

ADMIN PANEL COMPONENT

Matteo Belenchia, Marcin Massalski

POLITECHNIKA ŁÓDZKA, INTERNATIONAL FACULTY OF ENGINEERING ul. Żeromskiego 116, Łódź

Contents

Purpose and range.....	3
Requirements	3
Functional requirements	3
Non-Functional requirements	4
Use case diagram.....	5
Component Diagram	5
Class Diagram	6
Use case implementation	7
Join chat.....	7
Read chat.....	8
Send message	11
Delete message	12
Leave chat.....	13
Set max players.....	14
Set minimum bet	15
Close Game session	17
See running Games.....	18
See connected Players.....	19
Edit Player	20
Ban Player	21
Refresh stats	23
Show game specific stats.....	23
Show today's stats.....	24
Show player stats.....	25
Show total stats	26
Show history	27
Activity diagrams	27
Delete Message	28
Send Message.....	28
Show Player statistics	29
State transition diagrams	29
AdminPanel Class	29
Chat Class.....	30
DataViewer Class	30
Timing Diagram.....	31

Package Diagram	33
Composite structure diagrams	33
Chat Class.....	33
DataViewer Class	34
Interaction Overview Diagram	35
Conclusions.....	36
User documentation.....	36
Chat.....	36
Functional description	36
User manual.....	36
Complete description	38
Installation description	38
Admin Panel.....	38
Functional description	38
User manual.....	39
Complete description	44
Installation description.....	44

Purpose and range

The Admin Panel component is designed to work as part of the Online Casino system, the purpose of which is the development of a web application that provides the basic functionalities of a Casino, such as playing slots, BlackJack and Roulette, withdraw and deposit money, chat with other Players and more.

The purpose of the Admin Panel component is to provide the administration and monitorization of Players and visualization of the relevant data and information present in the system, both as runtime data and database data. The component offers various administration tool such as banning Players, closing Games, limiting the number of connected Players, setting a minimum bet requirement and more, as described in the requirements section and use case diagram. At the same time, the list of logged Players and currently running games is also visualized. There also is a great deal of variety in the kind of data that can be visualized, that spans Player personal statistics to Game specific statistics to global total or daily statistics of the system as a whole. Statistics can also be visualized day by day.

Additionally, we decided to develop also a Chat, which does not belong to the Admin Panel component, but is in fact part of the User Management component. To simplify things, the Chat doesn't use the Database Management component and does all its work internally to its own package, which is fully developed by us.

The range of the component is limited by its strong dependencies on User Management and Game Management components. All operations of the Admin Panel need to use the interfaces provided by the aforementioned components, and thus by itself the component is merely visualizing received data or formatting input data to be used in the appropriate interface, and dealing with the communication between the front-end GUI and the back-end Server.

Requirements

Functional requirements

The server management component shall:

1. Let the admins ban a player until a certain time in the future.
2. Allow the admin to close game sessions
3. Allow the admin to change any attribute of any user
4. Allow the admin to change the minimum allowed bet
5. Allow the admin to set the maximum number of concurrently connected users
6. Show connected users and admin(s)
7. Show currently running game sessions
8. Show, for each game type:
 1. House losses and gains total
 2. Average and minimum bet
 3. Number of game sessions
 4. Profitability
9. Show today's :
 1. Aggregated house losses and gains

2. House losses and gains per game type
 3. Number of game sessions
 4. Profitability
10. Show total:
1. Aggregated house losses and gains
 2. Number of game sessions
 3. Profitability
12. Show player's :
1. Anagraphic data
 2. Total losses, gains and current balance
13. Show all daily stats (history) in the last month
14. Refresh the shown data periodically without any user input
15. Allow joining and leaving the chat
16. Read latest messages in the global chat
17. Let admins and users alike send messages in the global chat
18. Let users delete their own messages as long as they are shown in the chat
19. Let admins delete any message as long as they are shown in the chat

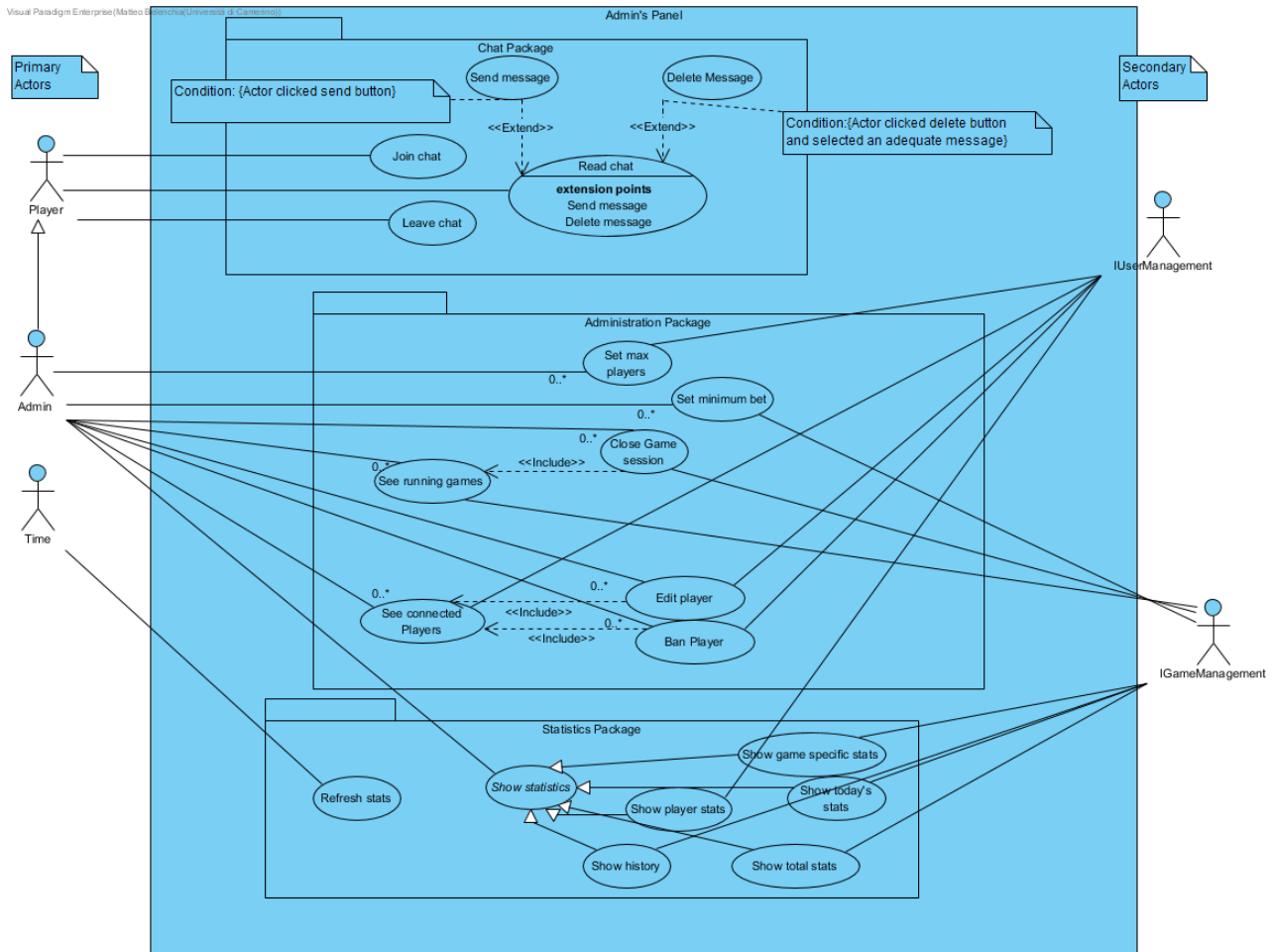
Non-Functional requirements

The server management component shall:

20. Let players be banned with a zero duration
21. Not allow a less than 1 minimum bet
22. Not allow to edit a Player into having an already used username
23. Not allow to edit a Player into having an already used email
24. Not allow a less than 1 maximum number of connected Players
25. Refresh shown statistics once every 5 minutes
26. Assign an ID to each chat message for identification and selection
27. Set the chat message timestamp as the server's local time upon receipt
28. Limit the number of messages shown on the chat
29. Highlight admin messages in the chat
30. Delegate the DB access for the editing of the users' data, kicking players, setting server capacity, query Player statistics and visualize connected Playersto the User Management Component by using the appropriate interface

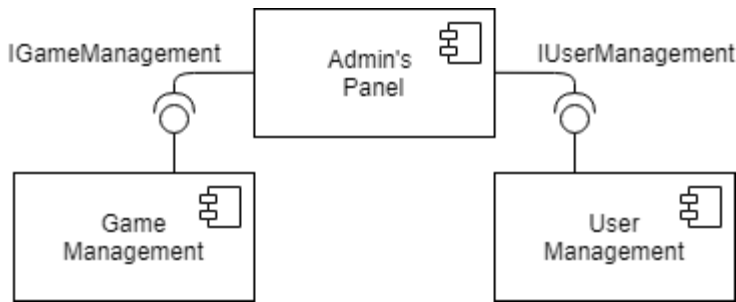
31. Delegate the DB access for closing games, setting the minimum bet and querying statistics to the Game Management Component by using the appropriate interface
32. Have a user-friendly graphical user interface

Use case diagram



In the Use-Case diagram actors on the left side are primary actors while those on the right are supporting actors. The actor Admin inherits from the more general Player actor as it is a more specialized role with additional use cases but which can still use all the features reserved for Players. The Time actor is used to trigger the Refresh stats use case. The supporting actors are associated with the use cases that depend on them for their functionality. The use cases are split in 3 packages. The package Chat contains the use cases relevant to the chat. Send message and Delete message are use cases which provide extended functionality to the Read chat use case, as in our analysis these functionalities only make sense after having read the chat. In the Administration package there are various use cases related to administration functionalities. We modeled Close Game session to include See running games because after closing a game we'd need to refresh the running games screen, and for the same reason Edit player and Ban player include the See connected players use case. The last package contains the statistics functionalities and features an abstract use case Show statistics which is a generalization of the other more specialized use cases.

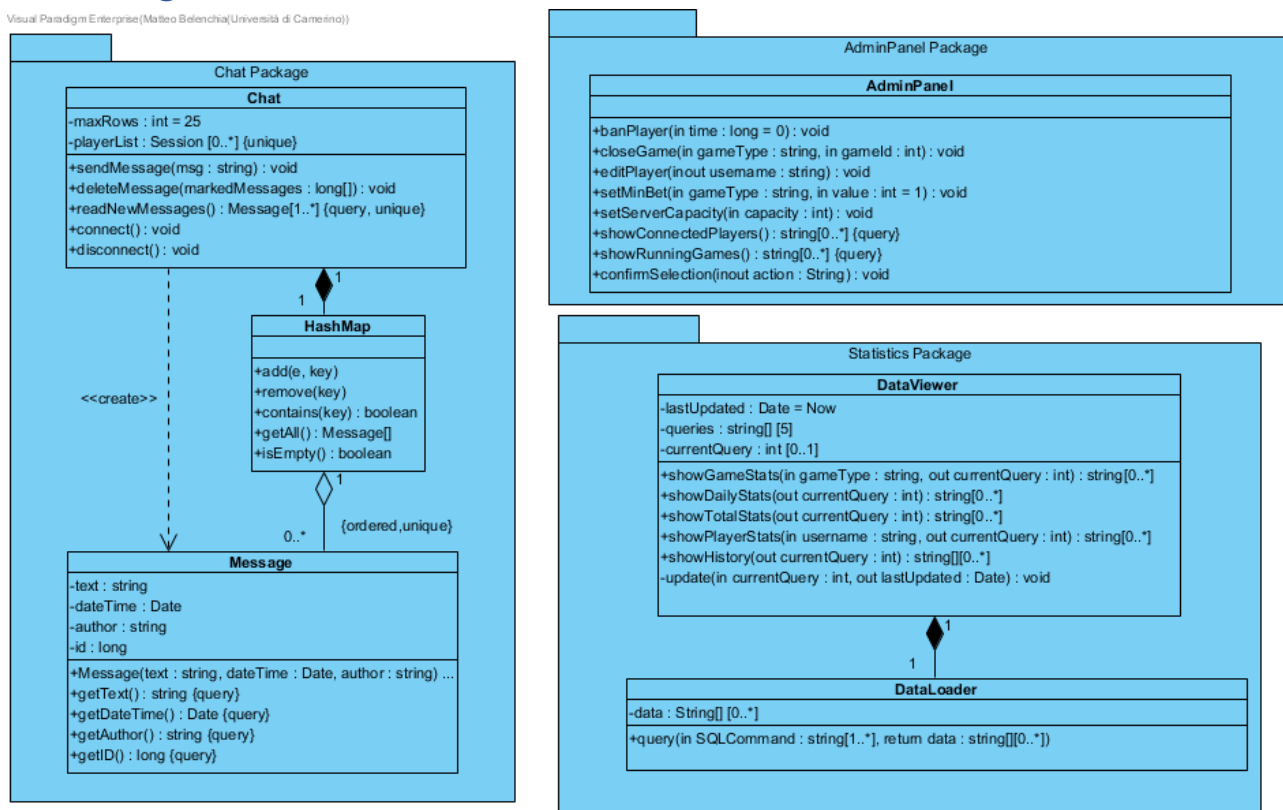
Component Diagram



This fragment of the component diagram here shows the Admin's Panel connections with the rest of the system (which is not shown fully). The component uses the IGameManagement interface to relate with the Game Management component and the IUserManagement interface to communicate with the User Management component. The component does not provide any interface.

Class Diagram

Visual Paradigm Enterprise (Matteo Belenchia/Università di Camerino)



The class diagram is split into 3 packages.

The Chat package contains the Chat class which is the main class for the Chat functionality. The Chat class provides methods to connect, disconnect, send, delete and read messages. Its attributes specify the maximum number of lines the chat has and a list of Sessions to hold information about the Players logged to the chat. The class contains exactly one HashMap object to hold the messages; the chat does not use a database connection. In the HashMap are stored Message objects which are maintained in a certain order and are at all times unique thanks to the id attribute. The Message class contains getters and a constructor method. The Message objects are created by the Chat class but are aggregated into the HashMap.

The AdminPanel Package contains only one class "AdminPanel" which contains a series of methods. Notice that the confirmSelection() method is used to confirm the changes made with the closeGame() and editPlayer() methods because they can be used to perform multiple changes.

The Statistics Package main class is DataViewer which contains the methods for the various statistics type that can be shown. These methods call update() which in turn calls the query() method of the DataLoader class, the single instance of which is contained in the DataViewer class. The various SQL queries are stored in the attribute “queries” and the one currently selected index is saved in the attribute “currentQuery”, which is used as argument of the update() method. The “lastUpdated” attribute is used to determine if the 5 minute timeout for refreshing data has elapsed. Some methods also require the type of game or the username of a player to run.

