S2-Project_B4

Alumnes:

- Albert Tomàs
- Anna Ferrer
- Guillem Roselló
- Marta Vilaseca
- Laura Pascual

Tutor Projecte :

- Pol Muñoz

Projecte escollit:

BETWEEN US

Índex

Resum del projecte 3

Disseny de la interfície gràfica 4

Pantalla inicial 4
Login i registre 5
Pestanya Warning 6
Conta 7
Configuració i eliminació de conta 8
Pestanya d'estadístiques 9
Gestió de partides (+ esborrat de partides) 11
Creació i duplicació de partides 14
Partida 18
Taula Logs 21
Pestanyes finals (Victoria / Derrota) 22

Diagrama de classes 23

UML inicial 23
UML final: 23

Metodologia de desenvolupament 25

SPRINT 1 27
SPRINT 2 28
SPRINT 3 29
SPRINT 4 30

Dedicació 31

Taula (Temps aproximat) 31

Gràfic 31

Argumentació 32

Comprensió de les especificacions 32
Anàlisi i disseny 32
Codificació 32
Documentació. 33

Conclusions 35

Bibliografia consultada 37

1. Resum del projecte

Per aquest projecte, havíem de crear una "aplicació" que simulés un dels jocs que ja existeix, " *l' Among US* "..

En la nostra versió, els diferents usuaris poden entrar a la seva conta i poden gestionar les seves diferents partides. Podent-ne crear de noves, ja sigui amb la mateixa configuració que una altre, com completament noves, podent jugar-les i esborrar-les.

A més a més, cada jugador pot accedir a les seves estadístiques segons la quantitat de partides guanyades, i pot eliminar la seva conta.

Tenint en compte el que he mencionat, cap dels diferents jugadors pot tenir el mateix nom d'usuari, ni tampoc, el mateix correu. El mateix passa amb les partides, cap jugador pot tenir més d'una partida amb el mateix nom, però, diferents jugadors sí que poden tenir partides amb noms iguals.

A cada partida, el jugador pot ser d'un color diferent, el qual l'haurà escollit durant la creació de la partida. Un cop a dins la partida, el jugador es pot anar desplaçant per el mapa (però sempre jugarà com a tripulant). En aquesta versió, el jugador jugarà amb la pròpia aplicació, és a dir, no es connectarà online per jugar amb altra gent, a diferència del "*Among US*" original.

Durant la partida, el jugador podrà decidir quins amongUS són tripulants i quins són impostors. Un cop hagi decidit, en una sala en concret del mapa (diferent a cada un d'ells), podrà comprovar si les seves sospites eren correctes. Si ho fossin, acabaria la partida i guanyaria. En cas contrari, es motraria un missatge i continuaria jugant. A més a més, el jugador també pot accedir a la GameLogs durant la partida (també en una sala en concret diferent localitzada a cada mapa), aquesta li aportarà la informació de cada NPC, quan ha entrat a cada sala i en quin instant.

Durant la partida, els NPC s'hauran d'anar movent i els impostors, a més a més, hauran de poder saltar d'habitació per el conducte de ventilació (quan estigui sol), i podrà eliminar altres NPC (1 sol tripulant a la mateixa habitació.

2. Disseny de la interfície gràfica

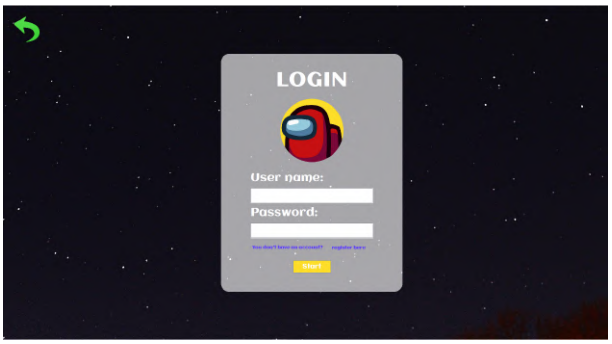
- Pantalla inicial

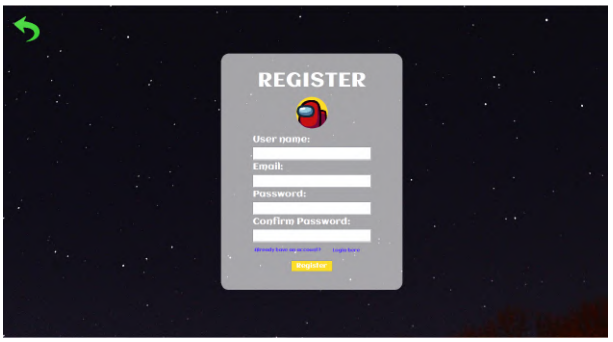
Per aquesta vista, s'ha fet ús d'una JImagePanel per fer el fons amb un BorderLayout.

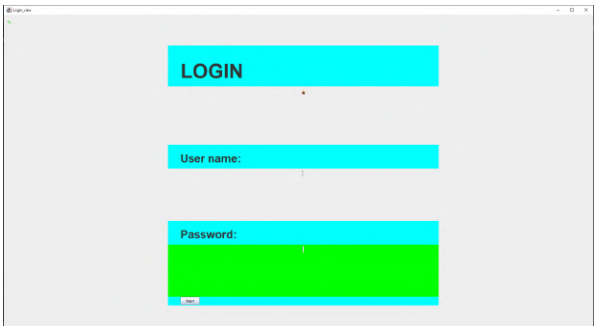
Al west, s'ha col·locat el botó de sortida dibuixant un ImageIcon (com en tots els botons que són dibuixos)

Al center s'ha fet el mateix, s'ha col·locat un botó, però en el JPanel que s'ha ficat en el Center (també un BorderLayout) se li ha creat un Empty border per col·locar el botó exactament a on es desitjava.

- ## Login i registre







La primera pestanya que es va fer amb el Java swing va ser la del Login. Al ser la primera, al principi anava quedant bastant deformada (igual que el que va passar durant les AC de vista).

El problema principal va ser que es va intentar fer amb un box layout i per tant, els tamanys que indicaves no s'acabaven de respectar.

Finalment, es va aconseguir aquest resultat fent:

Imatge del fons com a border layout

West → la fletxa de back

Center → Imatge rectangle amb un GridBagLayout

Jlavels, JImagePanel, JTextfields, flowLayout (per lo de register here) i JButons.

Tot ficat amb el gridBagLayout de forma vertical

A partir d'aquesta pantalla, ja vaig veure les coses més clares i vaig entendre millor el swing. (Ja que vaig haver de fer moltes proves abans d'arribar a aquí.) → (Laura).

Per la del registre, es va agafar la vista ja creada del login i es va adaptar al que necessitava. Els JPanels s'han mantingut.

- ## Pestanya Warning

Pel que fa a la creació de la pestanya Warning, aquesta està feta a base de un JPanel que afegim en un JDialog.
A més a més, dins d'aquesta afegim un rectangle on s'introduirà la informació del warning.
Dins del rectangle s'estructura com:

El text "Error", el qual utilitzem un JLabel. A més a més també afegim un nou JLabel per mostrar el missatge d'error per pantalla en funció del error que realitza l'usuari. Aquest s'actualitza a través de la funció updateWarning(). L'objectiu d'aquesta vista es que sigues adaptable per a qualsevol error que pogués sorgir durant l'execució. Per tant el text s'ha de adaptar al tamany de la vista i no sortir dels límits. Per fer això hem trobat una manera eficient de fer-ho: utilitzant html. Al ficar dins del JLabel el missatge d'error en el següent format: <html> + error + </html>. D'aquesta manera en cas de que el error sigui molts llarg, el text es posarà en la següent línia automàticament

A més a més també, tenim el botó d'acceptar, introduït amb un JButton, aquest farà que la pantalla es tanqui quan es doni a l'accept.

- Conta

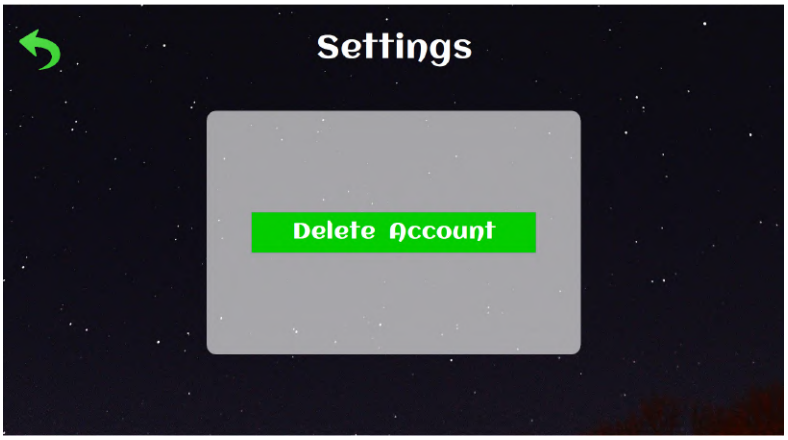
En aquest cas, aquesta pestanya és molt semblant a la inicial. S'ha afegit 2 JButton amb un ImageIcon cada un. El de tornar enrere es troba en el West del BorderLayout de la imatge del fons (JImagePanel), i el de configuració en el East.

Finalment, pels JButton que es troben al mig, s'ha fet un JPanel de tipus BorderLayout que conté un JPanel de tipus GridbagLayout (Ja et centra les coses directament). El JPanel del Border Layout ens va ajudar a limitar el tamanys del Gridbag per així, poder abaixar els dos botons. (Sense el Empty border del center, quedaven massa amunt).

Per acabar de concretar, la distància entre els botons Games i Stats es va fer afegint un JPanel buit entre aquests dos, ja que, el weighty del GridBagLayout els separaba massa tot i afegint el mínim de separació.

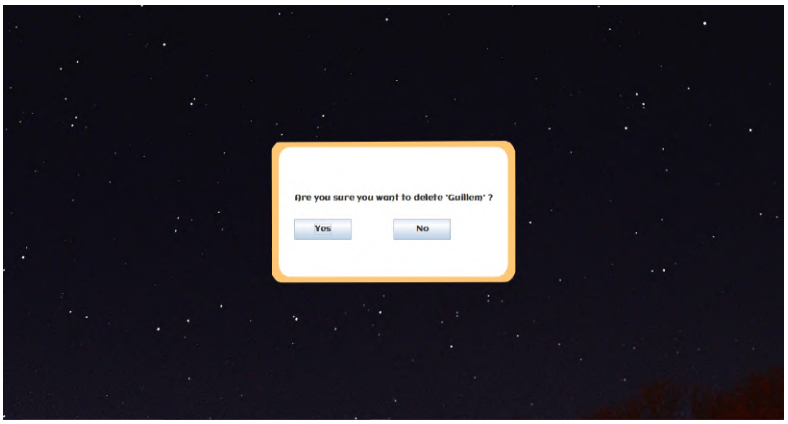
- ## Configuració i eliminació de conta

Un cop s'accedeix al botó delete account el joc et mostra la vista de confirmació de delete account, on es podrà escollir entre "yes" o "no", en ambos casos al apretar el botó el joc et redirigirà de pantalla, únicament canviarà a nivell de lògica eliminar la conta de l'usuari.



La vista en qüestió és la següent:

La vista està formada per una imatge de fons (JImagePanel), és a dir, la imatge amb les estrelles, a més a més disposa d'una altre imatge superposada, el rectangle central (JImagePanel). Per tal d'insertar la imatge de rectangle s'ha utilitzat un GridLayout, aquest es troba centrat. S'ha utilitzat un border per aconseguir aquest efecte, amb 20 al top, bottom, left i right.



Per tal d'organitzar el text del rectangle i els butons s'ha utilitzat novament un nou GridLayout. En el cas del text "*Are you sure you want to delete?* ", s'ha utilizat un JLabel el qual s'ha modificat la lletra i el tamany. Aquest ocupa 1 cel·la d'altura i 5 cel·les de llargada. També s'ha utilitzat la propietat de insets per tal de deixar un marge de 30 amb la part superior de la vista.

Pel que fa als botons s'ha utilitzat JButton, els dos ocupen 1x1.

- Pestanya d'estadístiques

Pel que fa a la vista d'estadístiques s'ha dividit en dues classes diferents, per un costat trobem PlayerEvolutionView i per l'altre costat trobem GraphDrawer.
En primer lloc trobem l'estructura general per tal de crear la gràfica, és a dir GraphDrawer. Aquesta s'ha realitzat:

En primer lloc, trobem l'ús de la classe Graphics, aquesta ens ha permès dibuixar les línies i mostrar la part de text dels eixos.

Per tal de definir les separacions entre dades, s'ha restat el final i el principi tant de l'eix x com de l'eix y. En el cas de l'eix Y s'ha dividit entre 10, donat que és un valor fixe, sempre anirà del 0% al 100%.

En el cas de l'eix X, s'ha divit entre el número total de partides jugades pel jugador, per tant dependrà en cada cas.
A continuació, utilitzem un bucle per tal de dibuixar les línies tant de l'eix X, com l'eix Y. Utilitzant drawLine.

A continuació, per tal de generar el text dels eixos s'utilitza una String, la qual s'introdueix a la gràfica utilitzant FontMetrics, metrics.stringWidth.

