom pot tenir accés a ella. D'aquesta manera en el repositori guardavem el codi i tothom hi podia accedir quan volgués.

5. DEDICACIÓ

Per a dur a terme el projecte, hem hagut de dedicar-li moltes hores. A continuació desglossarem les hores dedicades en 4 grups diferents i estimarem el temps dedicat a cadascun.

(1) **comprendió de les especificacions:** com el projecte era de grans dimensions i per tant tenia un elevat nombre d'especificacions, al principi de tot vam haver de llegir l'enunciat diverses vegades i anotar aquelles que consideràvem importants. A més, durant el desenvolupament del treball, ha calgut també anar-les consultant novament per a complir amb totes les restriccions i requeriments que se'ns havien donat. En total, considerem que per comprendre i tenir clares totes les especificacions, vam haver de dedicar unes 4 hores en total com a grup.

(2) **anàlisi i disseny:** dissenyar i estructurar el què volem fer és una de les part més importants del desenvolupament de qualsevol projecte. En el cas d'aquest en concret, abans de començar a codificar res, calia dissenyar la interfície amb l'usuari, la base de dades on s'emmagatzema tota la informació relacionada amb els usuaris i els partits i el diagrama de classes que ens permet veure d'una manera molt visual, com organitzaríem tot el codi. Fer aquesta feina ben feta, facilita molt després la feina de codificació es per això, que vam dedicar-li molt de temps per a que després treballéssim amb facilitat i sense problemes. En total, estimem que hem dedicat unes 7 hores a crear el diagrama de classes, unes 10 hores més a crear i dissenyar la bbdd i unes 12 hores en dissenyar la UI. Dedicant en total 29 hores a l'apartat d'anàlisi i disseny.

(3) **codificació:** aquesta és obviament la part central del projecte i per tant, a la que li hem dedicat moltes hores. Codificar no només tracta de picar codi sinó també d'anar verificant que tot el que s'està fent, està funcionant correctament. Per a calcular les hores dedicades a la codificació, hem decidit desglossar-ho en els següents punts:

Implementació bbdd: 60 hores

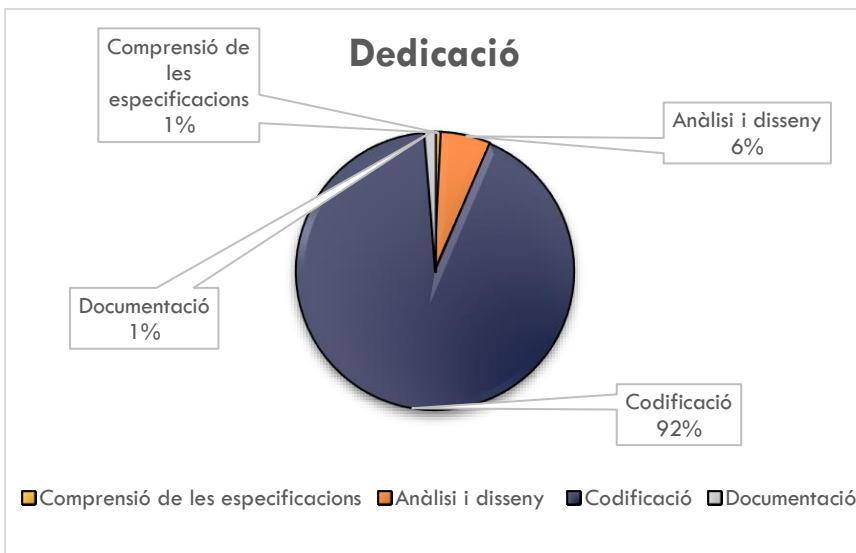
Reunions: 4 hores aproximadament

Feina a classe: 130 hores aproximadament.

Feina a casa: 280 hores aproximadament.

Així doncs estimem que a aquest apartat li hem dedicat aproximadament 474 hores.