Use case implementation

Join chat

1. Join chat

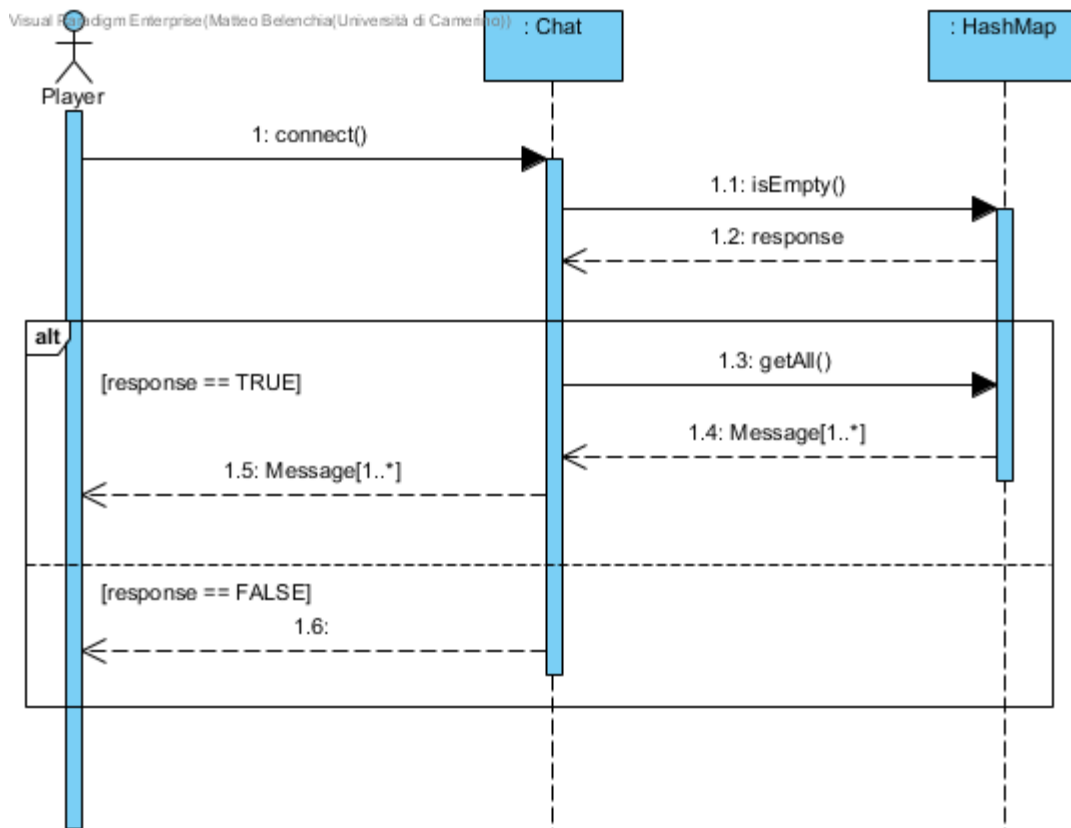
ID: UC14

Primary Actors	♀ Player
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	Player is not connected to the chat
Post-conditions	Player is connected to the chat
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when the Player opens the chat Web page
2. **SYSTEM** logs the Player in the list of connected players
3. **if** there are messages in the Chat::HashMap
 - 3.1. **for each** Message in Chat::HasMap
 - 3.1.1. **SYSTEM** sends the Message to the Player
 - end for each**
- end if**



Read chat


1. Read chat

ID: UC01

Primary Actors	♂ Player
Level	N/A
Complexity	N/A
Use Case Status	Complete
Implementation Status	Scheduled
Preconditions	Join chat
Post-conditions	None
Author	Marcin Massalski
Assumptions	N/A

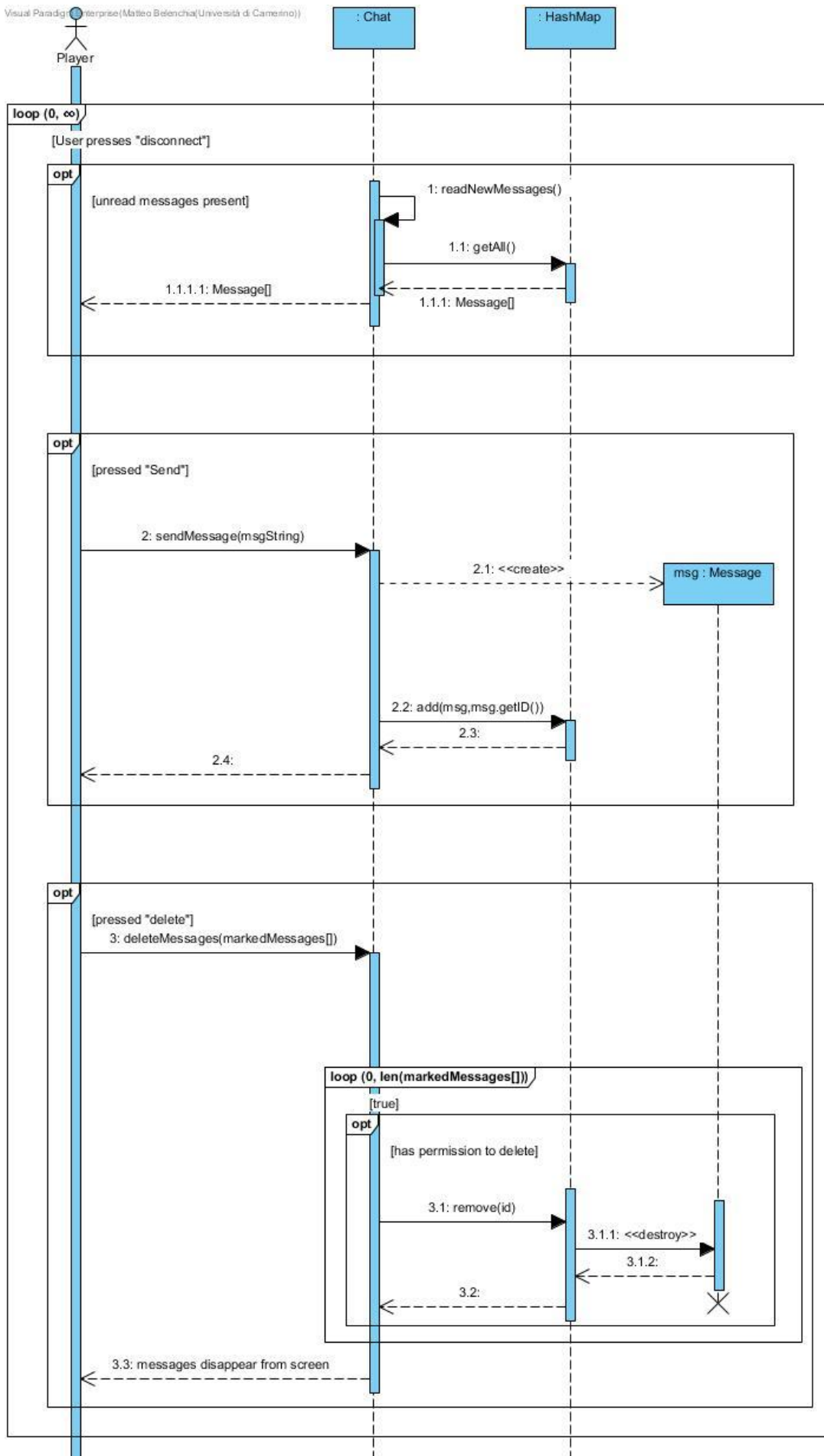
1.1. Scenarios

1.1.1. Scenario

1. The use case starts when the Player has completed  Join chat
2. while The Player hasn't pressed "Disconnect"
 - 2.1. if There are unread Messages in the Chat
 - 2.1.1. SYSTEM gets the new Messages from Chat::HashMap
 - 2.1.2. SYSTEM sends the new Messages to Player
 - end if
 - 2.2. The Player reads the messages
- end while

Extension:

- 2.2.a. The Player types a Message
 1. The Player presses "Send"
 2. SYSTEM creates a Message object
 3. SYSTEM adds the Message to the Chat::HashMap
- 2.2.b. The Player selects a number of Messages
 1. The Player presses "Delete"
 2. if The player has the permission to delete any of the selected Messages
 - 2.1. SYSTEM deletes those Messages from the Chat::HashMap
 - end if



Send message

1. Send message

ID: UC02

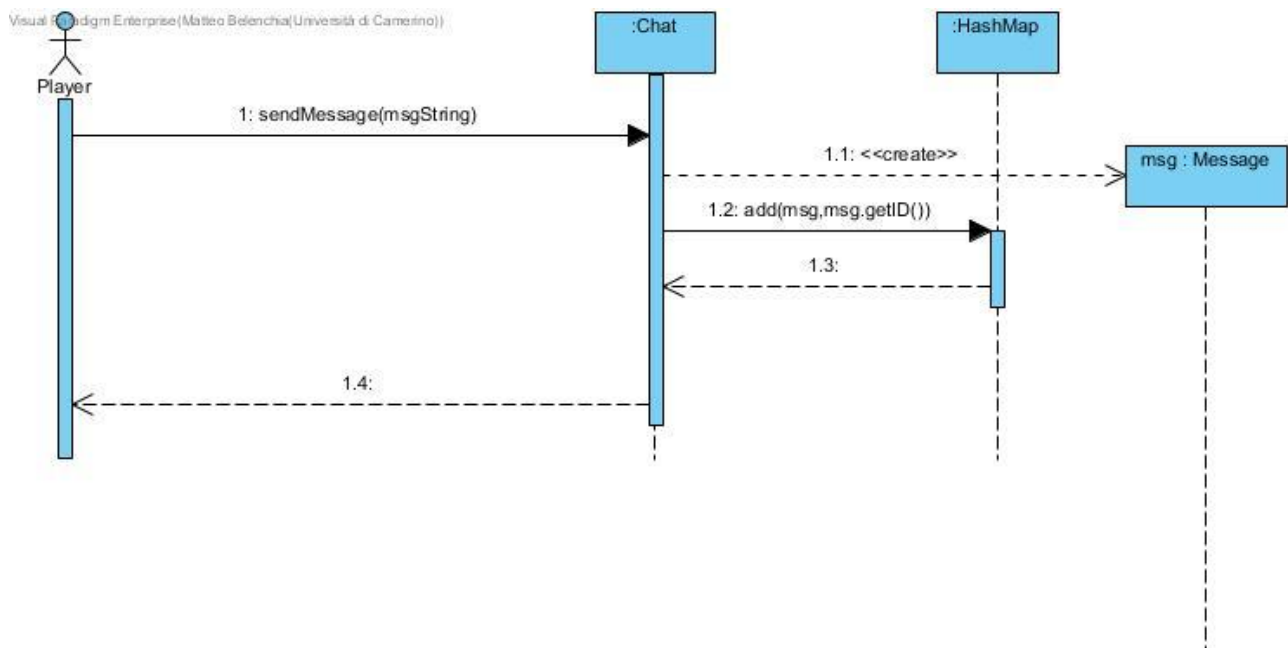
Primary Actors	♂ Player
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	Join chat
Post-conditions	The Message is added to the Chat
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Player types a Message
 - 1.1. The Player presses "Send"
 - 1.2. **SYSTEM** creates a Message object
 - 1.3. **SYSTEM** adds the Message to the Chat::HashMap

Visual Modeling Enterprise (Matteo Belenchia/Università di Camerino)



Delete message

1. Delete Message

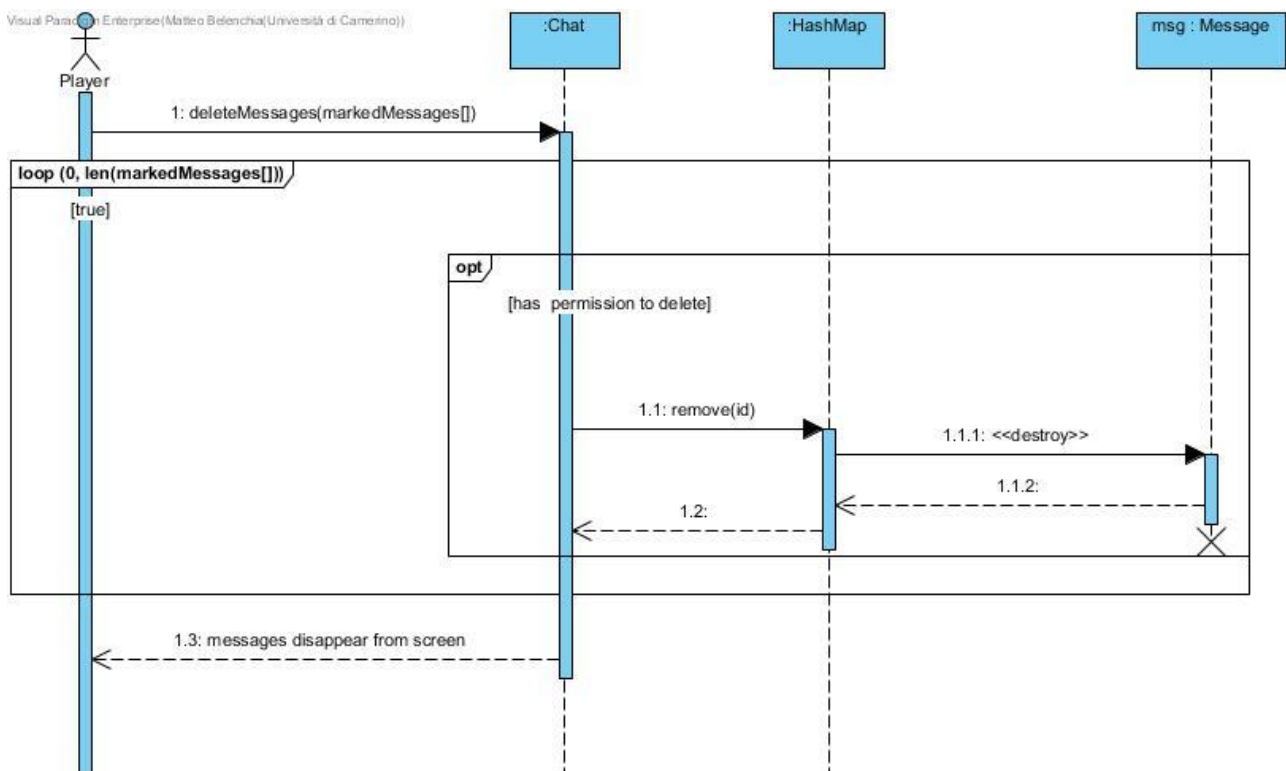
ID: UC03

Primary Actors	♂ Player
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	Join chat There are Messages in the Chat Player has permission to delete any of the selected Messages
Post-conditions	The Messages are remove from the Chat
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Player selects a number of Messages
 - 1.1. The Player presses "Delete"
 - 1.2. **if** The player has the permission to delete any of the selected Messages
 - 1.2.1. **SYSTEM** deletes those Messages from the Chat::HashMap
 - end if**