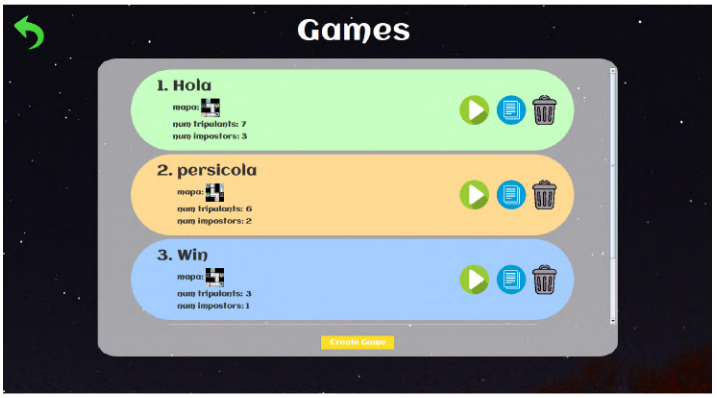
Finalment, per tal de dibuixar els diferents punts de la gràfica, utilitzem drawLine, on fem que x1 i y1 son sempre les coordenades de x2 i y2 anteriors.

En segon lloc, trobem l'estructura general de la vista, és a dir el PlayerEvolutionView. Aquest es divideix en:

1. La part del nord, dins d'aquesta trobem el botó per tal d'anar endarrera. Hem utilitzat un JButton, aquest conté la imatge de la fletxa endarrere. També trobem el títol de "Player Evolution", on utilitzem un JLabel.
2. La part central, trobem la gràfica, la qual s'ha explicat anteriorment.

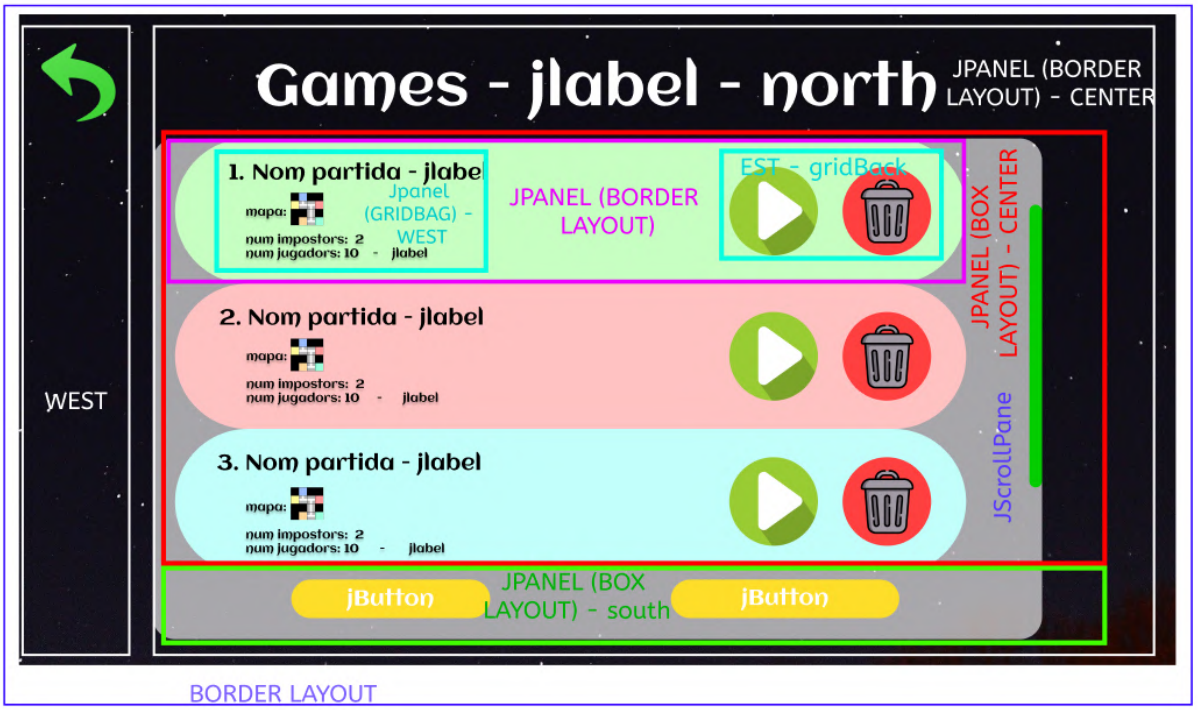
Totes aquestes parts s'introdueixen al JImagePanel, que conté el fons de les estrelles.

- ## Gestió de partides (+ esborrat de partides)



Per aquesta vista, s'ha mantingut el border layout principal, però, el botó de tornar enrere s'ha afegit al nord juntament amb el Games. A dins el nord s'ha fet un nou JPanel en BorderLayout per ficar el botó del back a la zona West del Nord. En el centre es troba el JLabel amb un Empty border al left per acabar de centrar.

Per fer la part del centre, es va fer un JImagePanel amb el rectangle i amb un BorderLayout. En el centre, es va afegir un JScrollPane amb un JPanel que de tipus boxLayout, per així poder anar ficant dinàmicament les partides.
En el sud, es va ficar un GridBagLayout i a dins d'ell un botó. D'aquesta manera, gràcies al GridBag, vam evitar que el botó de "Create Game" no s'extengués per tot el sud.

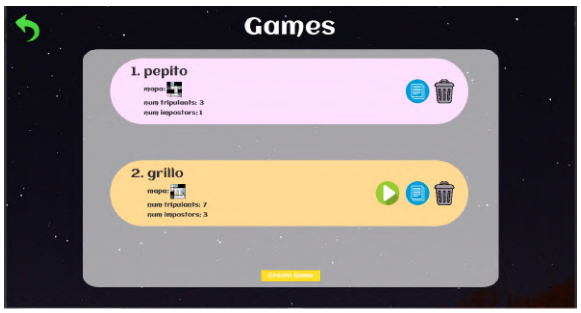
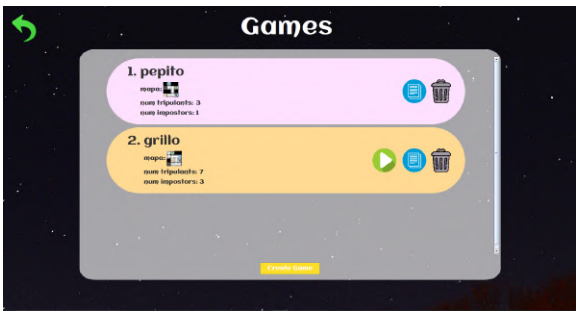


Cada partida, és un JPanel de tipus FlowLayout. A dins d'ell hi ha el JImage panel amb una imatge diferent segons el color del propi jugador en aquella partida en concret. Aquest JImagePanel té un BorderLayout.

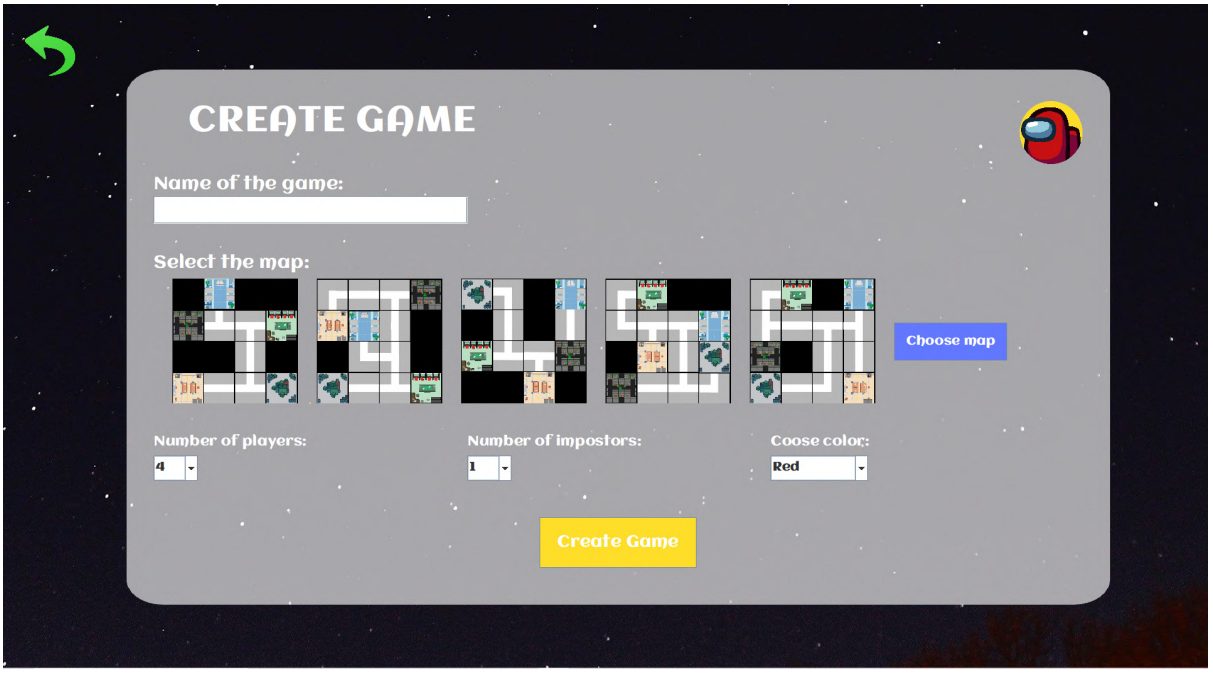
En el west es troba un nou JPanel de tipus GridBagLayout per ficar la informació referent a la partida.

En el East, també es troba un Jpanel amb un GridBagLayout on es troben els diferents botons. Aquelles partides que ja han estat finalitzades, no tenen el botó de play.

Finalment, comentar que hi ha un JPanel buit amb un empty border en alguns casos.
Quan el nombre de partides era 2, es mostraven amb un espai el mig, per evitar-ho, en el cas de que només siguin dos partides, es fica en el BoxLayout el JPanel buit, i quan no són dos, en el cas d'haver aquest JPanel, s'elimina.

BEFORE	AFTER
	

- ## Creació i duplicació de partides



Per aquesta vista, s'ha mantingut també un border layout, en el que posem un JImagePanel del background (fotografia de les estrelles que es veu de fons).

Després s'ha fet un mètode que retorna un JPanel anomenat "drawWest" per posar la imatge de la fletxa "back" que és un JButton, el qual té una "ActionCommand" quan es pressiona per tornar a la pantalla de llista de partides.

Després tenim el mètode "drawCenter", que retorna un JPanel on dibuixarem tot el que es mostra al centre de la vista.
Primer, creem un altre Border Layout el qual serà el rectangle gris que veiem, una JImagePanel.

A continuació, creem un GridBag Layout, que serà la base de tota la nostra vista, de manera que la quadrícula que ens crea el GridBagLayout, ens permetrà anar col·locant cada peça d'aquesta vista en una part de la cuadrícula, ocupant una sèrie de quadrats determinats.

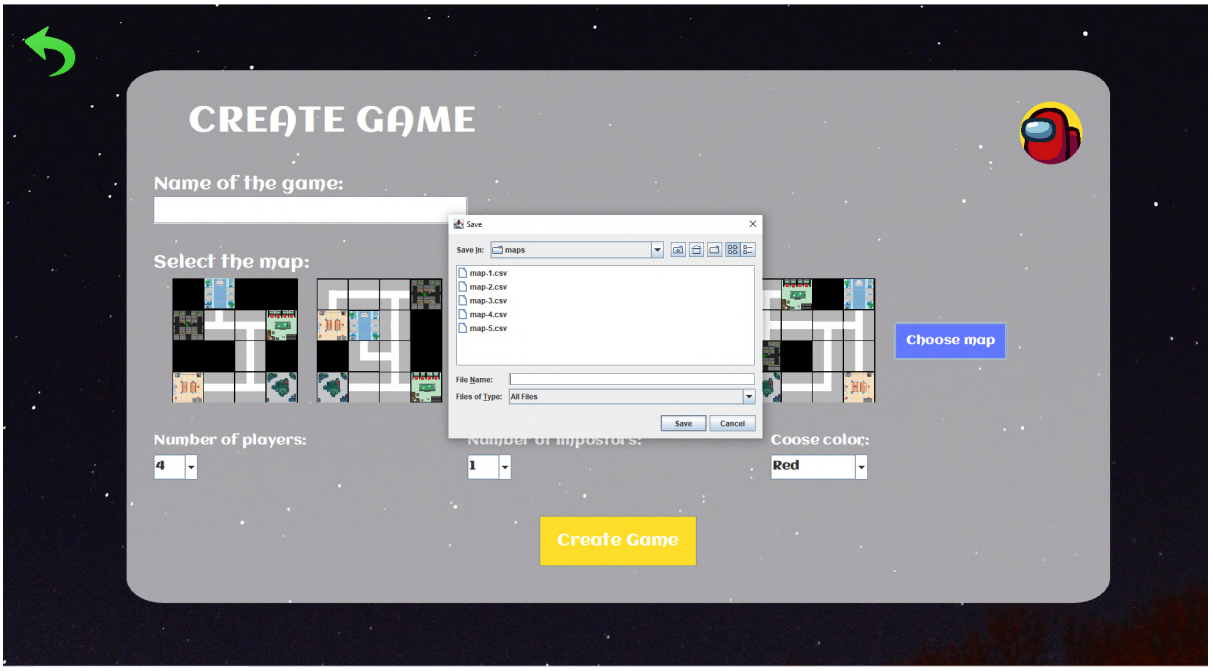
Així a grans trets, la vista està composada de JLabels, els quals són els títols de cada apartat de la vista (CREATE GAME, Name of the game,

Select the map, Number of players, Number of impostors i Choose Color).