(4) **documentació:** finalment, com en tot treball, cal realitzar un informe escrit en el que es detallin punts com; què s'ha fet, com s'ha fet, com s'ha treballat i què s'ha après. Moltes vegades aquest escrit es fa els últims dies previs a l'entrega ja que és quan s'ha acabat o s'està a punt d'acabar la feina i per tant es pot explicar a la perfecció cadascun dels punts mencionats anteriorment. La memòria d'aquest projecte no era molt extensa ja que els apartats de *Disseny de l'interfície gràfica i Diagrama de classes*, havíem començat a treballar en ells des de l'inici del projecte. Estimem doncs que a l'apartat de documentació hi hem invertit unes 6 hores.



6. CONCLUSIONS

En aquest projecte, hem desenvolupat una aplicació en Java que permet als usuaris consultar tots els partits de les diferents lligues que hem creat. A través d'aquest projecte, hem tingut l'oportunitat d'aplicar les classes i els conceptes de programació orientada a objectes que hem après durant el curs de Disseny i Programació Orientada a Objectes. A més, hem tingut l'oportunitat d'explorar l'ús de bases de dades en línia, les quals no havíem treballat anteriorment.

Una de les principals conclusions que hem tret és la importància de treballar en grup en el desenvolupament de projectes d'aquesta magnitud. L'experiència de col·laborar en un equip és molt valiosa, ja que moltes empreses d'avanguarda actuals requereixen una gran cooperació entre múltiples persones per dur a terme projectes de gran envergadura. Això ens ha proporcionat una oportunitat de aprendre i aplicar les habilitats de treball en equip en el context de la nostra formació en enginyeria informàtica.

Una bona comunicació entre tots els membres del grup ha estat essencial per al bon desenvolupament del projecte. Hem utilitzat diverses eines en línia, com ara repositoris i taules de planificació com "BitBucket" i "Jira" d'Atlassian, per organitzar i repartir les tasques de manera eficient. Aquestes eines ens han permès treballar de manera cooperativa en la implementació del codi, ja que ja les havíem utilitzat en assignatures anteriors i ens vam adaptar-hi ràpidament.

Per a una comunicació efectiva i ràpida, també vam utilitzar grups de xat i trucades de vídeo en línia. Això ens va permetre comunicar-nos entre tots els membres del grup de manera eficient, tot i que no estiguéssim en el mateix lloc físic. Aquests recursos tecnològics ens han ajudat a mantenir-nos connectats i a treballar col·laborativament durant tot el projecte.

L'entorn de programació IntelliJ ens ha proporcionat eines molt útils per al desenvolupament del projecte. Hem pogut connectar el nostre directori local amb el repositori en línia que utilitzàvem, facilitant la gestió de versions i el control de codi font. Això ens ha ajudat a coordinar i integrar els canvis realitzats per diferents membres del grup.

Com s'ha pogut veure a l'apartat del disseny de la interfície gràfica, hem volgut experimentar molt amb swing atrevint-nos a fer coses diferents i per demostrar que swing ens ofereix més opcions de les que ens pensem. Tot i que ha sigut una feina complicada i a vegades pesada, estem molt contents de la interfície gràfica que hem aconseguit fer.

Per a la base de dades, utilitzar una en línia i NoSQL com és la de Firebase de Google, ha tingut diferents avantatges i inconvenients. Com a avantatges destacar que guardar coses en aquella base de dades és molt més senzill que en una base de dades SQL que s'ha de distribuir molt bé tota la informació i gestionar les consultes que se li han de fer per obtenir-la. També que concretament en la base de dades Firebase et permetia gestionar totes les dades que teníem guardades mitjançant una interfície gràfica molt intuitiva i fàcil d'utilitzar. Com a desavantatges és que al ser en línia es necessita una bona connexió a internet amb molt ampli de banda, si no el projecte funciona de manera molt lenta fins a punts que pot requerir massa temps entre pantalla i pantalla (més del que ens hagués agradat).