Leave chat

1. Leave chat

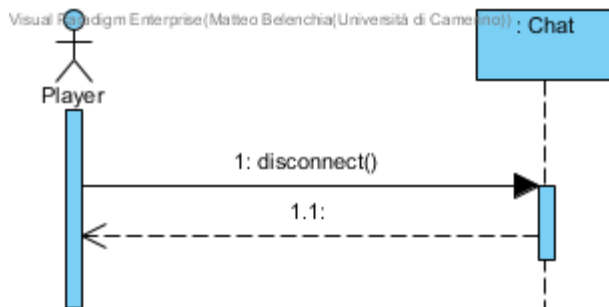
ID: UC13

Primary Actors	♂ Player
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	Player is connected to the Chat
Post-conditions	Player is not connected to the Chat
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when the Player closes the Chat Web page
2. **SYSTEM** removes the Player from the list of Players logged to the Chat



Set max players

1. Set max Players

ID: UC22

Primary Actors	Admin
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	None
Post-conditions	Players cannot log in until the number of connected Players falls below the max users value
Author	N/A
Assumptions	N/A

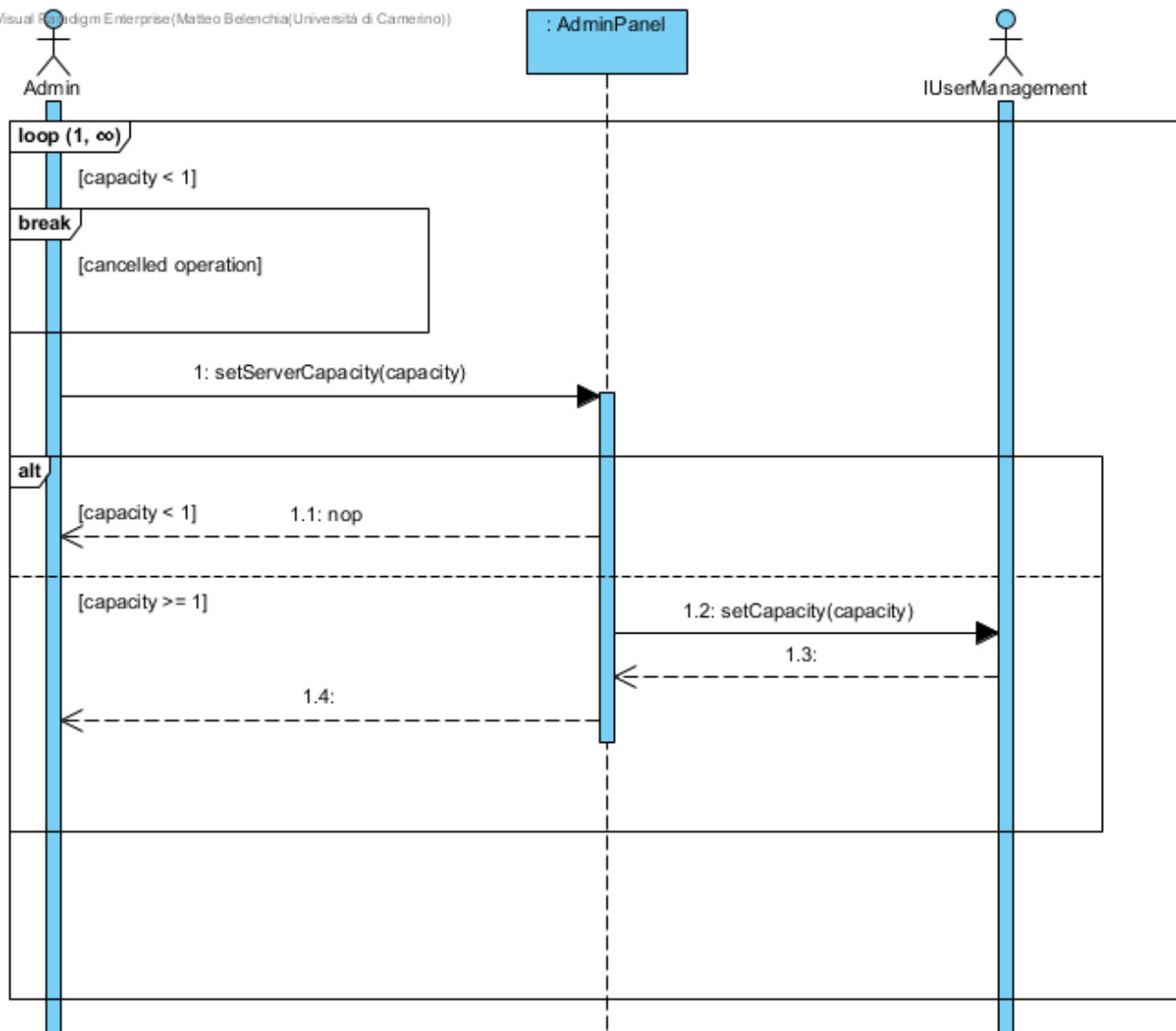
1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Admin presses "Set server capacity" button
2. Admin enters the server capacity value
3. if the value is less than 1
 - 3.1. **SYSTEM** resets the value to 1end if
4. Admin confirms the operation
5. **SYSTEM** reads the server capacity value
6. **SYSTEM** sends the value to IUserManagement

1.1.2. Scenario2

1. The alternative flow begins at any time before step 4 of the main flow
2. Admin cancels the operation



Set minimum bet

1. Set minimum bet

ID: UC17

Primary Actors	Admin
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	N/A
Post-conditions	A new value for minimum bet for the selected game type has been set Players cannot bet a value less than the new value set for that game type
Author	N/A
Assumptions	N/A

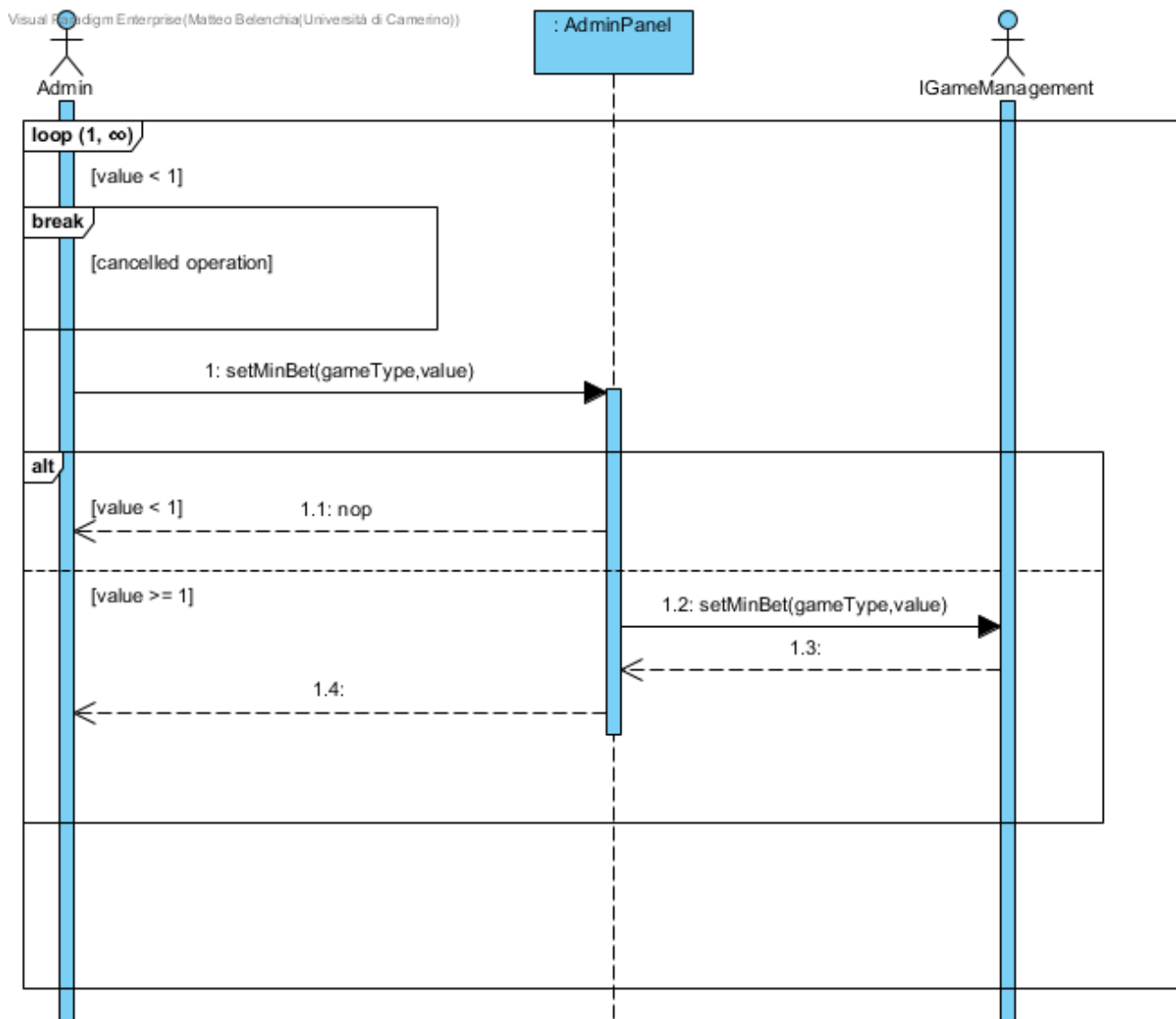
1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Admin presses "Set Minimum Bet" button
2. Admin selects a Game Type from the list of Game Types
3. Admin inserts a value
4. **if** the value is less than 1
 - 4.1. **SYSTEM** resets the value to 1**end if**
5. Admin confirms the operation
6. **SYSTEM** reads the Game Type selected
7. **SYSTEM** reads the value to set as Minimum Bet
8. **SYSTEM** calls the IGameManagement to enforce the Minimum Bet

1.1.2. Scenario 2

1. The alternative flow begins at any time before step 5 of the main flow
2. Admin cancels the operation



Close Game session


1. Close Game session

ID: UC05

Primary Actors	♀ Admin
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are games currently running on the system
Post-conditions	The selected games are bound to be closed by the system
Author	N/A
Assumptions	N/A

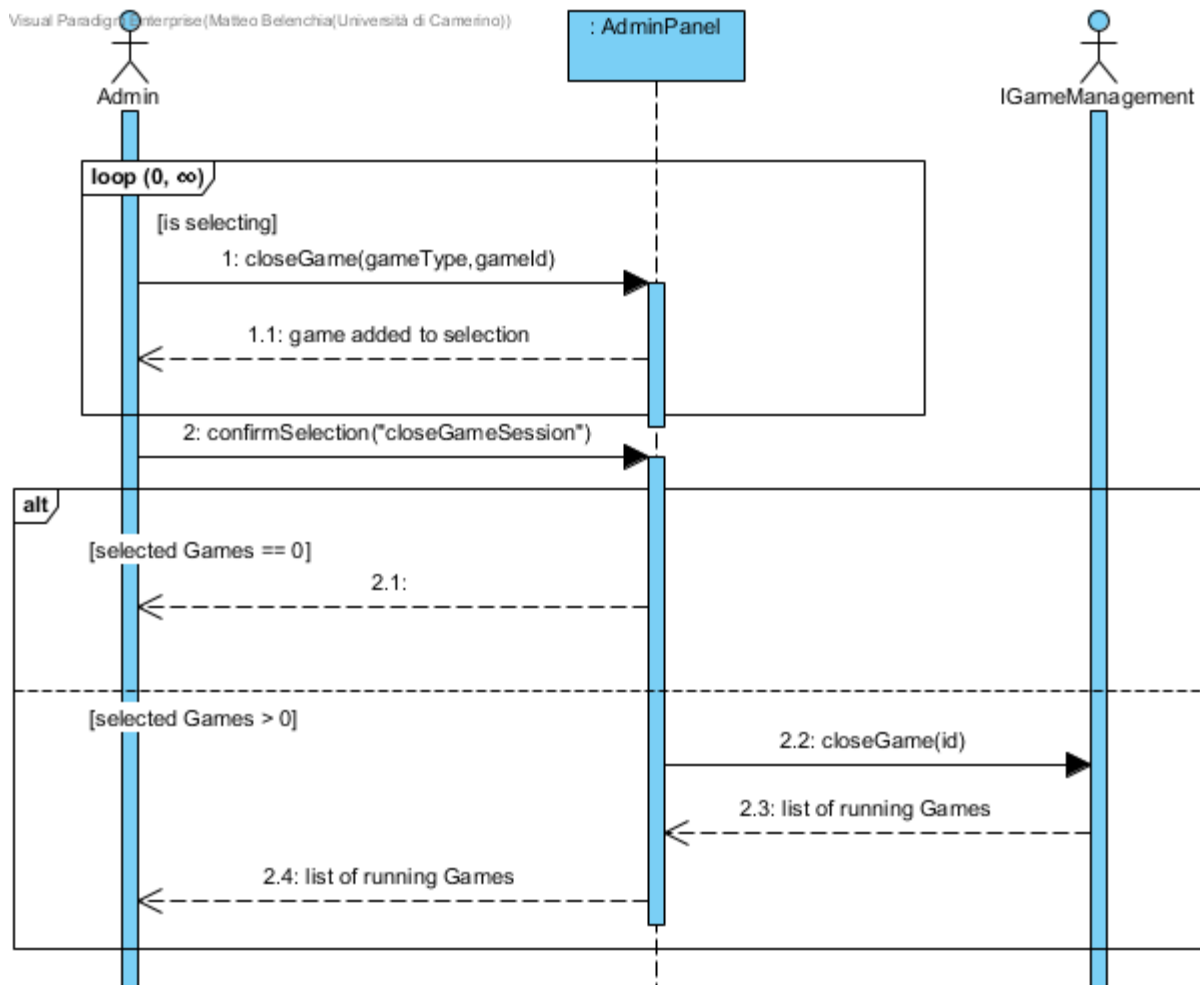
1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Admin selects a running Game
2. **SYSTEM** adds Game to selection
3. **while** Admin is selecting Games
 - 3.1. **if** Admin deselects a Game
 - 3.1.1. **SYSTEM** removes Game from selection
 - 3.2. **else if** Admin selects a Game
 - 3.2.1. **SYSTEM** adds Game to selection**end if**
- end while**
4. Admin confirms the operation
5. **SYSTEM** asks IGameManagement to close the Games in selection
6. **SYSTEM**  [See running games](#)

1.1.2. Scenario2

1. The alternative flow begins after step 3.1.1 of the main flow
 2. **if** there are no games selected
 - 2.1. the operation is canceled
- end if**