Després tenim un JTextField on escriurem el nom de la partida, que després serà passat a la base de dades per guardar-ho.

També tenim un FlowLayout el qual mostra les imatges dels 5 diferents mapes i el JButton del final que posa "Choose map". Es va decidir utilitzar un FlowLayout ja que aquest va col·locant els elements que van a dins seu de manera ordenada una al costat de l'altre.

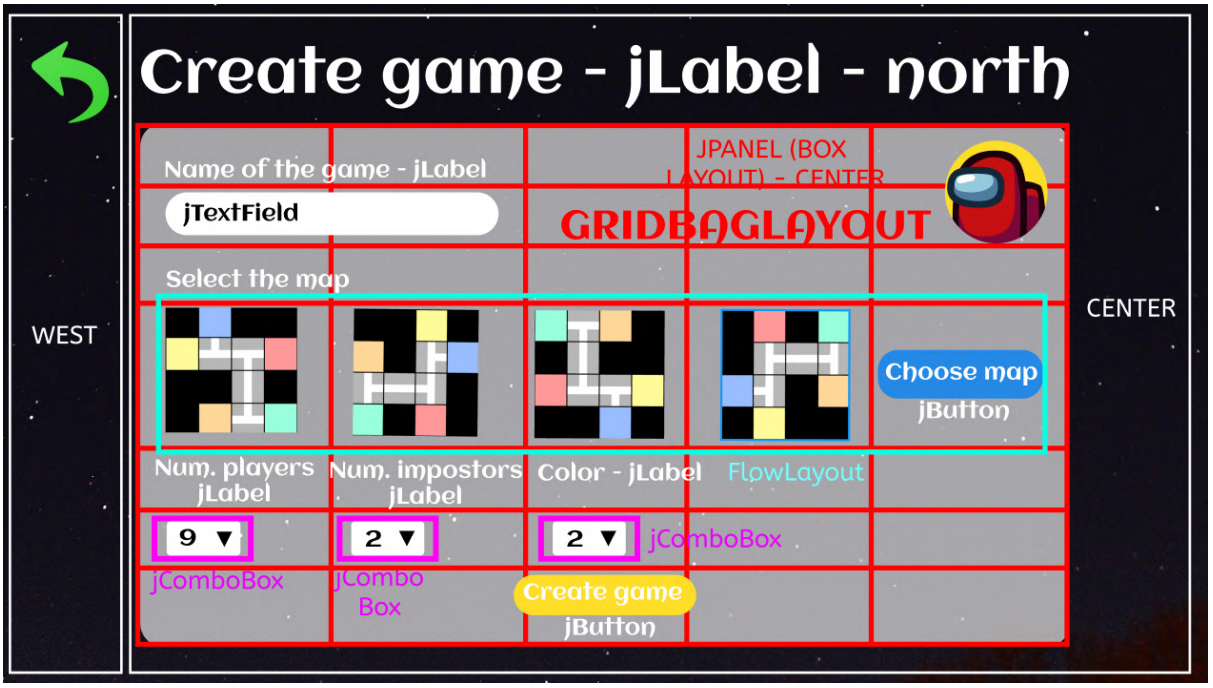
Quan arribem al botó de "choose map" si el pressionem, ens obre un JFileChooser, el qual ens permet triar quin mapa volem jugar dels que tenim guardats, com es pot observar a la següent imatge:



Després baixem unes files més del GridBag Layout i ens trobem amb tres JComboBox, un per triar el nombre de jugadors de la partida, un altre per triar el nombre d'impostors i un últim per triar el color del jugador principal (l'usuari). Vam triar de fer-ho amb JComboBox ja que és molt més fàcil controlar que ens insereix l'usuari.

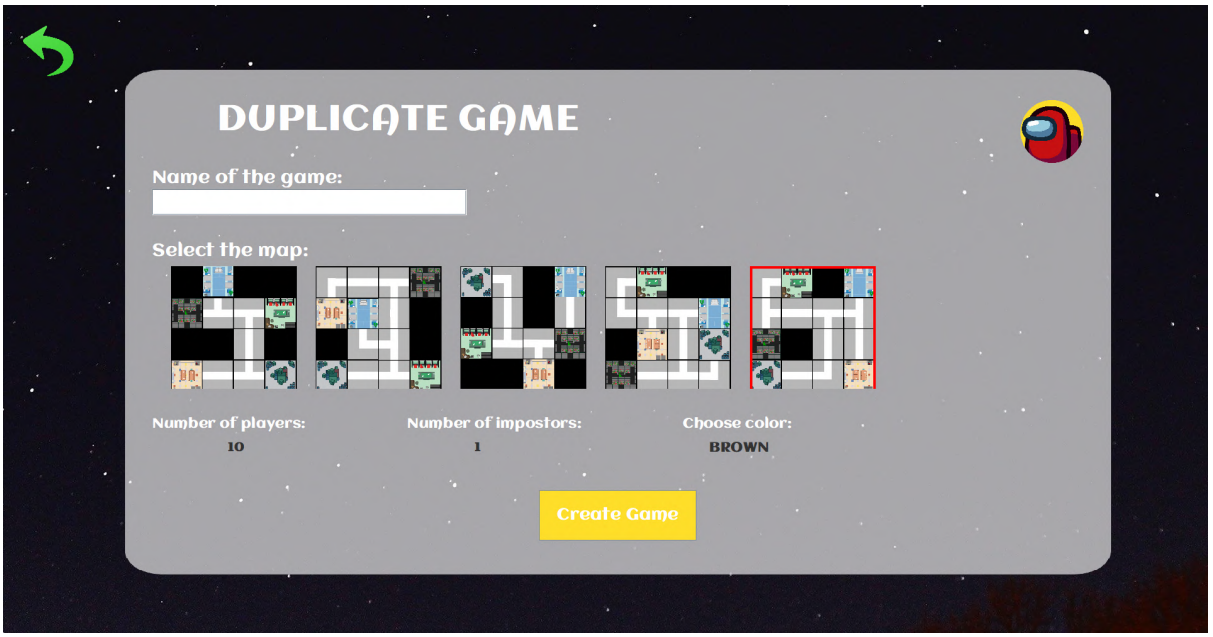
Un cop s'ha omplert tots els camps, s'ha de pressionar el botó "create game" i d'aquesta manera ens retorna a la pàgina de llista de partides, on s'afegeix la nova partida que s'acaba de crear i està llesta per ser jugada.

Aquesta vista, com s'ha pogut observar, està formada per molt layouts, és per això que per a realitzar-la, va anar molt bé fer-se un esquema on es dibuixéssin els diferents layouts, per tal d'organitzar-se millor i tenir una base sobre la qual després codificar. Aquest esquema es va realitzar amb figma i es veu tal que així:

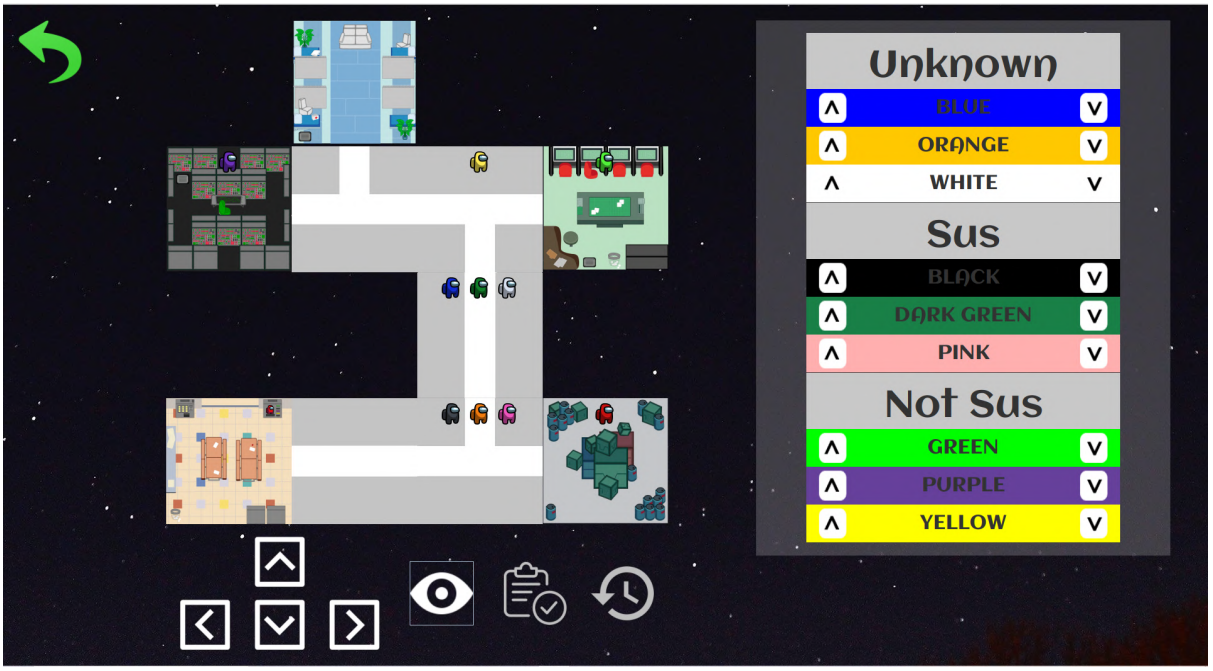


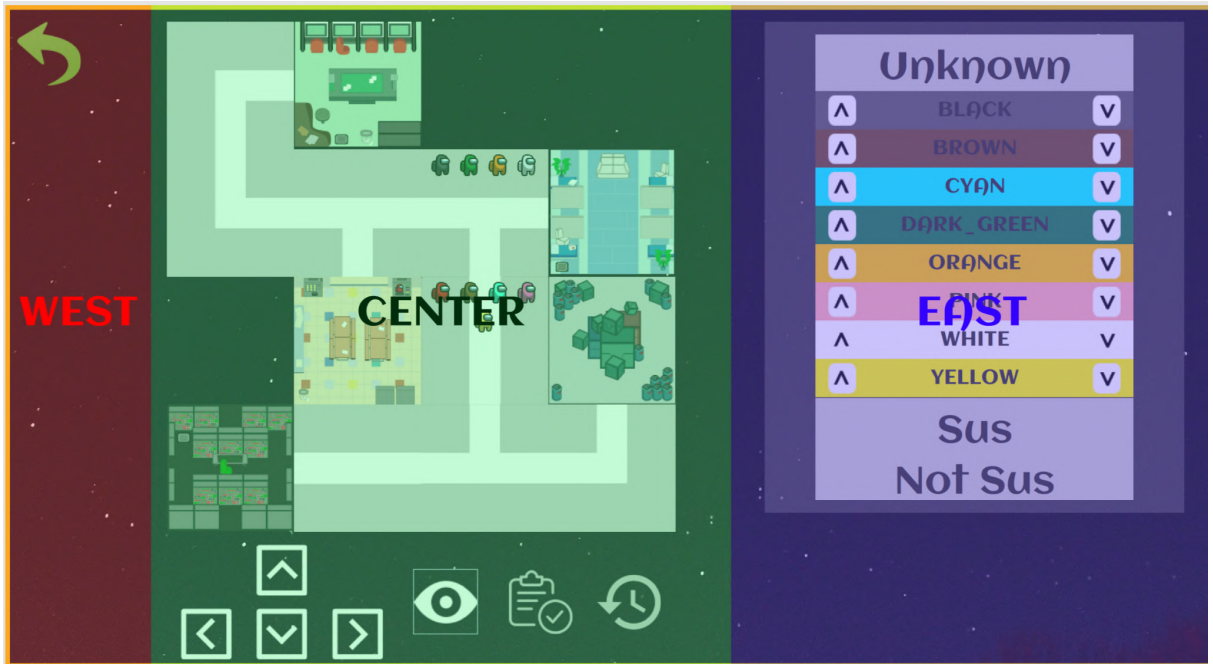
Ara passem a la duplicació de partides:

Per a aquesta, s'ha mantingut exactament el mateix disseny que a la creació d'aquestes, excepte pel fet que no es pot triar el nombre de jugadors, nombre d'impostors, el color del jugador ni el mapa, simplement es mostren per a què vegis les característiques de la partida, però no son modificables. La gràcia està en que l'usuari pugui reutilitzar una configuració de partides que ha creat anteriorment per estalviar-se feina. El mapa, surt remarcat amb vermell per a què l'usuari pugui veure quin mapa es jugarà. Els layouts, com es pot observar a continuació, son exactament iguals per tal de distribuir tots els elements com a la creació de partides:


DUPLICATE GAME
Name of the game:
Select the map:
Number of players:
10
Number of impostors:
1
Choose color:
BROWN
Create Game

- Partida





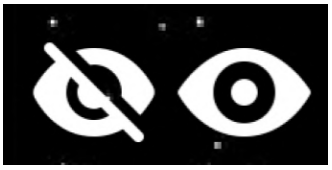
Per a la vista de la Partida (GameView), hem creat un JPanel amb un BorderLayout utilitzant el WEST, CENTER i EAST.