Però, finalment, amb paciència i dedicació, cada membre del grup s'ha especialitzat en les diferents tasques assignades, com ara, gestions de base de dades, vistes, lògica del sistema i comunicació entre les diferents classes. Això ha permès que el projecte es desenvolupi de manera més ràpida i amb menys errors. La divisió estructurada de tasques i l'especialització ens han permès aconseguir una major eficiència en el desenvolupament.

En resum, tots els membres del grup estem molt satisfets amb els resultats del projecte. Hem aconseguit complir tots els requeriments establerts en el període de temps que se'n ha demandat.

7. BIBLIOGRAFIA CONSULTADA

Borra datos de Cloud Firestore | Firebase. (s.f.). Firebase.

<https://firebase.google.com/docs/firestore/manage-data/delete-data?hl=es-419>

Date and Time Classes (The Java™ Tutorials > Date Time > Standard Calendar). (s.f.). Moved.

<https://docs.oracle.com/javase/tutorial/datetime/iso/datetime.html>

Decorating a JTextField with an image and hint. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/6089410/decorating-a-jtextfield-with-an-image-and-hint>

Firebase - Queries. (s.f.). Online Courses and eBooks Library.

https://www.tutorialspoint.com/firebase/firebase_queries.htm

Fit size of an ImageIcon to a JButton. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/36957450/fit-size-of-an-imageicon-to-a-jbutton>

How do I center a JTextField. (s.f.). Stack Overflow. <https://stackoverflow.com/questions/15507639/how-do-i-center-a-jtextfield>

How to change the size of the font of a JLabel to take the maximum size. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/2715118/how-to-change-the-size-of-the-font-of-a-jlabel-to-take-the-maximum-size>

How to make drawn images transparent in Java. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/14097386/how-to-make-drawn-images-transparent-in-java>

How to make Scrollbar of scroll pane make movable automatically with the cursor as we press TAB Button on keyboard? (s.f.). Stack Overflow. <https://stackoverflow.com/questions/7227441/how-to-make-scrollbar-of-scroll-pane-make-movable-automatically-with-the-cursor>

How to resize JLabel ImageIcon? (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/6714045/how-to-resize-jlabel-imageicon>

JTable, disable user column dragging. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/17641123/jtable-disable-user-column-dragging>

JTable with horizontal scrollbar. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/2452694/jtable-with-horizontal-scrollbar>

Painting a custom JScrollBar. (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/1290379/painting-a-custom-jscrollbar>

Realiza consultas simples y compuestas en Cloud Firestore | Firebase. (s.f.). Firebase.

<https://firebase.google.com/docs/firestore/query-data/queries?hl=es-419>

Scheduling algorithm for a round-robin tournament? (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/6648512/scheduling-algorithm-for-a-round-robin-tournament>

Table Selection Events and Listeners : JTable « Swing « Java Tutorial. (s.f.). Programming Tutorials and Source Code Examples.

http://www.java2s.com/Tutorial/Java/0240_Swing/TableSelectionEventsandListeners.htm

Threads in Java. (s.f.). Stack Overflow. <https://stackoverflow.com/questions/2865315/threads-in-java>

The transient Keyword in Java | Baeldung. (s.f.). Baeldung. <https://www.baeldung.com/java-transient-keyword>

What does 'synchronized' mean? (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/1085709/what-does-synchronized-mean#:~:text=The%20synchronized%20keyword%20prevents%20concurrent,of%20them%20at%20a%20time>

What does "synchronized" mean in Java? (s.f.). Stack Overflow.

<https://stackoverflow.com/questions/7848471/what-does-synchronized-mean-in-java?noredirect=1&lq=1>

Grafiati: Generador automático de citas online. (s.f.). Grafiati: Оформити списки використаних джерел онлайн. <https://www.grafiti.com/es/>

A part del que hem buscat a internet, també hem consultat els apunts de classe i si teníem algun dubte, també preguntàvem al professorat i als becaris.