See running Games

1. See running games

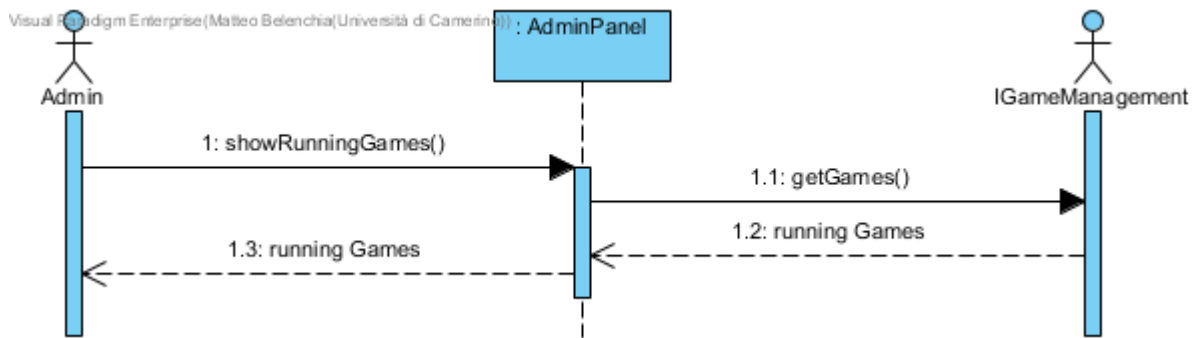
ID: UC19

Primary Actors	♂ Admin, ♂ Time
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are running Games in the system
Post-conditions	None
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Admin opens the Admin Panel Web page or there is a change in running games
2. **SYSTEM** queries IGameManagement for the list of running Games
3. **SYSTEM** visualizes the list of running Games



See connected Players

1. See connected Players

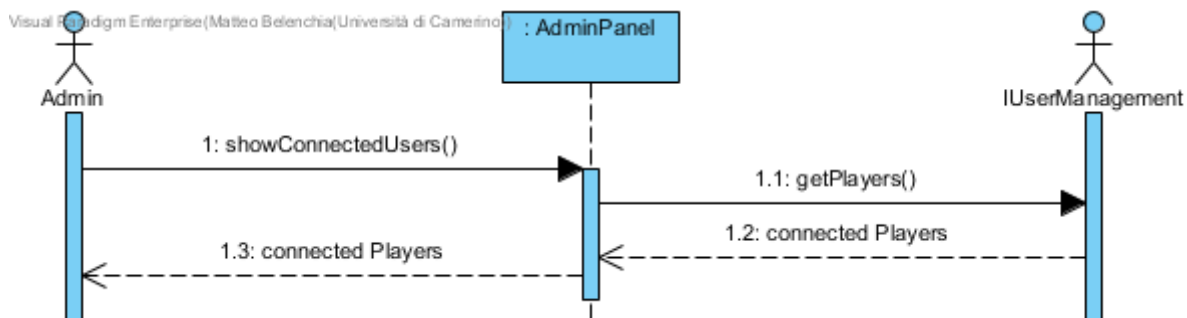
ID: UC18

Primary Actors	Admin, Time
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are connected Players on the system
Post-conditions	None
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The use case starts when Admin enters the Admin Panel Web page or if there is a change in connected Players
2. **SYSTEM** queries IUserManagement for the list of connected Players
3. **SYSTEM** visualizes the list of connected Players



Edit Player


1. Edit Player

ID: UC15

Primary Actors	♀ Admin
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are Players registered to the system
Post-conditions	The selected Player's attributes are updated to the new values
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

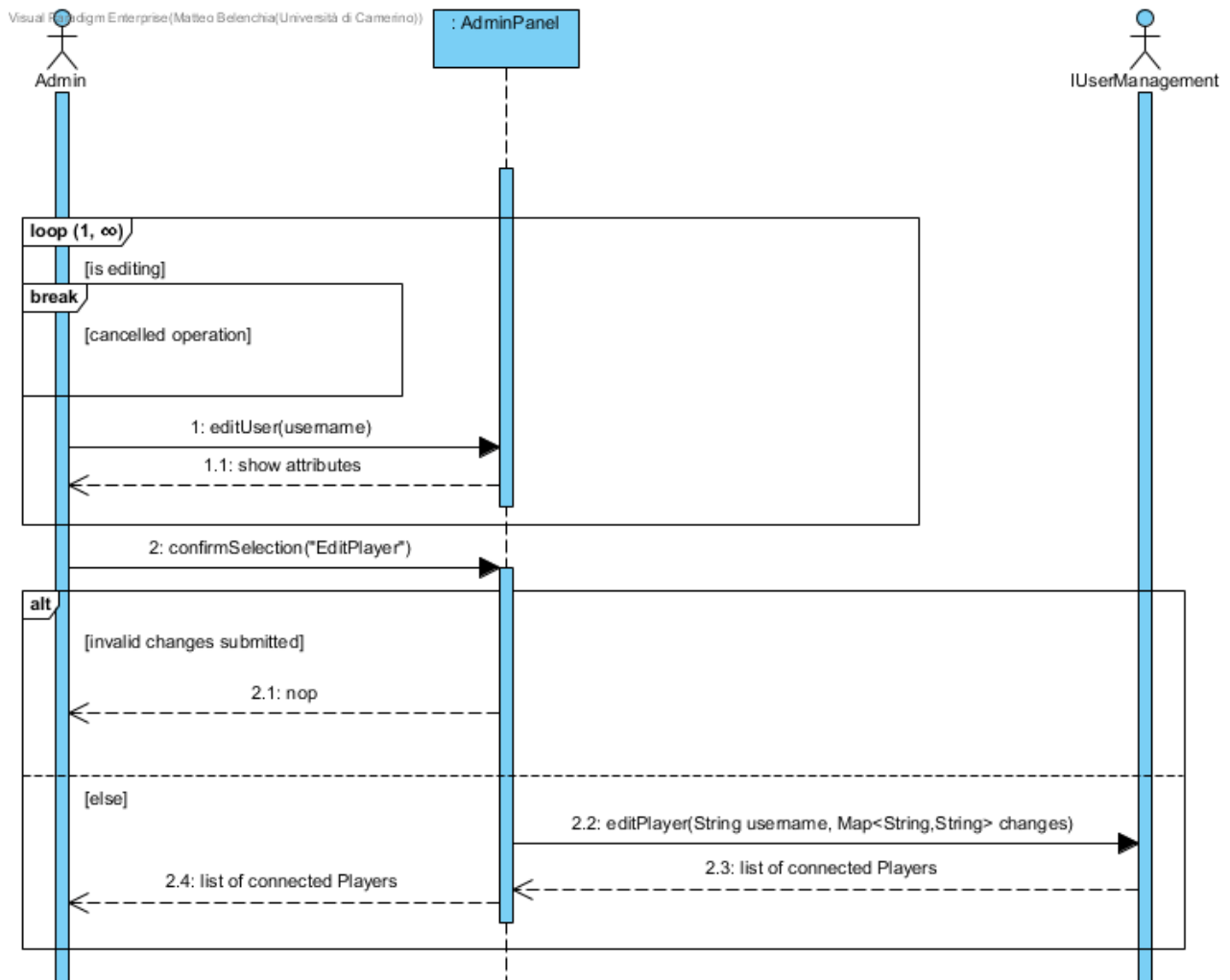
1. The use case starts when Admin presses "Edit Player" button
2. Admin selects a Player from a list of all Players
3. while Admin isn't done editing
 - 3.1. Admin changes a Player attributeend while
4. Admin confirms the operation
5. SYSTEM reads the Player selected to be edited
6. SYSTEM reads the set of attributes set by Admin
7. SYSTEM calls the IUserManagement interface to carry out the changes
8. SYSTEM  [See connected Players](#)

1.1.2. Scenario2

1. The alternative flow begins at any time before step 4 of the main flow
2. Admin cancels the operation

1.1.3. Scenario3

1. The alternative flow begins after step 6 of the main flow
2. SYSTEM finds unallowed changes in the edits proposed by Admin
3. SYSTEM cancels the operation



Ban Player


1. Ban Player

ID: UC04

Primary Actors	Admin
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are Players registered to the system
Post-conditions	The selected Player is logged off The selected Player cannot log in until the time is elapsed
Author	N/A
Assumptions	N/A

1.1. Scenarios

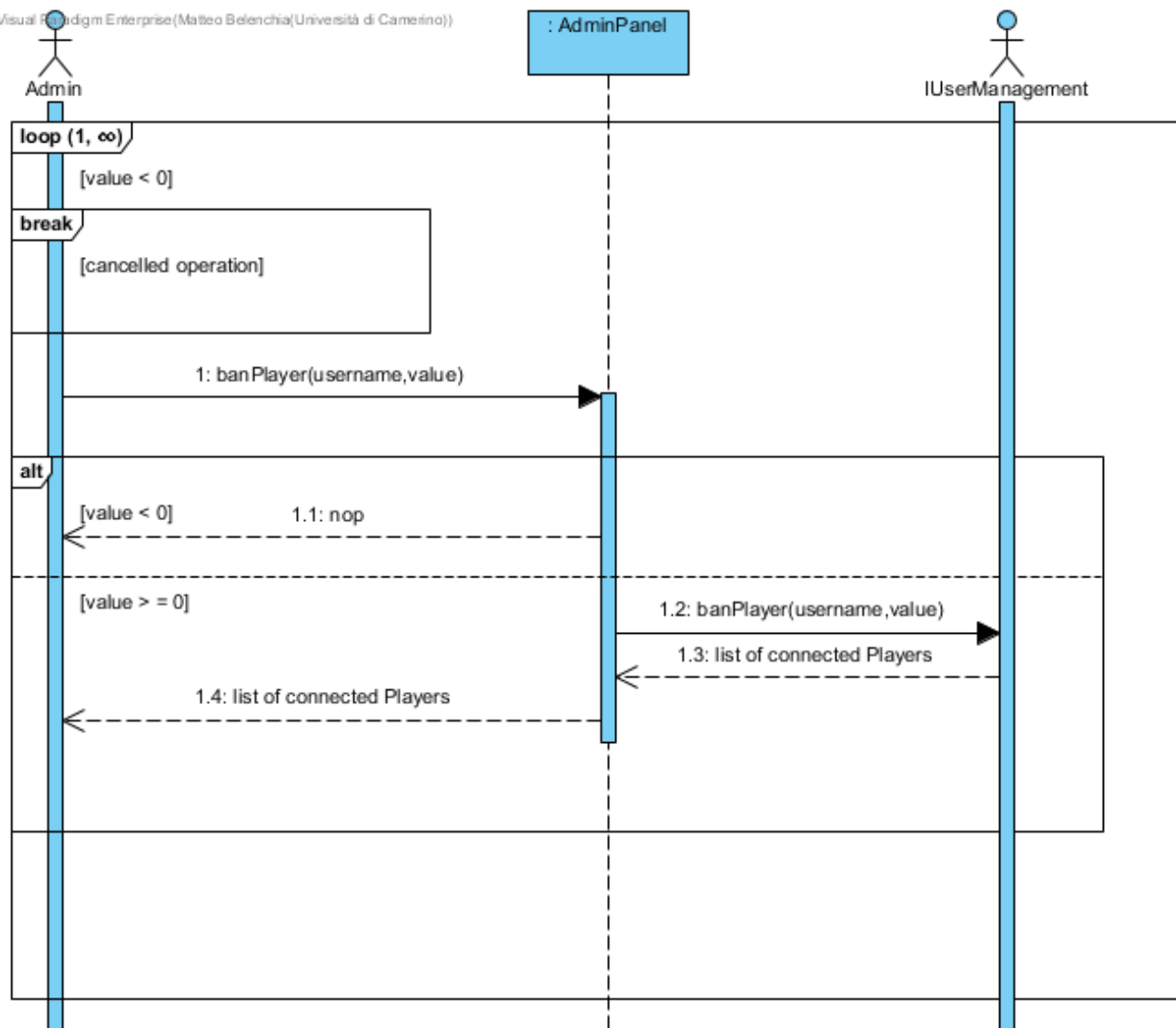
1.1.1. Scenario

1. The use case starts when Admin presses the "Ban Player" button
2. Admin selects a Player from a list of all Players
3. Admin inserts a value in seconds
4. **if** the value is negative
 - 4.1. **SYSTEM** resets the value to 0**end if**
5. Admin confirms the operation
6. **SYSTEM** reads the Player selected to be banned
7. **SYSTEM** reads the value in seconds used to ban the Player
8. **SYSTEM** calls the IUserManagement interface to carry out the ban
9. **SYSTEM**  [See connected Players](#)

1.1.2. Scenario2

1. The alternative flow begins at any time before step 5 of the main flow
2. Admin cancels the operation

Visual Paradigm Enterprise (Matteo Belfenchia (Università di Camerino))



Refresh stats

1. Refresh stats

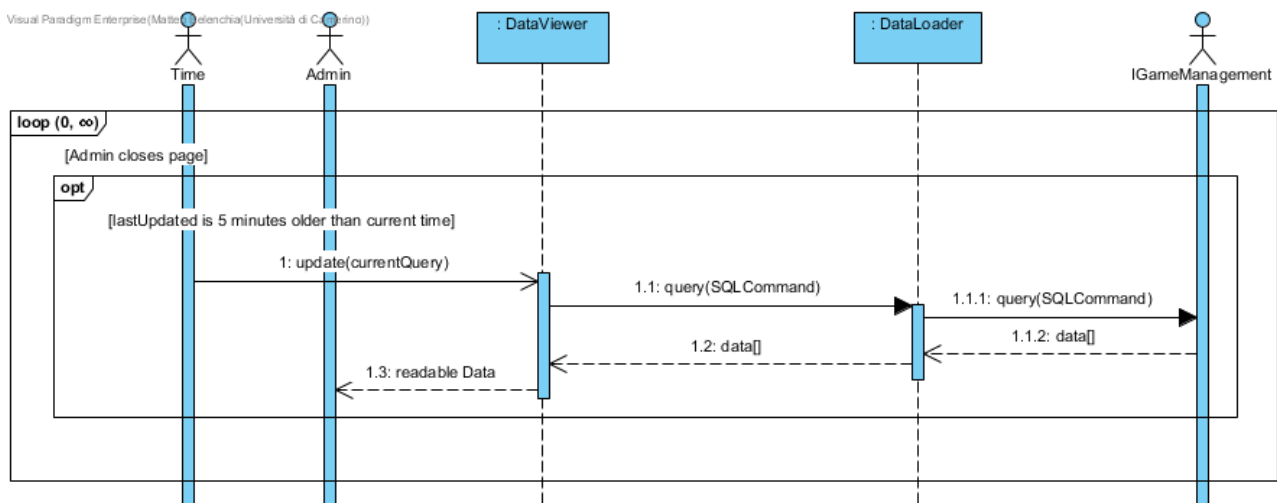
ID: UC06

Primary Actors	Time
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are statistics to be shown The last query submitted is known
Post-conditions	Game statistics are visualized on screen
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The Use Case starts if more than 5 minutes have passed since the last query
2. **SYSTEM** queries the Game Management
3. **SYSTEM** elaborates the data received
4. **SYSTEM** offers a view of the data to the Admin