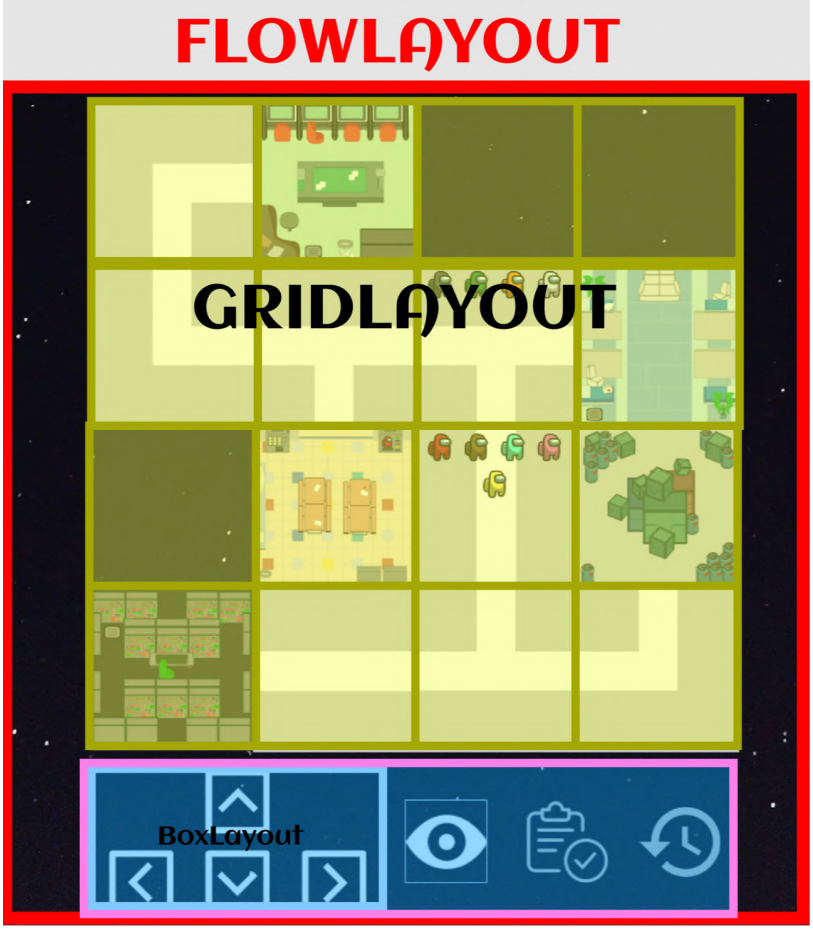
WEST:

En el west afegim un JPanel amb FlowLayout i una imatge de la fletxa per tornar enrera.

CENTER:

En la vista del center tenim dos **JPanels**, el **GridLayout**, en el que tenim el mapa ficat, una imatge per cada casella. Per les caselles buides també estem utilitzant una imatge negre amb opacitat reduïda.
A sota del **GridLayout** tenim els controls del Player en un **JPanel** amb FlowLayout. Dins d'aquest JPanel tenim un altre JPanel amb un BoxLayout amb els controls de moviment i conjuntament el control de les accions del jugador.

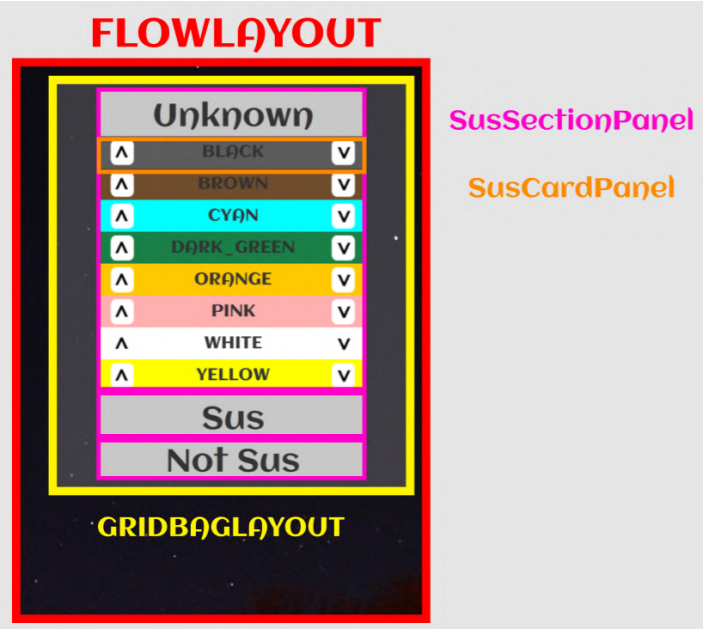


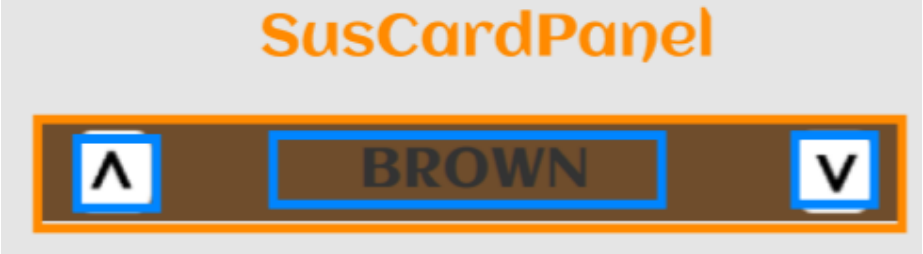


Quan apretem el botó de canviar la vista global a local i viceversa, la imatge del botó també canvia. Els botons de logs i de verificació de la taula de classificació només son activables en la habitació en la que pots realitzar l'acció.

EAST:

En el East tenim un JPanel amb un GridBagLayout.
En aquest GridBagLayout tenim 3 SusSectionPanel.
Aquest SusSectionPanel es un JPanel que té el nom de la secció ("Unkown", "Sus", "Not Sus") i a dins d'aquest JPanel també tenim les SusCardPanel.
Un SusCardPanel té 2 JButtons (per pujar o per baixar) i el nom del NPC a qui li correspon.
D'aquesta manera les SusSectionPanel es van "passant" entre elles els SusCardPanel.





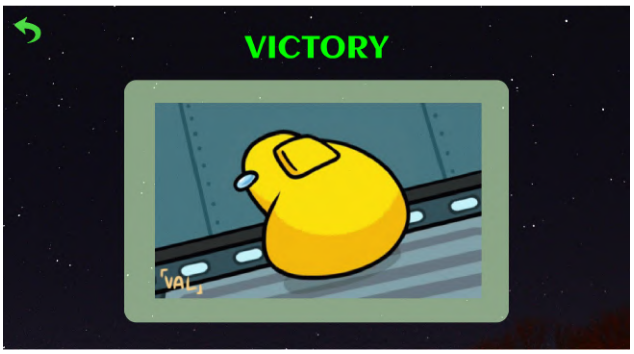
- Taula Logs

Per aquesta vista, s'ha mantingut el border layout principal. A dins el nord s'ha fet un nou JPanel en BorderLayout per ficar el botó JButton de sortir a la zona East del Nord.

Per fer la part del centre, es van fer 3 JLabels, un per cada títol de la taula, i ens van colocar en les posicions pertinents (WEST, CENTER, EAST) dins d'un JPanel amb BorderLayout. També, vem ficar un altre JPanel amb la JTable i el JScrollPane.
La totalitat es va ficar en un altre JPanel amb BorderLayout, el JPanel dels JLabels al nord i el JPanel de la JTable i el JScrollPane al centre.

En quan a la Jtable, vem treure el header i adecuar el nombre de columnes, files i espais amb les funcions pròpies. Amb el CustomTableModel i el DefautTableCellRenderer vem gestionar el color de les files i la data.
CustomTableModel és una classe que hem creat nosaltres. Fa un extends de AbstractTableModel i d'aquesta manera guardem totes les dades sobre les files, les columnes i el color de la fila.

- Pestanyes finals (Victoria / Derrota)

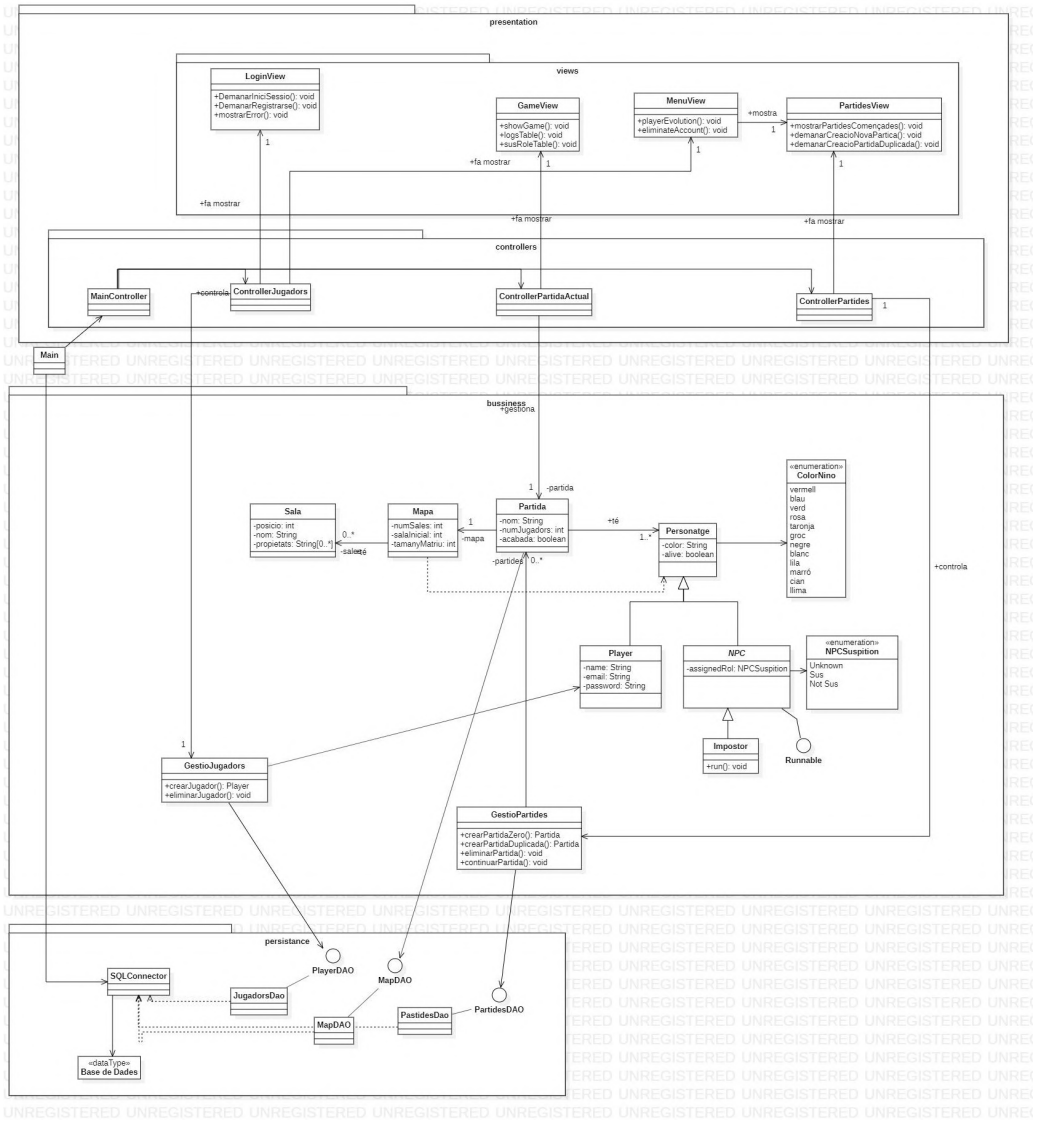


Aquestes dues pestanyes s'han fet exactament igual, només canviant el gif i la imatge de fons.

Tot els fons exceptuant el botó d'enrere i el gif, són una imatge ficada com a fons.
La imatge del fons és un JImagePanel que té un borderLayout. Al West, com de costum es troba el botó de "back" i al centre, en un gridbackLayout es troba el gif.
També s'ha necessitat un empty border.