Show game specific stats

1. Show game specific stats

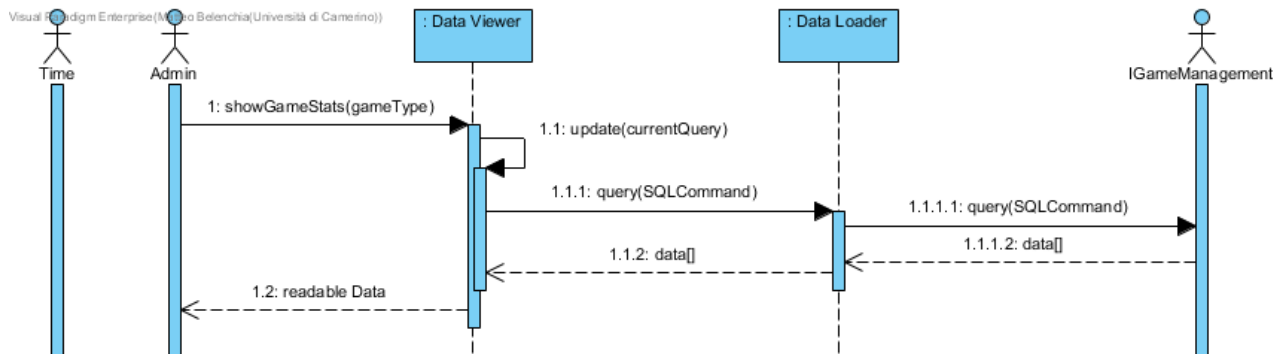
ID: UC09

Primary Actors	Admin
Supporting Actors	IGameManagement
Level	N/A
Complexity	N/A
Use Case Status	Complete
Implementation Status	Scheduled
Preconditions	There are statistics to be shown
Post-conditions	Game statistics are visualized on screen
Author	Matteo Belenchia
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario 1

1. The Use Case starts when the Admin clicks the "Show game statistics" button
2. Admin selects the game type
3. **SYSTEM** queries the Game Management
4. **SYSTEM** elaborates the data received
5. **SYSTEM** offers a view of the data to the Admin



Show today's stats

1. Show today's stats

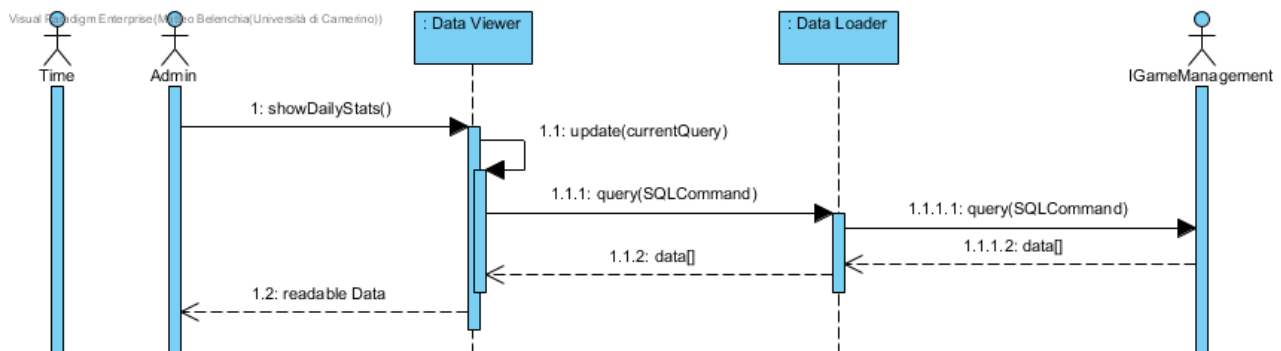
ID: UC07

Primary Actors	Admin
Supporting Actors	IGameManagement
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are statistics to be shown
Post-conditions	Daily statistics are visualized on screen
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The Use Case starts when the Admin clicks the "Show daily statistics" button
2. **SYSTEM** queries the Game Management
3. **SYSTEM** elaborates the data received
4. **SYSTEM** offers a view of the data to the Admin



Show player stats

1. Show player stats

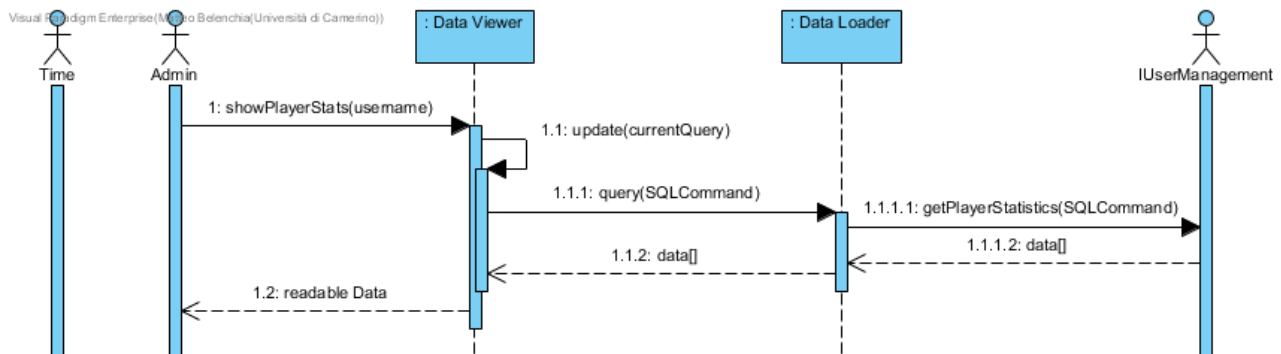
ID: UC20

Primary Actors	Admin
Supporting Actors	IUserManagement
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are statistics to be shown There are Players in the system
Post-conditions	Player statistics are visualized on screen
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The Use Case starts when the Admin clicks the "Show player statistics" button
2. Admin selects a Player
3. **SYSTEM** queries the Game Management
4. **SYSTEM** elaborates the data received
5. **SYSTEM** offers a view of the data to the Admin



Show total stats

1. Show total stats

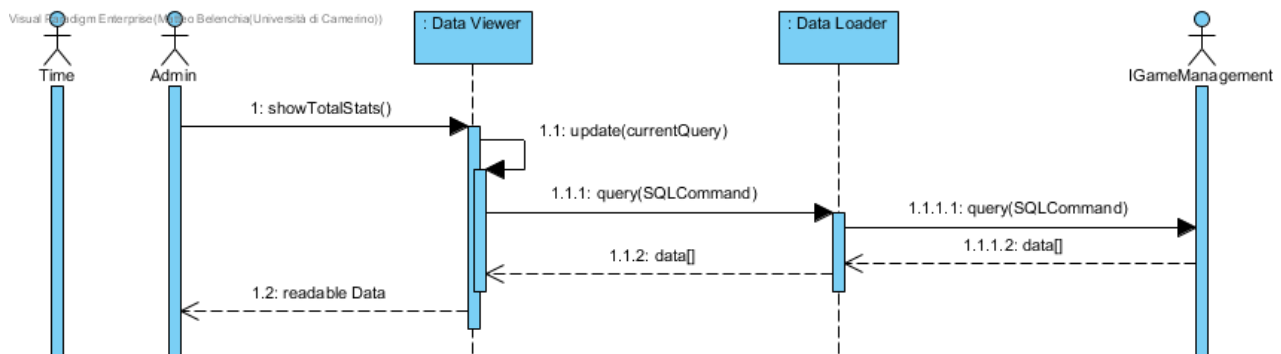
ID: UC08

Primary Actors	Admin
Supporting Actors	IGameManagement
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are statistics to be shown
Post-conditions	Total statistics are visualized on screen
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

1. The Use Case starts when the Admin clicks the "Show total statistics" button
2. **SYSTEM** queries the Game Management
3. **SYSTEM** elaborates the data received
4. **SYSTEM** offers a view of the data to the Admin



Show history

1. Show history

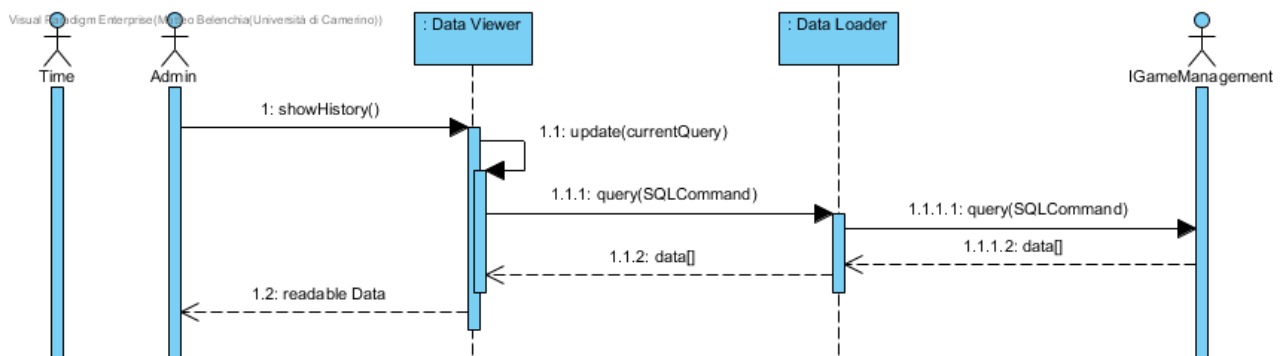
ID: UC21

Primary Actors	♀ Admin
Supporting Actors	♀ IGameManagement
Level	N/A
Complexity	N/A
Use Case Status	N/A
Implementation Status	N/A
Preconditions	There are statistics to be shown
Post-conditions	A history of daily statistics are visualized on screen
Author	N/A
Assumptions	N/A

1.1. Scenarios

1.1.1. Scenario

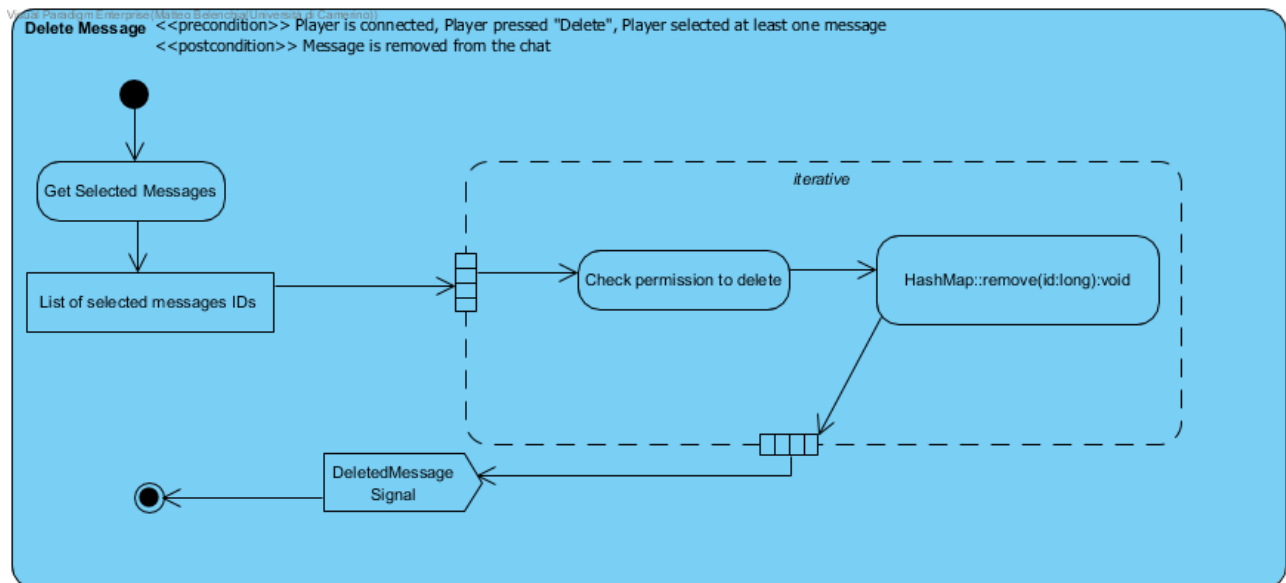
1. The Use Case starts when the Admin clicks the "Show history" button
2. **SYSTEM** queries the Game Management
3. **SYSTEM** elaborates the data received
4. **SYSTEM** offers a view of the data to the Admin



Activity diagrams

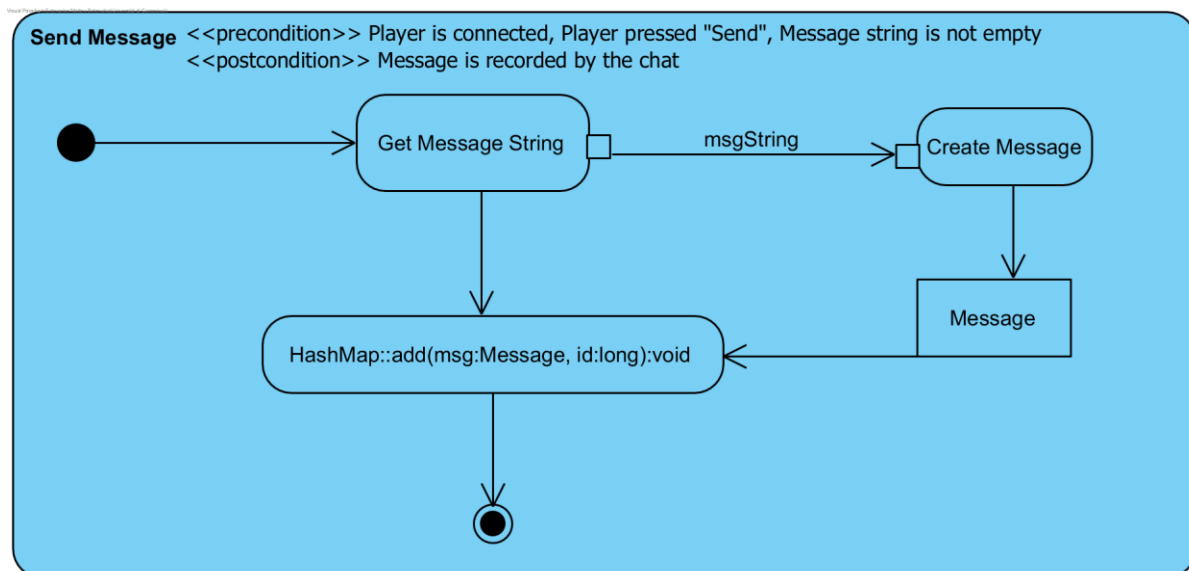
The following activity diagrams show the execution flow of the use cases "Delete Message", "Send Message" and "Show Player statistics".

Delete Message



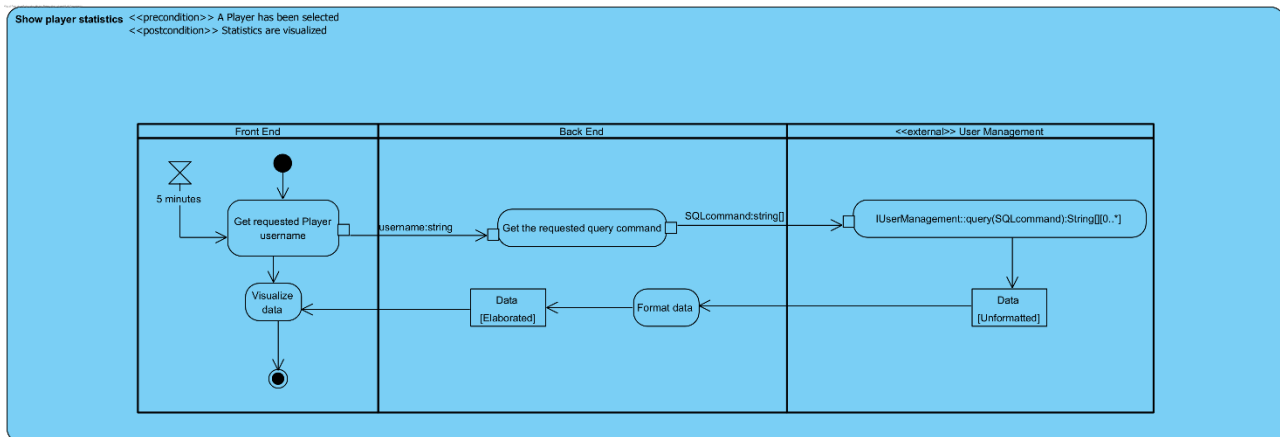
The preconditions before the activity can start is that the Player is connected to the chat, he has pressed the Delete button and selected at least one message for deletion. Once the list of selected messages is retrieved, we pass the list of the Message IDs to iteratively check for permission to delete and if permission is granted the message is removed from the HashMap. Then a signal is launched to mark that the chat has been modified and it signifies that the current chat has to be sent to all connected Players. The post conditions is that the Messages for which there was permission to delete are removed from the chat.

Send Message



The preconditions is that the Player is connected to the Chat, it has pressed send and the Message he or she composed is not empty. The Message string is read and used to create a Message object which is then passed to the HashMap to be added to it. The postcondition is that the Message is saved to the chat.

Show Player statistics



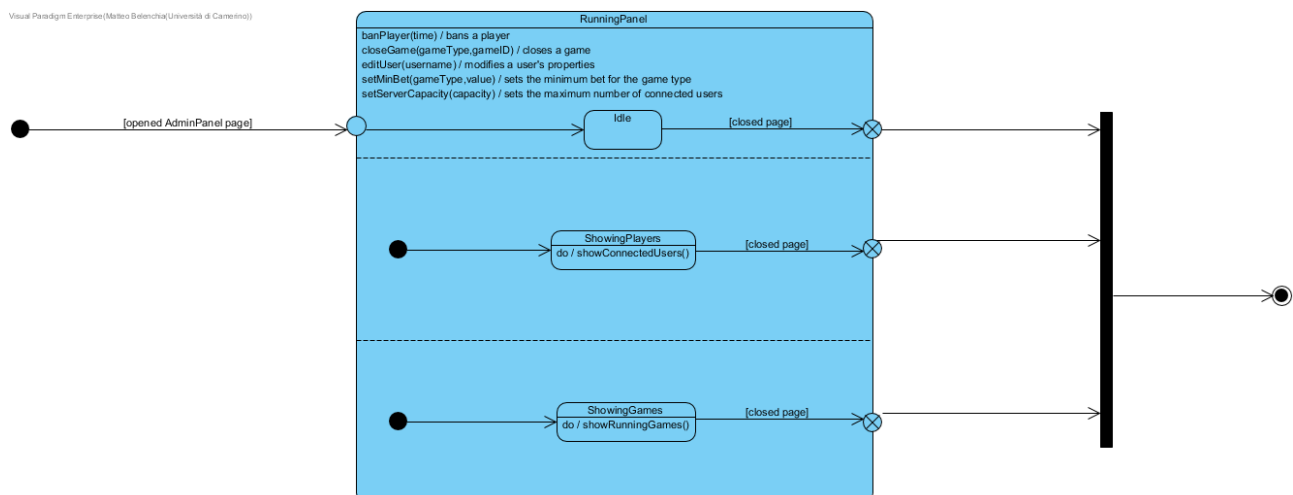
The precondition is that there is a Player currently selected. The Front End application gets the Player username and passes it to the Back End where the username is added to the appropriate query string which is then passed as argument to the external User Management component. The component then returns some raw Data which is processed in the Back End and is visualized in the Front End. Additionally, every 5 minutes the process is repeated again. The postcondition is that statistics are currently visualized in the Front End application.

State transition diagrams

The state transition diagrams are relative to the most important classes of the component, Chat, AdminPanel and DataView.

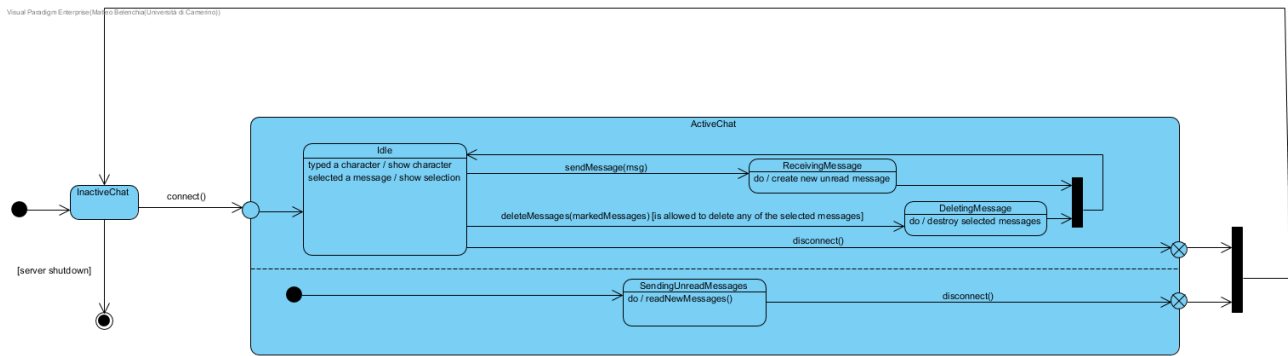
AdminPanel Class

Visual Paradigm Enterprise (Matteo Boleghetti/Università di Camerino)



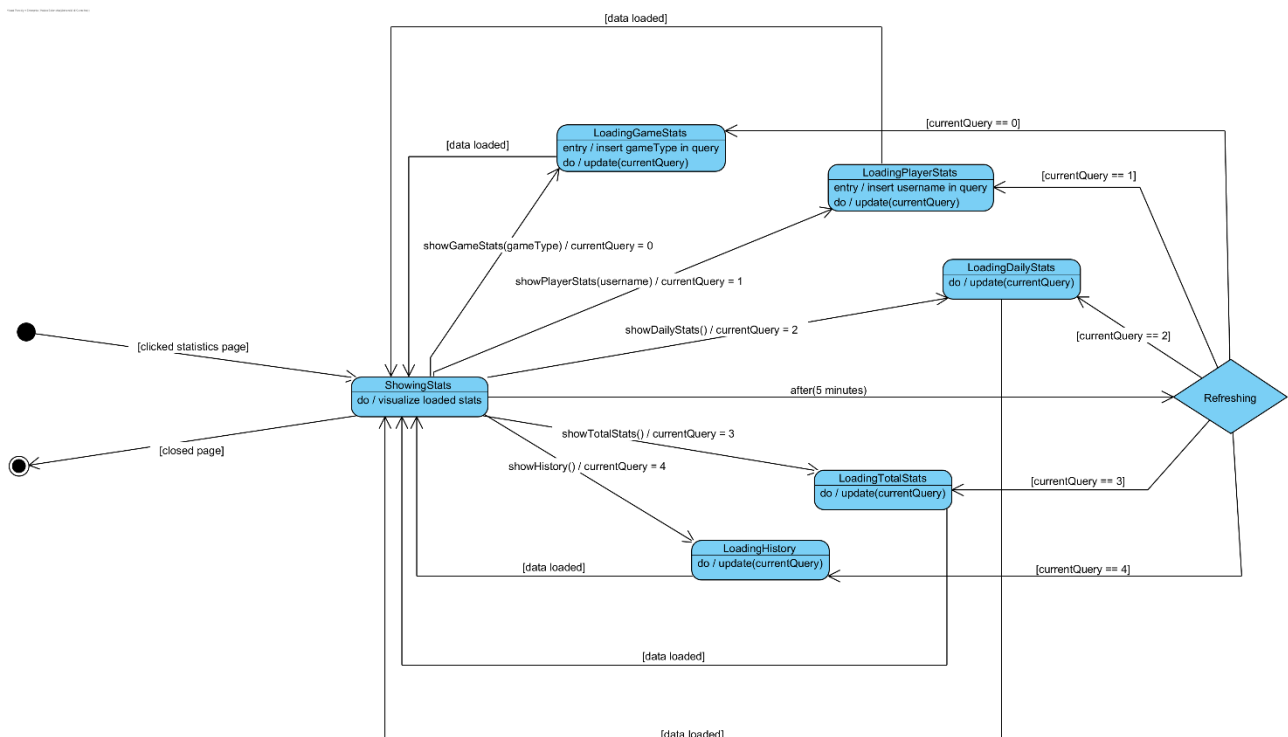
The AdminPanel class is modeled with a single composite state RunningPanel which is entered when the webpage is opened. This state is composed by 3 parallel running substates. The Idle substate inherits all the actions defined in the RunningPanel state, which correspond to the various actions defined in the AdminPanel class, and these actions do not provoke any state transition but can be repeated as often as the Admin wants. Concurrently the substates ShowingPlayers and ShowingGames run the methods showConnectedUsers() and showRunningGames(). On page close all the substates terminate and the state transitions out from RunningPanel.

Chat Class



The Chat class starts in the InactiveChat state which will transition to ActiveChat when the method `connect()` is called. The ActiveChat state is composed by two section that run in parallel. In the top section the control goes to the Idle state which contains actions related to typing a message and selecting messages. Then when calling the `sendMessage()` method the state transitions in the ReceivingMessage substate in which the chat is creating the Message object and adding it to the HashMap, while instead calling `deleteMessages()` when the caller has permission to delete at least one of the selected messages fires the transition to DeletingMessage substate in which the Messages are removed from the HashMap. Both DeletingMessage and ReceivingMessage transition back into the Idle substate. In parallel in the bottom section runs the SendingUnreadMessages substate which runs the `readNewMessages()` method. When calling the `disconnect` method both sections exist the composite state and the current state becomes once again the InactiveChat state.

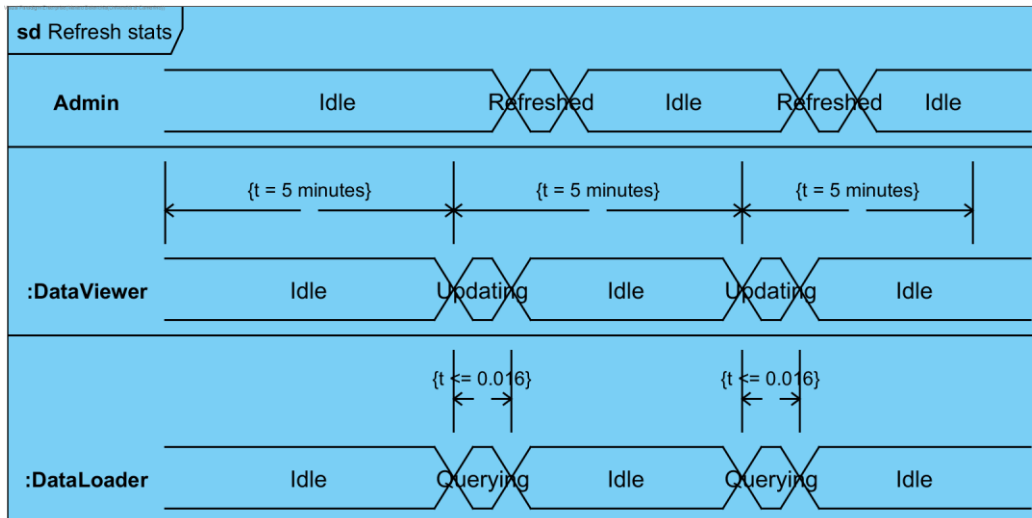
DataViewer Class



The initial state is reached upon opening the statistics webpage. In the ShowingStats state the statistics currently loaded are displayed if any. From such state, calling any of the following methods trigger a transition to the corresponding state : `showGameStats()` to LoadingGameStats, `showPlayerStats` to LoadingPlayerStats and so on. In each transition the variable `currentQuery` is set to a particular number which represents the corresponding index in the array of SQL queries. This value is then used in each of these states as argument to the `update()` method. When the data has finished loading the state transitions

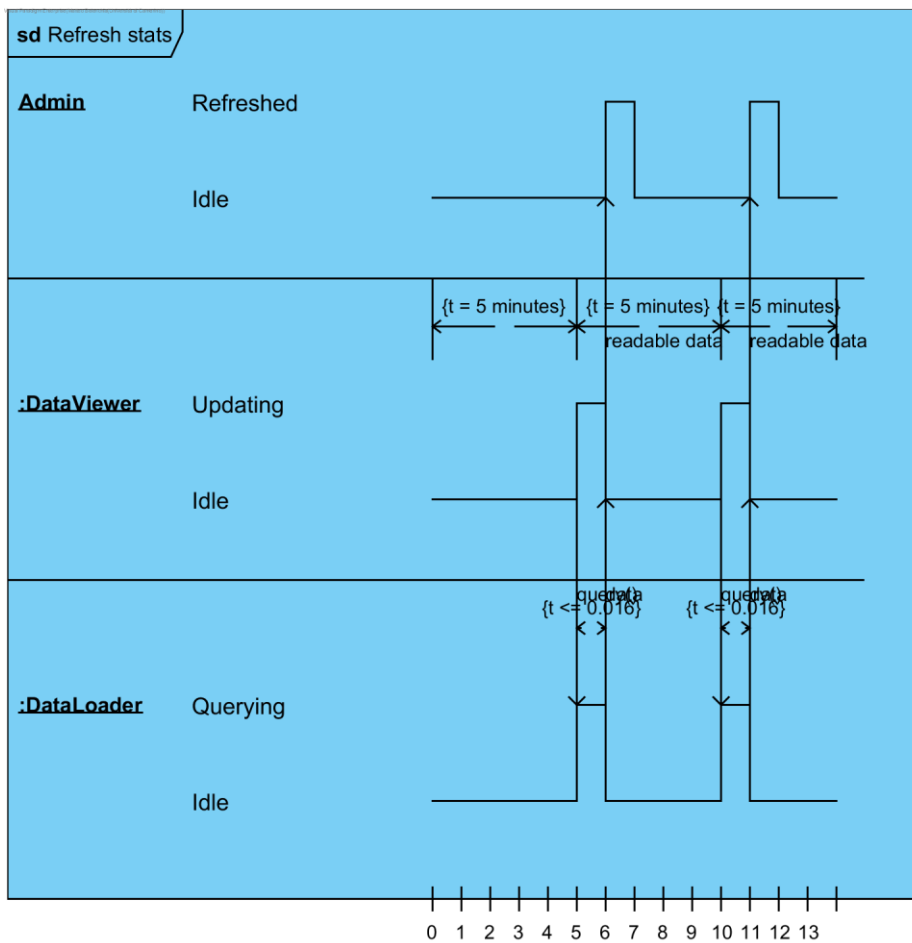
back to ShowingStats. From such state, when 5 minutes have passed since the last call to update(), the state transitions into the Refreshing choice, where according to the value of currentQuery the state will change into any of the Loading states previously mentioned. From ShowingStats then the State diagram terminates when the webpage is closed.