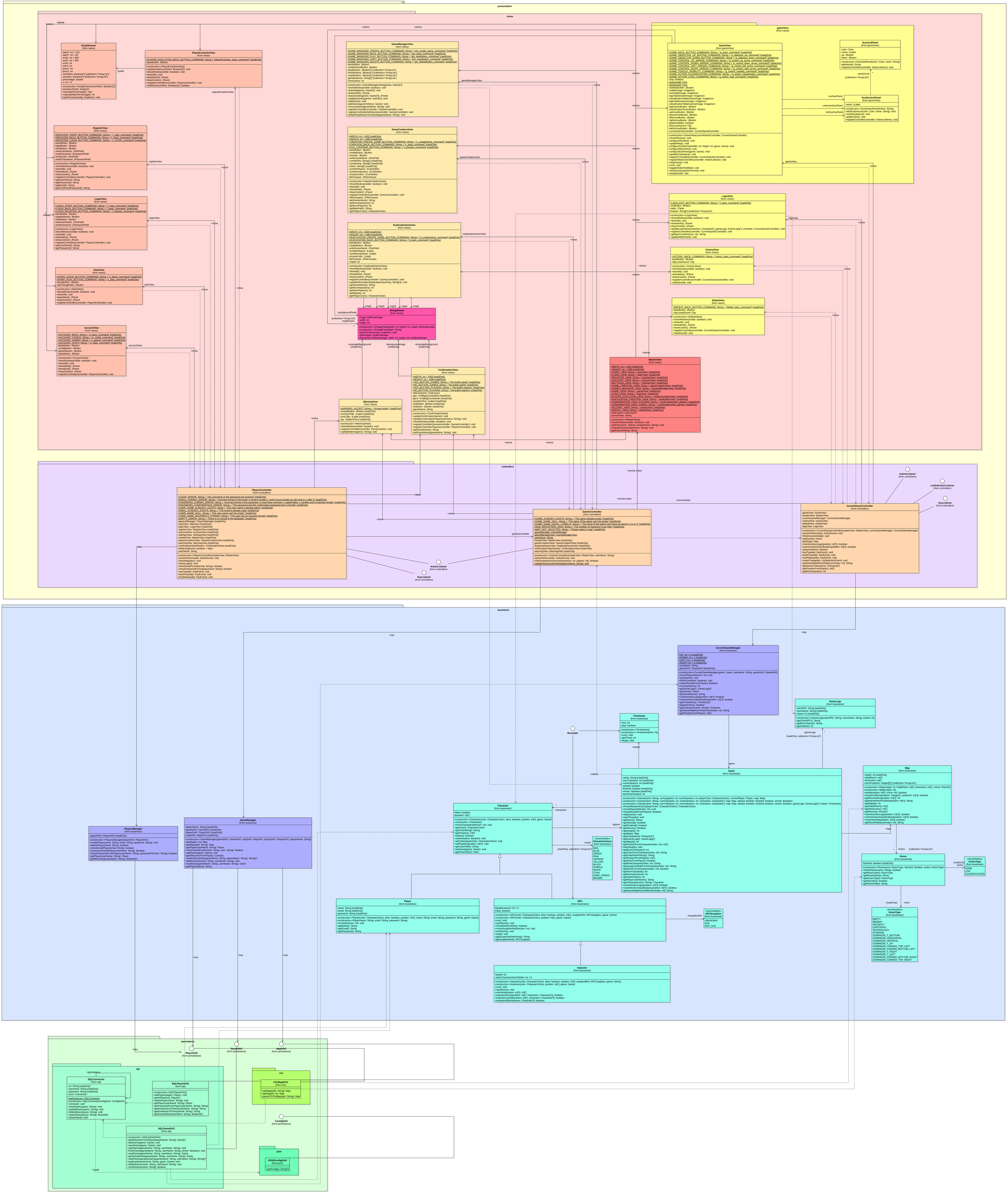
3. Diagrama de classes

UML inicial



La nostra idea principal abans de començar, va ser la següent. En general, la nostra idea s'ha mantingut prou bé, però al final hem decidit fer una classe per cada vista, NO afegir un controller global sino relacionant els diferents controllers... En aquest disseny, ens va faltar considerar la taulaLogs, que no la vam tenir present en el seu moment, i altres detalls com el temps de partida...

UML final:


presentation
views
GraphDrawer
(from views)
-startX: int = 120
-startY: int = 50
-endX: int = 800
-endY: int = 500
-unitX: int
-unitY: int
-prevX: int
-prevY: int
-numWins: Boolean[*] {collection="ArrayList"}
-winsInfo: Boolean[*] {collection="ArrayList"}
-percentage: double
-x: int = 0
+constructor+GraphDrawer(numWins: Boolean[*])
+booleanToInt(): Integer[*]
+calculatePercentage(): void
+calculateMaxPercentage(): int
+paint(component): Graphics): void
PlayerEvolutionView
(from views)
+PLAYER_EVOLUTION_BACK_BUTTON_COMMAND: String = "playerEvolution_back_command" [readOnly]
-backButton: JButton
+constructor+PlayerEvolutionView()
+updateGraph(numWins: Boolean[*]): void
+showWindow(visible: boolean): void
+drawAll(): void
+drawNorth(): JPanel
+drawCenter(): JPanel
+registerController(controller: PlayersController): void
+nuWins(numWins: Boolean[*]): boolean
-graph
-playerEvolutionView
GameManagerView
(from views)
+GAME_MANAGER_CREATE_BUTTON_COMMAND: String = "gm_create_game_command" [readOnly]
+GAME_MANAGER_BACK_BUTTON_COMMAND: String = "gm_back_command" [readOnly]
+GAME_MANAGER_PLAY_BUTTON_COMMAND: String = "gm_playButton_command" [readOnly]
+GAME_MANAGER_COPY_BUTTON_COMMAND: String = "gm_copyButton_command" [readOnly]
+GAME_MANAGER_DELETE_BUTTON_COMMAND: String = "gm_deleteButton_command" [readOnly]
-backButton: JButton
-createGameButton: JButton
-playButtons: JButton[*] {collection="ArrayList"}
-trashButtons: JButton[*] {collection="ArrayList"}
-copyButtons: JButton[*] {collection="ArrayList"}
-gamesNames: String[*] {collection="ArrayList"}
-numGames: int
+constructor+GameManagerView(games: Game[*])
+showWindow(visible: boolean): void
+drawAllgames: Game[*]): void
+drawNorth(): JPanel
+drawCenterGames: Game[*]): JPanel
+updateGames(games: Game[*]): void
+addGame(): void
+addGame(gameToStore: Game): void
+removeGame(gameName: String): void
+registerController(controller: GamesController): void
+registerControllerAddGame(controller: GamesController): void
+deletePlayButtonFromGame(gameName: String): void
-gameManagerView
GameView
(from gameView)
+GAME_BACK_BUTTON_COMMAND: String = "g_back_command" [readOnly]
+GAME_OBJECTIVE_UP_BUTTON_COMMAND: String = "g_objective_up_command" [readOnly]
+GAME_OBJECTIVE_DOWN_BUTTON_COMMAND: String = "g_objective_down_command" [readOnly]
+GAME_CONTROL_UP_ARROW_COMMAND: String = "g_control_up_arrow_command" [readOnly]
+GAME_CONTROL_DOWN_ARROW_COMMAND: String = "g_control_down_arrow_command" [readOnly]
+GAME_CONTROL_LEFT_ARROW_COMMAND: String = "g_control_left_arrow_command" [readOnly]
+GAME_CONTROL_RIGHT_ARROW_COMMAND: String = "g_control_right_arrow_command" [readOnly]
+GAME_TOGGLE_VISIBILITY_COMMAND: String = "g_control_visibility_command" [readOnly]
+GAME_ACTION_CLASSIFICATION_COMMAND: String = "g_action_classification_command" [readOnly]
+GAME_ACTION_LOGS_COMMAND: String = "g_action_logs_command" [readOnly]
-stop: boolean
-raclonical40: Font
-raclonical32: Font
-visibilityButton: JButton
-visibleImage: ImageIcon
-nonVisibleImage: ImageIcon
-logsTableActiveImage: ImageIcon
-classificationTableActiveImage: ImageIcon
-logsTableInactiveImage: ImageIcon
-classificationTableInactiveImage: ImageIcon
-logsActionButton: JButton
-classificationActionButton: JButton
-upArrowButton: JButton
-downArrowButton: JButton
-leftArrowButton: JButton
-rightArrowButton: JButton
-globalVisibility: boolean
-clpLooseSound: Clip
-backArrowButton: JButton
-currentGameController: CurrentGameController
+constructor+GameView(currentGameController: CurrentGameController)
+showWindow(): void
-configureWindow(): void
+updateMap(): void
-configureCenterView(width: int, height: int, game: Game): void
-configureWestView(): void
-configureEastView(game: Game): void
+updateCharacters(): void
+registerController(controller: CurrentGameController): void
+registerWestController(controller: ActionListener): void
+stopGame(): void
+run(): void
+toggleGlobalVisibility(): void
+showSusSuspectincorrect(): void
+stopSound(): void
-gameView
SusCardPanel
(from gameView)
-color: Color
-name: JLabel
-up: JButton
-down: JButton
+constructor+SusCardPanel(color: Color, name: String)
+getName(): String
+registerController(controller: ActionListener): void
-sectionList
{collection="ArrayList"}
*
-susSusPanel
-unknownSusPanel
-notSusSusPanel
SusSectionPanel
(from gameView)
-name: JLabel
+constructor+SusSectionPanel(name: String)
+addSuspicious(color: Color, name: String): void
+resetSuspicious(): void
+updateView(): void
+registerController(controller: ActionListener): void
RegisterView
(from views)
+REGISTER_START_BUTTON_COMMAND: String = "r_start_command" [readOnly]
+REGISTER_BACK_BUTTON_COMMAND: String = "r_back_command" [readOnly]
+REGISTER_LOGIN_BUTTON_COMMAND: String = "r_LOGIN_command" [readOnly]
-backButton: JButton
-loginButton: JButton
-startButton: JButton
-writeUserName: JTextField
-writePassword: JPasswordField
-writeEmail: JTextField
-writeCPassword: JPasswordField
+constructor+RegisterView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: PlayersController): void
+getUserName(): String
+getPassword(): String
+getEmail(): String
+getConfirmedPassword(): String
-registerView
LoginView
(from views)
+LOGIN_START_BUTTON_COMMAND: String = "l_start_command" [readOnly]
+LOGIN_BACK_BUTTON_COMMAND: String = "l_back_command" [readOnly]
+LOGIN_REGISTER_BUTTON_COMMAND: String = "l_register_command" [readOnly]
-backButton: JButton
-registerButton: JButton
-startButton: JButton
-writeUserName: JTextField
-writePassword: JPasswordField
+constructor+LoginView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: PlayersController): void
+getUserName(): String
+getPassword(): String
-loginView
GameCreationView
(from views)
+WIDTH: int = 1920 [readOnly]
+HEIGHT: int = 1080 [readOnly]
+CREATION_CREATE_GAME_BUTTON_COMMAND: String = "c_createGame_command" [readOnly]
+CREATION_BACK_BUTTON_COMMAND: String = "c_back_command" [readOnly]
+FILE_CHOOSER_BUTTON_COMMAND: String = "f_chooser_command" [readOnly]
-backButton: JButton
-createButton: JButton
-chooser: JButton
-writeGameName: JTextField
-numberPlx: String[*] [readOnly]
-numberImp: String[*] [readOnly]
-colors: String[*] [readOnly]
-numberPlayers: JComboBox
-numberImpostors: JComboBox
-chooseColors: JComboBox
-fileChooser: JFileChooser
+constructor+GameCreationView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: GamesController): void
+filechooser(): JFileChooser
+getGameName(): String
+getNumImpostors(): int
+getNumPlayers(): int
+getMapPath(): String
+getPlayerColor(): CharacterColors
-gameCreationView
LogsView
(from views)
+LOGS_EXIT_BUTTON_COMMAND: String = "l_back_command" [readOnly]
-exitButton: JButton
-table: JTable
#values: String[*] {collection="ArrayList"}
+constructor+LogsView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawNorth(): JPanel
+drawCenter(): JPanel
+updateLogsView(characters: Character[*], gameLogs: GameLogs[*], controller: CurrentGameController): void
+registerController(controller: CurrentGameController): void
+getNpcFromRow(row: int): String
+updateAfterEvent(): void
-logsView
DuplicateGameView
(from views)
+WIDTH: int = 1920 [readOnly]
+HEIGHT: int = 1080 [readOnly]
+DUPLICATION_CREATE_GAME_BUTTON_COMMAND: String = "d_createGame_command" [readOnly]
+DUPLICATION_BACK_BUTTON_COMMAND: String = "d_back_command" [readOnly]
-backButton: JButton
-createButton: JButton
-writeGameName: JTextField
-numberPlayers: JLabel
-numberImpostors: JLabel
-chooseColor: JLabel
-fileChooser: JFileChooser
-mapId: int
+constructor+DuplicateGameView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: GamesController): void
+updateInformationDuplicate(arrayString: String[*]): void
+getGameName(): String
+getNumImpostors(): int
+getNumPlayers(): int
+getMapId(): int
+getPlayerColor(): CharacterColors
-duplicationGameView
VictoryView
(from views)
+VICTORY_BACK_COMMAND: String = "victory_back_command" [readOnly]
-backButton: JButton
-clpLooseSound: Clip
+constructor+VictoryView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: CurrentGameController): void
+stopSound(): void
StartView
(from views)
+START_LEAVE_BUTTON_COMMAND: String = "s_leave_command" [readOnly]
+START_PLAY_BUTTON_COMMAND: String = "s_play_command" [readOnly]
-leaveButton: JButton
-startTiangbutton: JButton
+constructor+StartView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: PlayersController): void
-startView
JImagePanel
(from views)
-image: BufferedImage
-width: int
-height: int
+constructor+JImagePanel(width: int, height: int, image: BufferedImage)
+constructor+JImagePanel(path: String)
+paintComponent(g: Graphics): void
+getImage(): BufferedImage
+resize(img: BufferedImage, width: int, height: int): BufferedImage
-backgroundPanel
-map5
-map4
-map3
-map1
-map2
{collection="ArrayList"}
-mapRooms
*
-rectangleBackground
{readOnly}
-backgroundImage
{readOnly}
-rectangleBackground
{readOnly}
AccountView
(from views)
+ACCOUNT_BACK: String = "a_back_command" [readOnly]
+ACCOUNT_CONFIG: String = "a_config_command" [readOnly]
+ACCOUNT_GAMES: String = "a_games_command" [readOnly]
+ACCOUNT_STATS: String = "a_stats_command" [readOnly]
-backButton: JButton
-configButton: JButton
-gamesButton: JButton
-statsButton: JButton
+constructor+AccountView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawEast(): JPanel
+drawCenter(): JPanel
+registerController(controller: PlayersController): void
-accountView
ConfirmationView
(from views)
+WIDTH: int = 1920 [readOnly]
+HEIGHT: int = 1080 [readOnly]
+YES_BUTTON_GAMES: String = "Yes button game" [readOnly]
+NO_BUTTON_GAMES: String = "No button game" [readOnly]
+YES_BUTTON_PLAYERS: String = "Yes button players" [readOnly]
+NO_BUTTON_PLAYERS: String = "No button players" [readOnly]
-WarningView: GridLayout
-gbc: GridBagConstraints [readOnly]
-gbc2: GridBagConstraints [readOnly]
-questionText: JLabel [readOnly]
-yesButton: JButton [readOnly]
-noButton: JButton [readOnly]
-gameName: String
+constructor+ConfirmationView()
+updateConfirmationGame(): void
+updateConfirmationPlayer(userName: String): void
+showWindow(visible: boolean): void
+registerControllerGames(controller: GamesController): void
+registerControllerPlayers(controller: PlayersController): void
+getGameName(): String
+setGameName(gameName: String): void
WarningView
(from views)
+WARNING_ACCEPT: String = "Accept button" [readOnly]
-acceptButton: JButton [readOnly]
-incorrectTitle: JLabel [readOnly]
-errorTitle: JLabel [readOnly]
-jop: JOptionPane [readOnly]
+constructor+WarningView()
+showWindow(visible: boolean): void
+registerController(controller: EventListener): void
+updateWarning(error: String): void
MasterView
(from views)
+WIDTH: int = 1920 [readOnly]
+HEIGHT: int = 1080 [readOnly]
+START_VIEW: String = "startView" [readOnly]
+LOGIN_VIEW: String = "loginView" [readOnly]
+REGISTER_VIEW: String = "registerView" [readOnly]
+ACCOUNT_VIEW: String = "accountView" [readOnly]
+SETTINGS_VIEW: String = "settingsView" [readOnly]
+GAME_CREATION_VIEW: String = "gameCreationView" [readOnly]
+GAMES_MANAGER_VIEW: String = "gamesManagerView" [readOnly]
+GAME_VIEW: String = "gameView" [readOnly]
+LOGS_VIEW: String = "logsView" [readOnly]
+PLAYER_EVOLUTION_VIEW: String = "playerEvolutionView" [readOnly]
+DUPLICATION_CREATION_VIEW: String = "duplicationView" [readOnly]
+CONFIRMATION_VIEW_PLAYERS: String = "confirmationView_players" [readOnly]
+CONFIRMATION_VIEW_GAMES: String = "confirmationView_games" [readOnly]
+VICTORY_VIEW: String = "victoryView" [readOnly]
+DEFEAT_VIEW: String = "defeatView" [readOnly]
-cardLayout: CardLayout
-currentView: String
+constructor+MasterView()
+showWindow(visible: boolean): void
+addView(view: JPanel, constraintView: String): void
+changeView(constantView: String): void
+getCurrentView(): String
DefeatView
(from views)
+DEFEAT_BACK_BUTTON_COMMAND: String = "defeat_back_command" [readOnly]
-backButton: JButton
-clpLooseSound: Clip
+constructor+DefeatView()
+showWindow(visible: boolean): void
+drawAll(): void
+drawWest(): JPanel
+drawCenter(): JPanel
+registerController(controller: CurrentGameController): void
+stopSound(): void
controllers
PlayersController
(from controllers)
+LOGIN_ERROR: String = "The username or the password are incorrect" [readOnly]
+EMAIL_FORMAT_ERROR: String = "Incorrect format of the email. It mustn't contain ( ) and it must contain an (@) and a ( ) after it" [readOnly]
+PASSWORD_FORMAT_ERROR: String = "Incorrect format of the password. It must have minimum 1 capital letter, 1 number and 8 character length" [readOnly]
+PASSWORD_CONFIRMATION_ERROR: String = "The password and the confirmation password don't coincide" [readOnly]
+USER_NAME_ALREADY_EXISTS: String = "This user name is already taken" [readOnly]
+EMAIL_ALREADY_EXISTS: String = "This email is already used" [readOnly]
+USER_NAME_NULL: String = "The user name can't be empty" [readOnly]
+USER_NAME_INCORRECT_FORMAT: String = "The user has an incorrect format" [readOnly]
+EMPTY_GRAPH: String = "There is no record in the database" [readOnly]
-playersManager: PlayersManager {readOnly}
-startView: StartView {readOnly}
-loginView: LoginView {readOnly}
-registerView: RegisterView {readOnly}
-accountView: AccountView {readOnly}
-settingsView: SettingsView {readOnly}
-masterView: MasterView {readOnly}
-playerEvolutionView: PlayerEvolutionView {readOnly}
-warningView: WarningView {readOnly}
-confirmationViewPlayers: ConfirmationView {readOnly}
-isWarningActive: boolean = false
-userName: String
+constructor+PlayersController(masterView: MasterView)
+actionPerformed(e: ActionEvent): void
-checkRegister(): void
-checkLogin(): void
-checkEmailFormat(email: String): boolean
+checkPasswordFormat(password: String): boolean
+keyTyped(e: KeyEvent): void
+keyPressed(e: KeyEvent): void
+keyReleased(e: KeyEvent): void
GamesController
(from controllers)
+GAME_ALREADY_EXISTS: String = "This game already exist" [readOnly]
+GAME_NAME_NULL: String = "The name of the game can't be empty" [readOnly]
+GAME_NAME_EQUAL_COMILLA: String = "The name of the game can't have an equal (=) or a (?)" [readOnly]
+NUM_IMPOSTORS_HIGH: String = "The number of impostors is too high" [readOnly]
+MAP_NOT_SELECTED: String = "Please select a map" [readOnly]
-gameManager: GameManager
-gameManagerView: GameManagerView
-userName: String
-masterView: MasterView {readOnly}
-gameCreationView: GameCreationView {readOnly}
-duplicateGameView: DuplicateGameView {readOnly}
-confirmationViewGames: ConfirmationView {readOnly}
-warningView: WarningView {readOnly}
+constructor+GamesController(masterView: MasterView, userName: String)
+actionPerformed(e: ActionEvent): void
-checkImpostorsAmountsIimpostors: int, players: int): boolean
+updateGameJustFinished(gameName: String): void
CurrentGameController
(from controllers)
-gameView: GameView
-masterView: MasterView
-currentGameManager: CurrentGameManager
-victoryView: VictoryView
-defeatView: DefeatView
-logsView: LogsView
+constructor+CurrentGameController(masterView: MasterView, currentGameManager: CurrentGameManager)
+actionPerformed(e: ActionEvent): void
+finishGameDefeat(): void
+getGame(): Game
+getMap(): Map
+roomIsActionLogs(position: int[*]): boolean
+roomIsActionClassification(position: int[*]): boolean
+playerIsAlive(): boolean
+keyTyped(e: KeyEvent): void
+keyPressed(e: KeyEvent): void
+keyReleased(e: KeyEvent): void
+valueChanged(e: ListSelectionEvent): void
+getGameMapRoomPath(roomIndex: int): String
+getGameCharacters(): Character[*]
+getPositionFromPlayer(): int[*]
+getNumImpostor(): int
ActionListener
(from controllers)
ListSelectionListener
(from controllers)
KeyListener
(from controllers)
+canvia vistes
+canviaVistes
+canvia vistes
-gathersController
ActionListener
(from controllers)
KeyListener
(from controllers)
bussiness
CurrentGameManager
(from bussiness)
+UP: int = 0 [readOnly]
+DOWN: int = 1 [readOnly]
+LEFT: int = 2 [readOnly]
+RIGHT: int = 3 [readOnly]
-userName: String
-gameDAO: GameDAO {readOnly}
+constructor+CurrentGameManager(game: Game, userName: String, gameDAO: GameDAO)
+movePlayer(direction: int): void
+stopGame(): void
+finishGame(win: boolean): void
+checkResultsFromPlayer(): Boolean
+numImpostors(): int
+getGameLogs(): GameLogs[*]
+getGame(): Game
+getGameName(): String
+roomIsActionLogs(position: int[*]): boolean
+roomIsActionClassification(position: int[*]): boolean
+getCharacters(): Character[*]
+playerIsAlive(): boolean
+getCharacter(name: String): Character
+getGameMapRoomPath(roomIndex: int): String
+getPositionFromPlayer(): int[*]
+has
+has
+has
+has
+has
+has
GameLogs
(from bussiness)
-colorNPC: String {readOnly}
-roomName: String {readOnly}
-instant: int {readOnly}
+constructor+GameLogs(colorNPC: String, roomName: String, instant: int)
+getColorNPC(): String
+getRoomName(): String
+getInstant(): int
-gameLogs
{readOnly, collection="ArrayList"}
*
TimeGame
(from bussiness)
-time: int
-stop: boolean
+constructor+TimeGame()
+constructor+TimeGame(time: int)
+run(): void
+getTime(): int
+stop(): void
Runnable
-instant
-game
Game
(from bussiness)
-name: String {readOnly}
-numTripulants: int {readOnly}
-numImpostors: int {readOnly}
-started: boolean
-finished: boolean {readOnly}
-winner: Boolean {readOnly}
-lastTimeChecked: int
+constructor+Game(name: String, numJugadors: int, numImpostors: int, playerColor: CharacterColors, currentPlayer: Player, map: Map)
+constructor+Game(name: String, numTripulants: int, numImpostors: int, characters: Character[*], map: Map, started: boolean, finished: boolean, winner: Boolean)
+constructor+Game(name: String, numTripulants: int, numImpostors: int, characters: Character[*], map: Map, started: boolean, finished: boolean, winner: Boolean, gameLogs: GameLogs[*], instant: TimeGame)
-chooseRandomColor(playerColor: CharacterColors): CharacterColors
+movePlayer(direction: int): void
+checkResultsFromPlayer(): Boolean
+stopGame(): void
+startThreads(): void
+getName(): String
+getStarted(): boolean
+getFinished(): boolean
+getWinner(): Boolean
+getInstant(): int
+getMap(): Map
+getCharacters(): Character[*]
+getGameLogs(): GameLogs[*]
+getMapId(): int
+getPositionFromCharacter(index: int): int[*]
+hasStarted(): void
+getNumCharacters(): int
+getColorFromCharacter(index: int): String
+getColorPlayerString(): String
+getPositionFromPlayer(): int[*]
+getAliveFromPlayer(): boolean
+getRoleCharacter(index: int): String
+getSuspiciorRoleFromCharacter(index: int): String
+getAliveFromCharacter(index: int): boolean
+getNumTripulants(): int
+getNumImpostors(): int
+getIndexPlayer(): int
+getPlayerUserName(): String
+getCharacter(name: String): Character
+roomIsActionLogs(position: int[*]): boolean
+roomIsActionClassification(position: int[*]): boolean
+getGameMapRoomPath(roomIndex: int): String
+utilitza
Map
(from bussiness)
-mapId: int {readOnly}
-initialRoom: int[*]
-dimension: int[*]
-roomPositions: Integer[][*] {collection="ArrayList"}
+constructor+Map(mapId: int, initialRoom: int[*], dimension: int[*], rooms: Room[*])
+constructor+Map(mapId: int)
+canMove(pos: int[*], move: int): boolean
+checkPosition(position1: Integer[*], position2: int[*]): boolean
+getRoomIndex(position: int[*]): int
+getNameRoomPosition(position: int[*]): String
+getMapId(): int
+getInitialRoom(): int[*]
+getDimension(): int[*]
+getRooms(): Room[*]
+roomIsActionLogs(position: int[*]): boolean
+roomIsActionClassification(position: int[*]): boolean
+roomHasVent(position: int[*]): boolean
+getRoomPath(roomIndex: int): String
-map
-rooms
{collection="ArrayList"}
*
Character
(from bussiness)
#alive: boolean
#position: int[*]
+constructor+Character(color: CharacterColors, alive: boolean, position: int[*], game: Game)
+constructor+Character()
+moveCharacter(direction: int): void
+getColor(): CharacterColors
+getColorString(): String
+getPosition(): int[*]
+isAlive(): boolean
+setAlive(alive: boolean): void
+setColor(playerColor: CharacterColors): void
+setPosition(position: int[*]): void
+getColorPath(): String
+setGame(game: Game): void
+getColorColor(): Color
-characters
*
#game
#color
{readOnly, collection="ArrayList"}
«enumeration»
CharacterColors
(from bussiness)
RED
BLUE
GREEN
PINK
ORANGE
YELLOW
BLACK
PURPLE
WHITE
CYAN
DARK_GREEN
BROWN
Room
(from bussiness)
-hasVent: boolean {readOnly}
+constructor+Room(room: RoomType, hasVent: boolean, action: ActionType)
+checkName(name: String): boolean
+getRoomType(): RoomType
+getRoomName(): String
+getActionType(): ActionType
+getHasVent(): boolean
+getRoomPath(): String
{readOnly}
-action
«enumeration»
ActionType
(from bussiness)
NONE
LOG
CLASSIFICATION
{readOnly}
-room
«enumeration»
RoomType
(from bussiness)
EMPTY
MEDBAY
SECURITY
CAFETERIA
TECHNOLOGY
STORAGE
CORRIDOR_T_BOTTOM
CORRIDOR_HORIZONTAL
CORRIDOR_VERTICAL
CORRIDOR_T_UP
CORRIDOR_CORNER_TOP_LEFT
CORRIDOR_CORNER_BOTTOM_LEFT
CORRIDOR_T_RIGHT
CORRIDOR_T_LEFT
CORRIDOR_CORNER_BOTTOM_RIGHT
CORRIDOR_CORNER_TOP_RIGHT
Player
(from bussiness)
-name: String {readOnly}
-email: String {readOnly}
-password: String {readOnly}
+constructor+Player(color: CharacterColors, alive: boolean, position: int[*], name: String, email: String, password: String, game: Game)
+constructor+Player(name: String, email: String, password: String)
+move(direction: int): void
+getName(): String
+getEmail(): String
+getPassword(): String
NPC
(from bussiness)
#lastMovement: int = 0
#stop: boolean
+constructor+NPC(color: CharacterColors, alive: boolean, position: int[*], susipitionRol: NPCSuspition, game: Game)
+constructor+NPC(color: CharacterColors, position: int[*], game: Game)
+run(): void
+spcMoves(): void
+checkRoleSusRole(): boolean
+moveSuspitionRol(direction: int): void
+endGame(): void
+stop(): void
+getSuspicionRoleString(): String
+getSuspitionRol(): NPCSuspition
#suspitionRol
«enumeration»
NPCSuspition
(from bussiness)
UNKNOWN
SUS
NOT_SUS
Impostor
(from bussiness)
-lastKill: int
-indexCharacterNextToKiller: int = 0
+constructor+Impostor(color: CharacterColors, alive: boolean, position: int[*], assignedRol: NPCSuspition, game: Game)
+constructor+Impostor(color: CharacterColors, position: int[*], game: Game)
+run(): void
+spcMoves(): void
+useVent(position: int[*]): int[*]
+impostorAlone(position: int[*], characters: Character[*]): boolean
+impostorCanKill(position: int[*], characters: Character[*]): boolean
+impostorKill(characters: Character[*]): boolean
persistance
PlayerDAO
(from persistance)
GameDAO
(from persistance)
MapDAO
(from persistance)
ConfigDAO
(from persistance)
sql
+getInstance
SQLConnector
(from sql)
-url: String {readOnly}
-username: String {readOnly}
-password: String {readOnly}
-conn: Connection
+getInstance(): SQLConnector
+constructor+-SQLConnector(configDAO: ConfigDAO)
+connect(): void
+insertQuery(query: String): void
+updateQuery(query: String): void
+deleteQuery(query: String): void
+selectQuery(query: String): ResultSet
+disconnect(): void
SQLPlayerDAO
(from sql)
+constructor+SQLPlayerDAO()
+addPlayer(player: Player): void
+getAllPlayers(): Player[*]
+deletePlayer(name: String): void
+getPlayer(userName: String): Player
+getPasswordFromPlayer(userName: String): String
+getNameUserToCheck(userName: String): String
+getEmailUserToCheck(email: String): String
+getGamesWins(userName: String): Boolean[*]
SQLGameDAO
(from sql)
+constructor+SQLGameDAO()
+getAllGamesFromPlayer(playerName: String): Game[*]
+addGame(game: Game): void
+saveGame(game: Game): void
+startGame(gameName: String, userName: String): void
+finishGame(gameName: String, userName: String, winner: Boolean): void
+readGame(gameName: String, userName: String): Game
+getGameName(gameName: String, userName: String): String
+readInfoDuplicationGame(gameName: String, userName: String): String[*]
+duplicateGame(name: String, game: Game): void
+deleteGame(name: String, userName: String): void
+checkGame(name: String): boolean
+reads
csv
CSVMapDAO
(from csv)
+getMap(path: String): Map
+getMapId: int): Map
-parseCSVToMap(path: String): Map
json
JSONConfigDAO
(from json)
+getConfig(): String[*]
+has
+has
+has