Timing Diagram



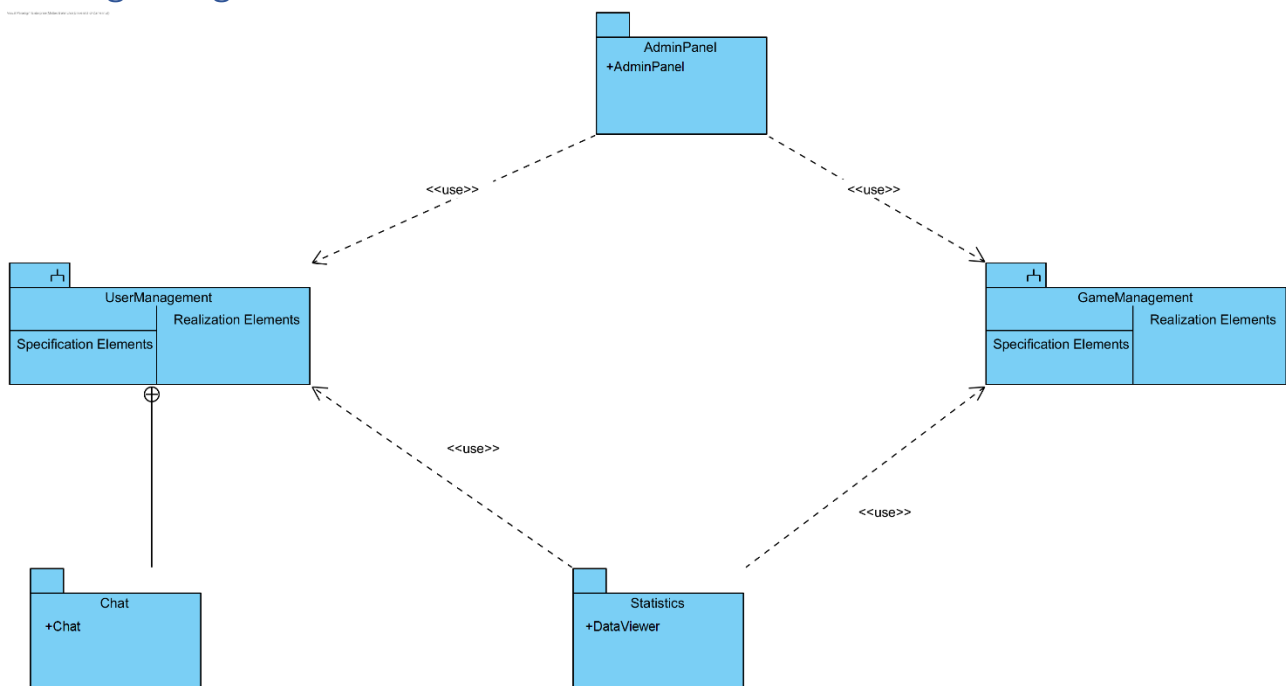
The timing diagram shows the timings relative to the use case Refresh stats. This timing diagram is in compact view because it's easier to visualize. Looking from the central lane every 5 minutes the state of DataViewer changes from Idle to Updating, which triggers the change of state in DataLoader from Idle to Querying. Since time is expressed in minutes, there's a constraint on Querying to finish in less than a second or in other words 0.016 minutes. This is in order to show that the Querying state is not actually taking a full minute but is a rather fast operation. After Querying and Updating states transition back to Idle, now the state of the Admin actor changes from Idle to Refreshed with the meaning that the visualized stats are now refreshed.

The following diagram is the same timing diagram but in expanded view, although the constraints make it harder to read. On the arrow going from DataViewer to DataLoader there's the method call query() which returns data, while on the arrow going from DataViewer to Admin there's readable data as return value. On the bottom the timing is expressed in minutes.



Time in
minutes

Package Diagram

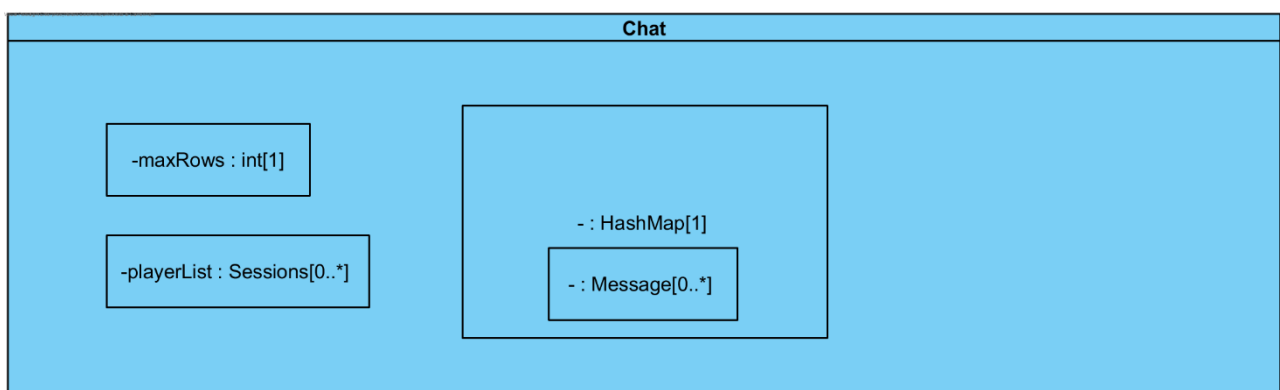


The package diagram shows the relationship between the various packages of the class diagram and the public members of such packages. The package AdminPanel contains a single AdminPanel class which is public and has use dependencies on the subsystems UserManagement and GameManagement, which are not part of our component. Likewise, the Statistics package has only DataViewer class as public member and has the same dependencies on UserManagement and GameManagement. Different is the case for the Chat package. It shows the Chat class as its only public element but it is logically contained in the UserManagement component as shown by the containment relationship.

Composite structure diagrams

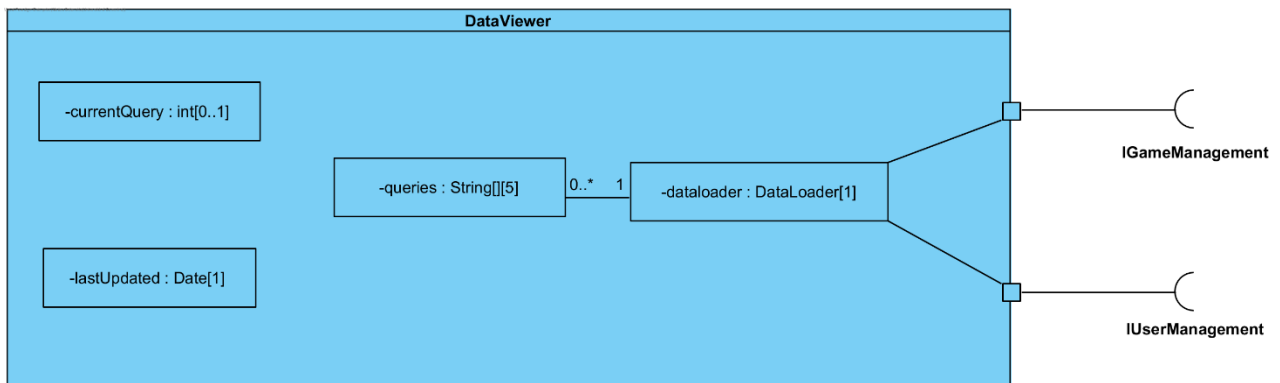
The composite structure diagram show the inner workings of the classes of the component. The following diagrams are relative to the Chat class and the DataViewer class.

Chat Class



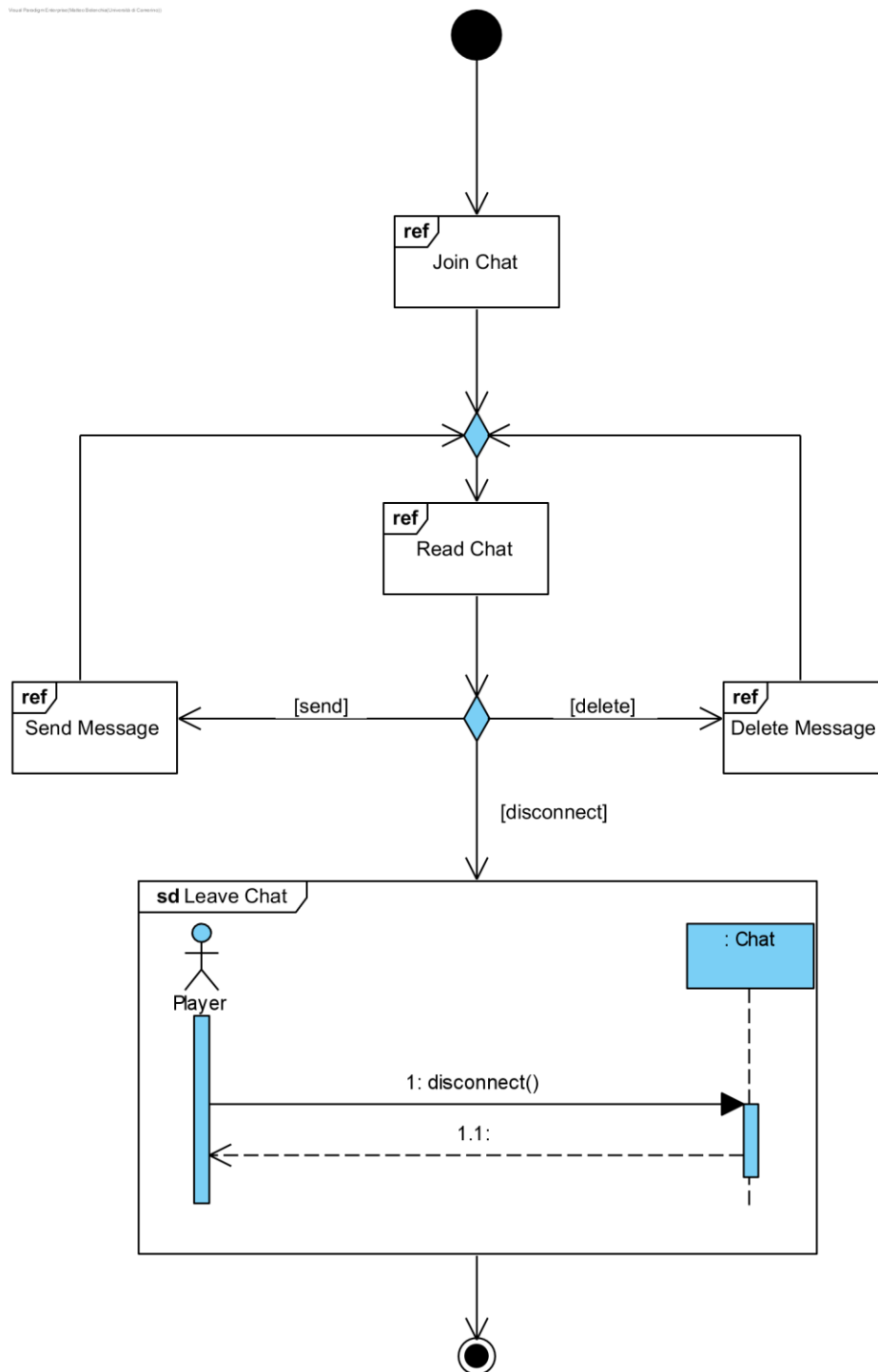
The Chat class contains the attribute maxRows with a multiplicity of 1, and the attribute playerList which can contain 0 to many sessions. Additionally it contains an HashMap object with multiplicity 1 which in turn contains 0 to many Message objects. The Chat doesn't have foreign dependencies.

DataViewer Class



The **DataViewer** class contains a **currentQuery** attribute with multiplicity 1 but which can be null at the beginning, and a **lastUpdated** attribute with multiplicity 1. Then it contains a **queries** attribute which is a collection of 5 String arrays containing SQL queries. These queries are associated with a single **DataLoader** object with multiplicity 1 which is connected to 2 pins mapping to external interfaces **IGameManagement** and **IUserManagement**, provided by the corresponding components. These interfaces are required by the **DataLoader** object in order to work.

Interaction Overview Diagram



The interaction overview diagram shows an overview of the operations of the chat. The first activity is a referenced sequence diagram Join Chat that describes the connection to the chat. From there the next activity is described by the Read Chat sequence diagram and from there the flow can split in different ways: if we go for sending a message we reference the Send Message sequence diagram while if we go for deleting messages we reference the Delete Message sequence diagram. From both these activities the control is merged to run again the Read Chat activity in a loop. If, instead, we choose to disconnect, then the Leave Chat sequence diagram describes the disconnection activity and it is reported in full for its simplicity. After that, the interaction overview diagram terminates.

Conclusions

The component has been developed in all the requested functional and non-functional requirements. The major problems with it is that it has been developed using a rather outdated technology of Java websockets (we would have favoured a Python approach or using the Node.js framework) and that the user interface could have used some better graphics. The integration of the component would also be rather difficult because the Statistics pages require a strict data format to be followed in order to work and lack of coordination with the other groups will prove to be decisive in this issue. We decided to use a very broad data format for messages between client and server making the component much more reliant on following a strict format of data compared to having had multiple types of messages (which would have been prohibitively many for the scope of the component).

User documentation

This user guide will provide the relevant information in order to use the component.

Chat

Functional description

The chat provides the functionality of reading, deleting and writing messages. Admin messages are written in red, and Admins can delete any message while normal Players can only delete their own.