4. Metodologia de desenvolupament

Per aquest apartat, volem comentar que, en tots els sprints, hem anat fent trucades (reunions) per decidir qui feia cada cosa, i per anar veient com anava el progrés. També, ens connectàvem sovint per resoldre dubtes.

Hem anat avançant el projecte amb el git, per així, que tot s'anés actualitzant a tothom a mesura que s'avançava. Ens hem comunicat principalment en un servidor de discord, però també, hem fet ús d'un grup de whatsapp.

- SPRINT 1

  - Blocks escollits

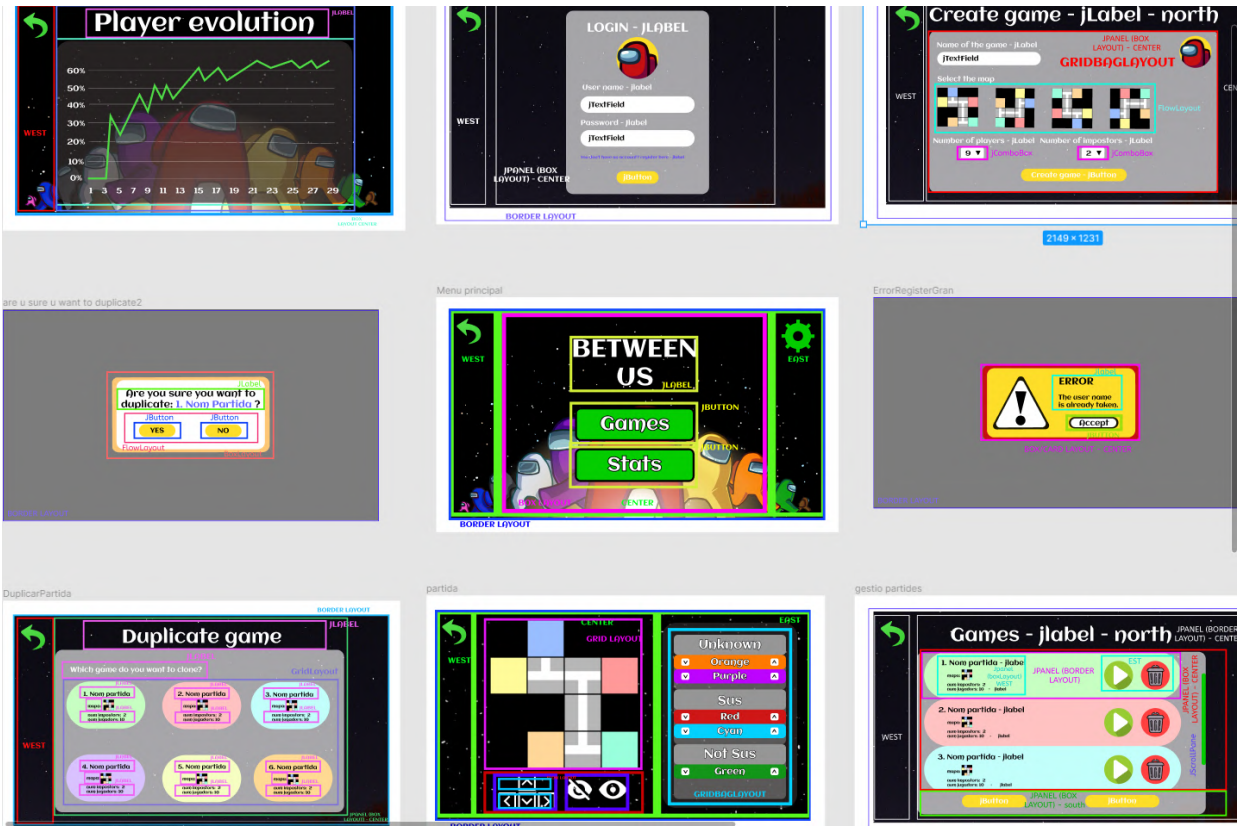
    Per aquest sprint no vam escollir cap block al ser igual per tots els grups.

  - Distribució de feina

    Com el disseny del programa era importartant per començar amb bon peu, vam decidir que cadascú fes un disseny (sense mètodes, en general l'estructura de classes i packages) i posteriorment el posaríem en comú per fer el UML final en el que tots estiguéssim d'acord.

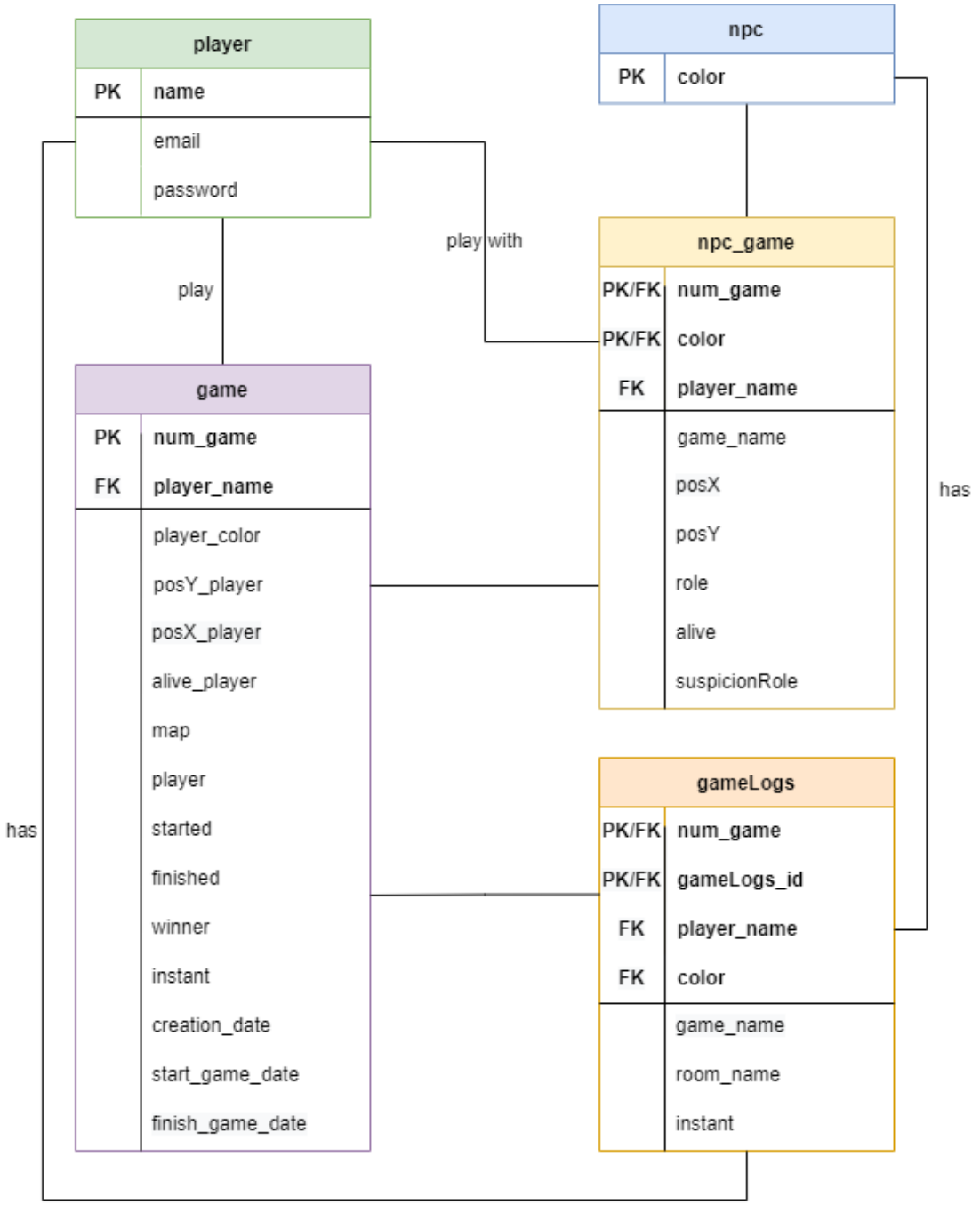
    Per el disseny de les pàgines, tothom va fer unes dues pestanyes també amb els seus respectius possibles layouts.

    A més a més, la Laura es va encarregar del disseny de la Base de dades, l'Albert la va crear, L'Anna i la Marta van fer els dissenys dels mapes i les sales i el Guillem va fer el "Config".



https://www.figma.com/file/NWhWufBiY38elEiJHUWtvy/DPO_practica2_mockups?node-id=14%3A2

- Disseny de la Base de Dades



- ## SPRINT 2

  - <u>Blocks escollits</u>

    - Database info
    - User sign-up GUI
    - User sign-up logic
    - User login GUI
    - User login logic
    - Map file format
    - Map file loading
    - Game creation logic
    - Game creation GUI
    - Game map GUI

  - <u>Distribució de feina</u>

    Per aquest sprint, vam decidir que el Guillem connectaria la Base de Dades amb el programa i el format del fitxer CSV del mapa.
    Els demés membres del grup ens vam ficar en parelles. El primer grup (Marta i Laura), ens tocava tant la vista del Login com la lògica. L'altre grup (Anna i Albert), anaven a fer la lògica i la vista del registre.

    Al final, el 1r grup va fer les dues vistes i l'altre grup la lògica.

    El Guillem va aconseguir acabar abans que els demés i va començar a avançar la vista de la partida actual.

    La Laura també va anar avançant, fent la vista de l'inici i del compte del jugador.

SPRINT 3

- <u>Blocks escollits</u>

  - Game Creation GUI -> ALBERT
  - Game Manager GUI -> LAURA
  - Game Creation logic (Crear una partida) -> GUILLEM
  - Game Management logic (Duplicar partides, etc ...) -> MARTA
  - Arreglar els Warnings del Login i register (sprint 2) -> MARTA
  - Taula Logs GUI -> ANNA
  - Players Action GUI -> GUILLEM
  - PLayers Action Logic -> LAURA
  - NPC simulation logic -> ALBERT
  - Game Objectif GUI -> GUILLEM
  - Game OBjective logic -> ANNA

- <u>Distribució de feina</u>

  La distribució inicial de la feina són els noms que hem ficat a cada tasca. L'assignació de tasques en si, no es van veure gaire modificades. Ara, aquí, es detallarà coses específiques que van canviar durant el sprint.

  (Si alguna cosa no apareix, vol dir que la persona assignada va anar fent els seus apartats i/o els va completar)

  El Guillem va fer la vista de del Action GUI amb el mapa i les sospites del NPC tot en una sola vista, agrupant tot el que calia incorporar-se en la partida. I per fer les sospites de la partida, ja va fer la lògica de canviar de UNKNOWN a SUS o NOT_SUS

  L' Albert va fer els threads i els moviments del diferents NPC però, com la seva assignació era prou extensa, vam decidir que no fes la part especial dels impostors.

  La Laura, va fer també molta lògica amb la base de dades. Va crear un mètode per llegir totes les partides de la base de dades per poder fer la seva vista, ,va fer la lògica addGame per quan es creava una nova partida (tant normal com duplicada), també, la lectura d'una partida en concret per poder-la jugar, i la lògica per guardar-la (ja començada, i per així, poder tornar a on era).

• SPRINT 4

- <u>Blocks escollits</u>

  Delete user account lògica -> MARTA
  Arreglar GUI delete player -> MARTA
  Estadístiques GUI -> ANNA I MARTA
  Estadístiques lògica -> ANNA I MARTA
  Memòria -> TOTS ELS MEMBRES
  Kill impostor lògica -> MARTA
  Ventilació impostor lògica -> ANNA
  Lògica end Game -> LAURA
  NPC movement -> ALBERT
  Threads current game -> GUILLEM
  Logout -> LAURA

- <u>Distribució de feina</u>

Pel que fa a la distribució de la feina per aquest sprint final no es va veure gaire modificada.

Per aquest sprint es va decidir que l'Anna i la Marta s'ocupessin més de la part de la lògica principal que quedava, ja que elles mateixes van preferir avançar més al adonar-se que els altres membres del grup havien avançat bastant en l'últim sprint.

Els demés, van anar intentant arreglar els errors que trobàvem, millorant certes parts del codi, i acabant certs detalls... La Marta i l'Anna van anar fent la lògica i la vista de les estadístiques mentres que els demés, van acabar de fer les dues vistes que quedaven, la de victoria i la de derrota.

La memòria es va anar redactant de forma paral·lela amb els últims retocs del codi final. Pel que fa a les vistes, cadascú va realitzar una breu descripció dels components utilitzats.

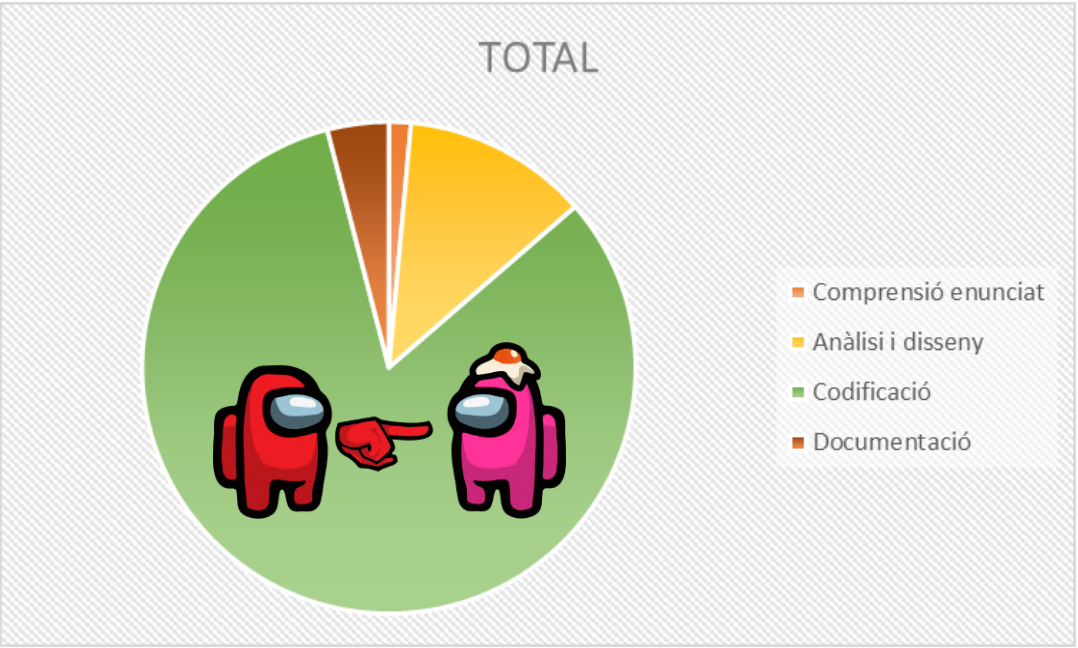
A més a més, la Laura va afegir un apartat en específic per explicar l'argumentació del disseny i l'evolució que hi havia hagut entre el disseny inicial del figma amb el resultat final del joc.

5. Dedicació

- Taula (Temps aproximat)

	Comprensió enunciat	Anàlisi i disseny	Codificació	Documentació
Albert Tomàs	1h	6h	56h	2h
Anna Ferrer	1h	12h	50h	1:30h
Marta Vilaseca	1h	12h	50h	1:30h
Guillem Roselló	1h	6h	65h	4h
Laura Pascual	1h	6,5h	68h	5h
TOTAL	5h	42,5h	289h	14h

- Gràfic



• Argumentació

- ## Comprensió de les especificacions

  En general, aquest apartat era llegir amb deteniment l'enunciat i cap de nosaltres hem hagut d'emprar massa temps per entendre'l

- ## Anàlisi i disseny

  En el anàlisi i el disseny, és per un igual per tots els membres. Perquè tal i com hem explicat, ens vam dividir la feina de manera bastant equitativa.

  La Marta i l'Anna van dedicar molt de temps a crear els detallats mapes del nostre joc. Tot amb illustrator i photoshop.

- ## Codificació

  ALBERT: Pel que fa la codificació, he dedicat la majoria d'hores al moviment dels NPCs (tripulants i impostors), fent que compleixin el que es demana i fent ús de threads per poder realitzar la tasca. El que més va costar va ser aconseguir que tinguessin comportaments normals, així com penal·litzar que un tripulant tingués menys probabilitats de tornar a la sala de la que venia. Al final, però tot va acabar sortint. També tota la part del Game Creation View (creació de partides) vaig dedicar algunes hores, vaig tenir alguns problemes amb swing i que a vegades no feia el que volia però també va acabar sortint bé, i es va poder reutilitzar la vista per la de duplicació de partides, que és gairebé igual. Després també he dedicat temps a ajudar a alguns companys sempre que he pogut

  ANNA: En quan a la codificació, he realitzat els diferents apartats que s'han comentat. En alguns casos, m'he topat amb alguns problemes que m'han fet avançar mes lent: la novetat de la gràfica del jugador, perdre algunes parts de codi o ficar els colors dels NPC a la JTable, entre altres. Per poder concentrarme en la lògica mes principal per l'últim sprint, el Guillem va acabar el tema dels

colors a la Jtable. També, per aquest últim sprint, la Marta i jo vem treballar bastant en equip, compartiem apartats i lògiques similar. Finalment, com tots, vaig revisar, corregir i afegir varies coses al codi final.

MARTA: Pel que fa a la codificació, ha anat realitzant els diferents apartats comentats als sprints. En alguns apartats he hagut de dedicar més temps que en altres donat que m'he trobat en diferents problemes en la programació, tot i així generalment he disposat de l'ajuda del grup, en concret de la Laura, per qualsevol inconvenient.
Un dels majors inconvenients han estat les vistes al pasar de Mac a Windows, donat que no se'm veien correctament i per fer comprovacions havia de dependre d'un altre integrant del grup.

GUILLEM: He realitzat les parts de Sprint que se m'han assignat i sovint ajudava els meus companys amb algun dubte que hagin tingut. Per les LogsView vaig estar bastant de temps mirant com fer que es fiquessin bé els colors i fer que funciones el click.

LAURA: Degut a la gran quantitat d'hores que he estat, explicaré la meva dedicació en aquest apartat. Principalment, el que més he tocat han estat vistes i persistència en les games, feina que m'ha estat assignada. Però, per li he dedicat tantes hores a la codificació ja que, molts cops he ajudat als meus companys i he anat supervisant que tot anés funcionant, tant la cooperació entre membres com en la codificació. També anava revisant el codi per apuntar-me les coses que no quadraven per arreglar-les. Per exemple, vam veure que no teniem cap manera de guardar les Logs i per aquest motiu vaig modificar la BBDD, els mètodes de connexió amb aquesta i vaig afegir la classe logs i la del temps.

- ## Documentació.

Volem aclarar en aquest apartat que, tot i que les parts grupals les hagi escrit algú en concret, tots els demés hem comprovat i si ha estat necessari, s'ha modificat per estar tots d'acord.

ALBERT: He contribuït amb totes les parts que em corresponen en quant a les vistes i també m'he encarregat de que quedés la memòria ben tabulada i ordenada.

ANNA: En quan a la meva contribució, he redactat els apartats que se'm va assignar (menys graf) i he actualizat el disseny de la base de dades.

MARTA: He estat redactant els diferents apartats que se'm van assignar en referència a les vistes i a més a més he realitzat parts comunes.

GUILLEM: He realitzat les meves parts individuals a més de l'explicació de la GameView i part de la LogsView i la WarningView. També he estat fent gran part del diagrama UML, cosa que m'ha ocupat bastant temps.

LAURA: Per a facilitar la feina dels demés, vaig fer el format del word per a que, quan els altres membres escrivissin la memòria, només haguessin d'anar a aquell apartat i escriure-ho. Per tant, vaig preparar els titulars, la portada, l'índex... També em vaig encarregar de fer el resum del projecte a part de les meves parts individuals. Finalment, he fet gran part del diagrama UML, que és una de les coses que m'ha tret més temps.

6. Conclusions

Finalment podem arribar a la conclusió que a través d'aquest projecte hem pogut aplicar tots els coneixements adquirits a classe i els hem pogut posar en pràctica gràcies a la realització d'un projecte més complex com n'és el cas de Between Us.

L'apartat que més dificultat ens ha suposat, en major part, ha estat les vistes, donat que ens donava molt problemes a l'hora d'ordenar l'estructura de la vista i que aquesta es mostres tal i com estavem programant.
Tot i així abans de començar a programar vam dissenyar un figma, per tal de saber com voldríem que fossin les vistes finals i quins elements s'utilitzarien per programar-les, gràcies a això vam afrontar aquests problemes amb més facilitat.

Després de solucionar tots els problemes relacionats amb el Swing, estem molt contents amb el resultat final obtingut, donat que hem aconseguit el disseny inicial proposat.

A més a més, els threads també ens han suposat un problema, donat que ens apareixien molts lags, a vegades el player es movia saltant sales i els NPC també. Nosaltres teniem la convicció que eren problemes pels threads, que al activar tants podria haber un bug. Però, realment, després de molt de temps buscant, vam trobar que era per els listeners. Nosaltres pensavem que el setActionCommand dels listeners esborrava l'anterior i ficava el nou valor (com sol passar amb els sets de les variables), però realment, feia que els altres botons no nous, detectessin que es pressionaven varis cops.
Per tant, quan creàvem una partida i l'afegíem, les anteriors partides, si es pressionava el play, es runnejaven dos partides i per això la posició dels NPC feien coses rares (en una partida es movien a un lloc i en l'altre en un altre...).

També va passar algo semblant amb els botons de UNKNOWN, SUS i NOTSUS, fent que els botons del player es multipliquessin (com a lògica).

Si haguéssim de tornar a realitzar el projecte millloraríem diferents apartats de la pràctica, per tal de perfeccionar el codi al 100%, però en general, estem molt contents dels nostres resultats.

Pel que fa a l'organització del projecte, creiem que ha estat correcte, gràcies al jira ens ha sigut molt més fàcil portar-lo al dia i a través de la tutorització del Pol, hem pogut acabar el projecte a temps.

Com a extres, hem volgut afegir algunes animacions (gif al final de la partida, dos diferents segons si s'ha guanyat o perdut). A més a més, hem afegit música durant la partida per tal de fer més divertit el joc, i també, quan s'acaba..

Per tal d'assegurar-nos que tot funcionava correctament vam estar realitzant diferents proves, en trobar errors, ens distribuíem la feina i els anàvem arreglant.

En conclusió estem contents amb el resultat final que hem assolit de la pràctica. Creiem que hi ha un molt bon treball darrera, tant individual com de grup. També comentar que gràcies a aquesta pràctica, hem après molt i sobretot, hem assolit millor els coneixements de classe.

7. Bibliografia consultada

SWING

- https://es.acervolima.com/java-swing-mateborde/
- https://www.javatpoint.com/java-swing
- https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html
- http://www.chuidiang.org/java/layout/GridBagLayout/GridBagLayout.php
- Apunts de swing i listeners de la Salle/ Estudy
- https://www.w3schools.com/java/default.asp
- .

ALTRES

- Hem fet ús dels Oracle per entendre i buscar mètodes. No especificament el de la classe String, però volíem posar un exemple de tipus de pàgina. https://docs.oracle.com/javase/7/docs/api/java/lang/String.html
- Hem buscat molts dels nostres dubtes en forums de programacio com https://stackoverflow.com/