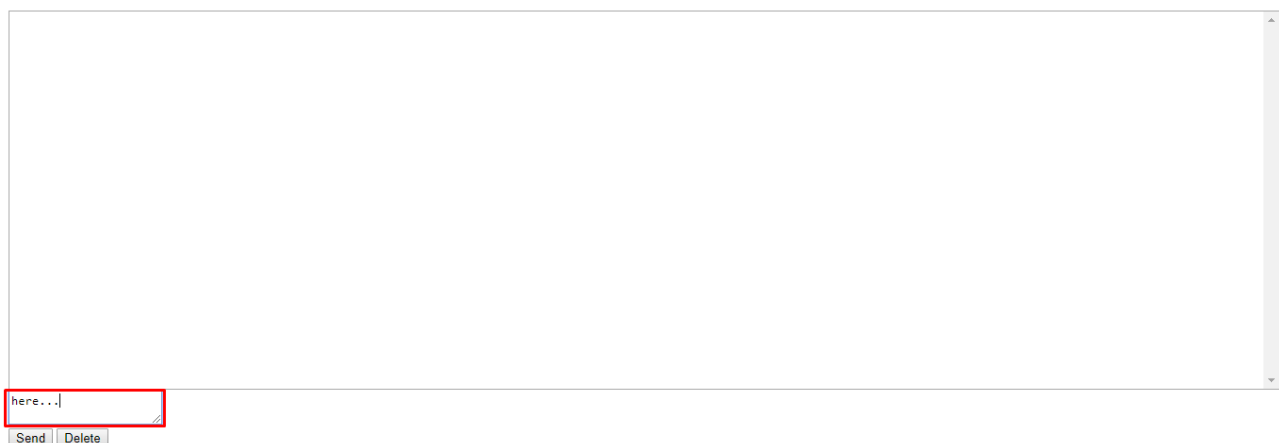
User manual

Connection and Disconnection

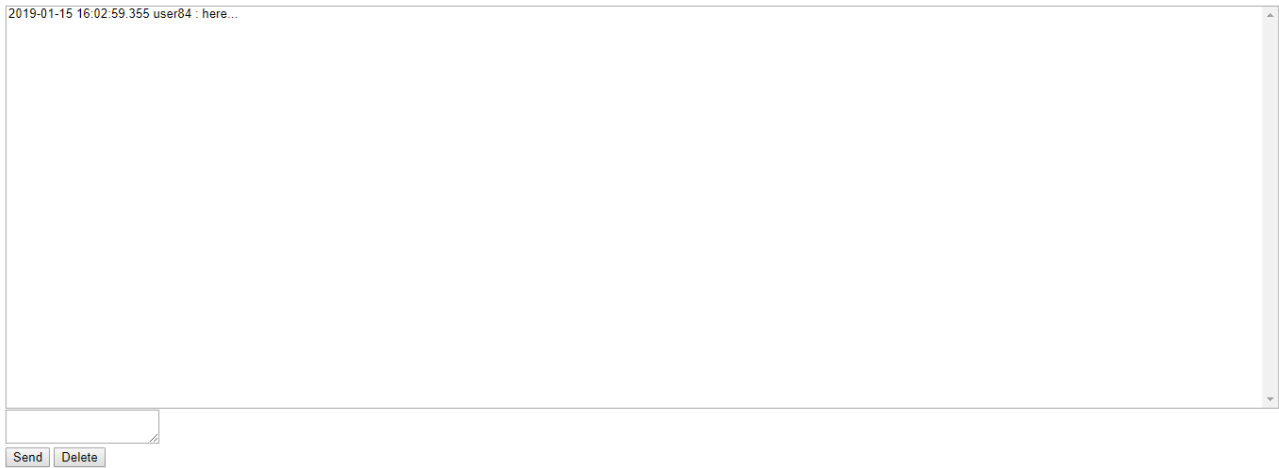
The chat resides on the following webpage: http://localhost:8080/SE_Chat/chat.html

Simply accessing it will connect the Player or Admin to the chat. Disconnection is likewise simple and it's enough to close the webpage. On connection all messages are downloaded.

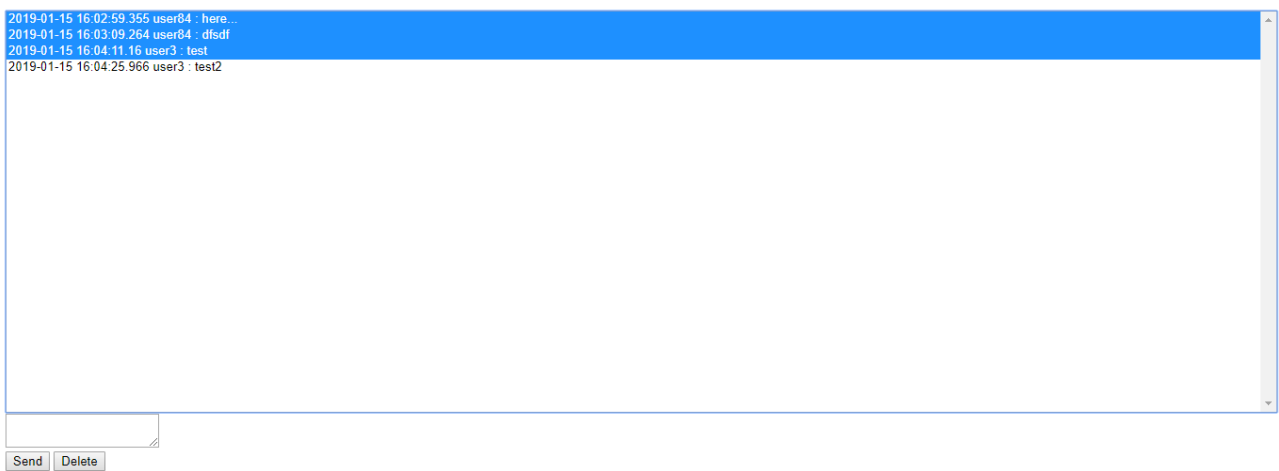
Sending a message

The image shows a web-based chat interface. It consists of a large rectangular area for displaying messages, which is currently empty. Below this area is a text input field containing the placeholder text "here...". This input field is highlighted with a red rectangular border. To the right of the input field is a small red 'X' icon. Below the input field are two buttons: "Send" and "Delete".

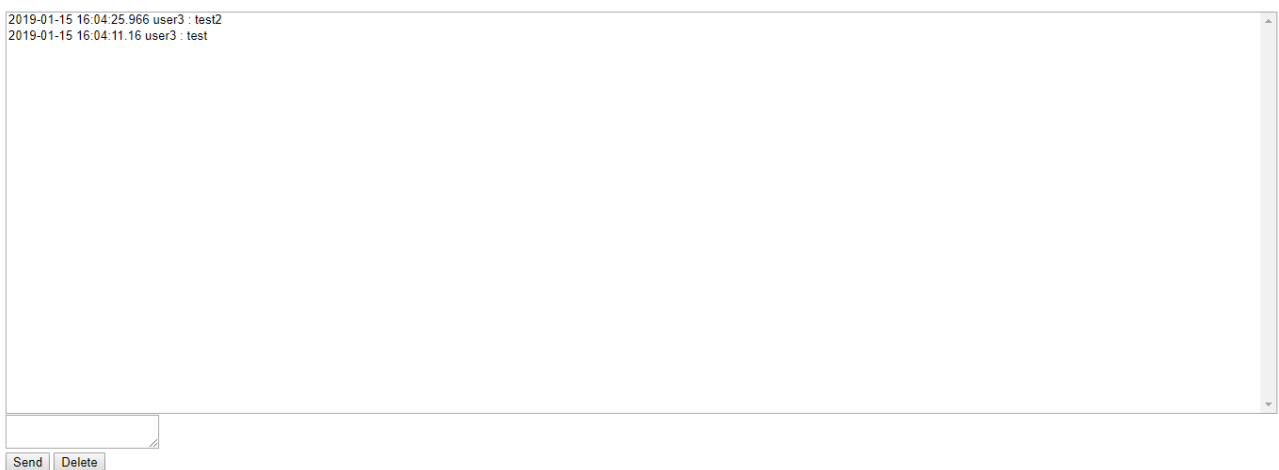
In order to send a message, simply type text in the highlighted textbox and press "Send" button. The message will appear in the chat.



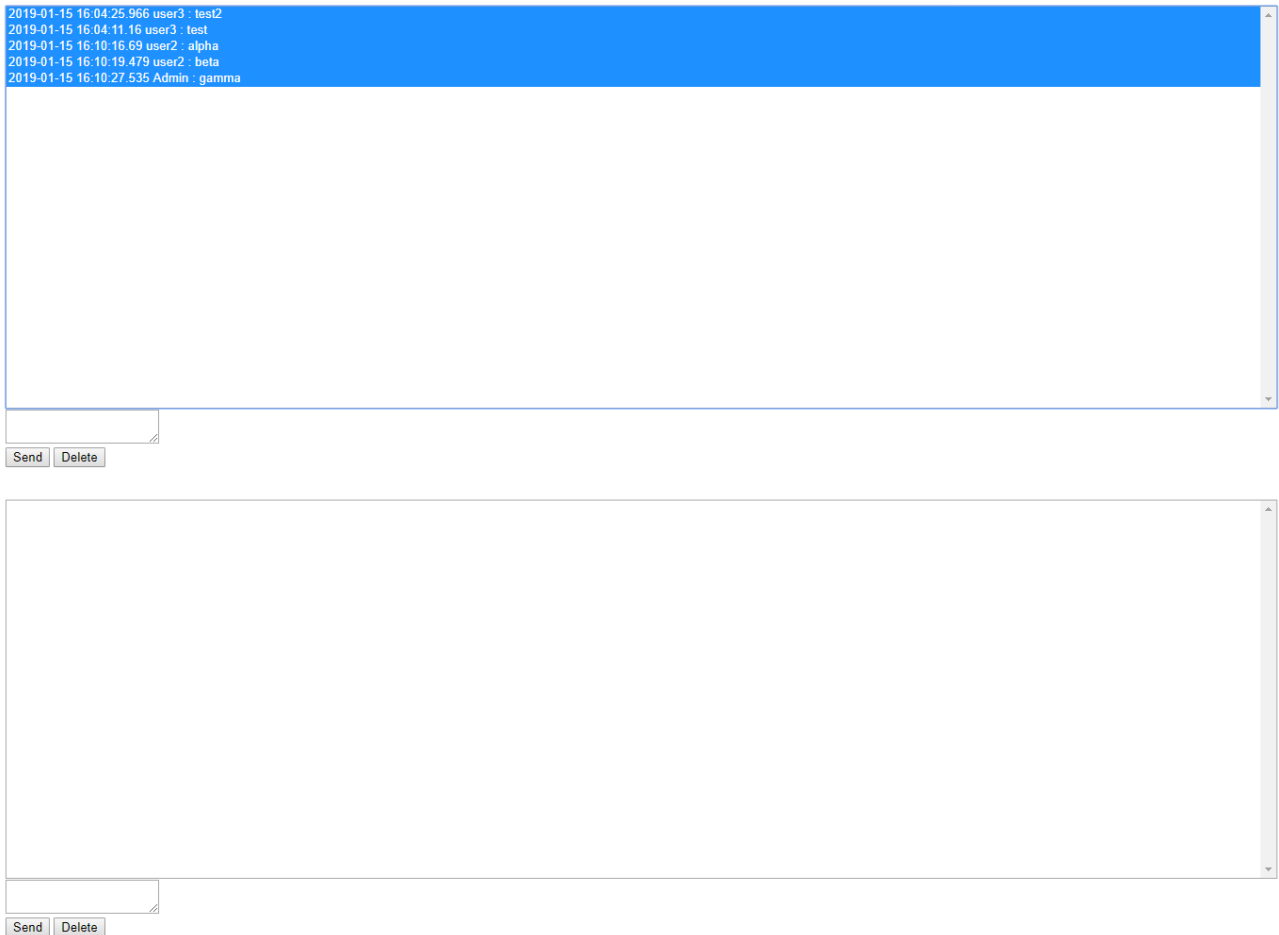
Deleting a message



Before deleting a message, select any number of messages in the chat. Selecting a message is as simple as clicking on it, and to deselect a message press CTRL + Click on the message. When finished selecting, just click the "Delete" button. If the client is a Player, like in the picture above (user84), then only its own messages will be deleted.



Otherwise, if the client is an Admin, all of the selected messages will be deleted:



```
2019-01-15 16:04:25 966 user3 : test2
2019-01-15 16:04:11 16 user3 : test
2019-01-15 16:10:16 69 user2 : alpha
2019-01-15 16:10:19 479 user2 : beta
2019-01-15 16:10:27 535 Admin : gamma
```

Complete description

The chat can display messages, receive new messages and delete messages. Messages are received containing only the type of message, the text and the username, and an ID and timestamp is then given on the server and relayed to all clients. Messages are deleted by sending a special type of message containing a type and a list of integers representing the message IDs to delete. When a new client connects or there is a deletion, the whole chat is sent again to the client or clients, in this case the message contains a type and a list of messages.

Installation description

The chat should be integrated in the User Management component. The installation will need to remove the index.html page which is used to provide a random username and instead provide the Player username stored in the webpage session. It also needs to set the Admin username as "Admin".

Admin Panel

Functional description

The Admin Panel offers functionalities to ban Players, not letting them log in until a certain amount of time expires. It lets the Admin to edit Player properties such as username and email. Admins can set a server capacity, not letting any Player log in until the number of connected Players is below the threshold. Admins can close game session, prompting an end to any game. A minimum bet for every game time can be set, forcing Players to bet from that amount upwards.

On the statistics page, we can see statistics for each game type, statistics for each registered Player, daily statistics, total statistics, and an overview of all daily statistics for each day saved in the system.

User manual

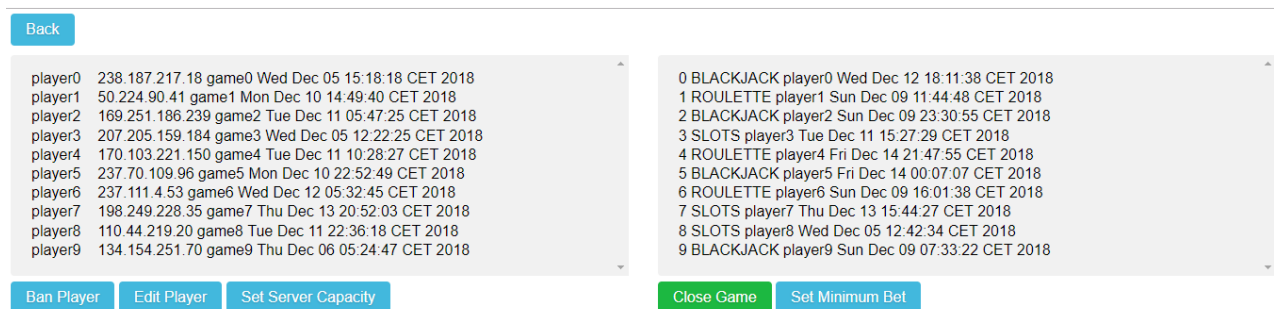
The entry point for the component is the page <http://localhost:8080/Administration/>



On this page, there are 2 buttons to access the relative component: Admin Panel (<http://localhost:8080/Administration/adminPanel/>) and Statistics (<http://localhost:8080/Administration/statisticsPanel/>).

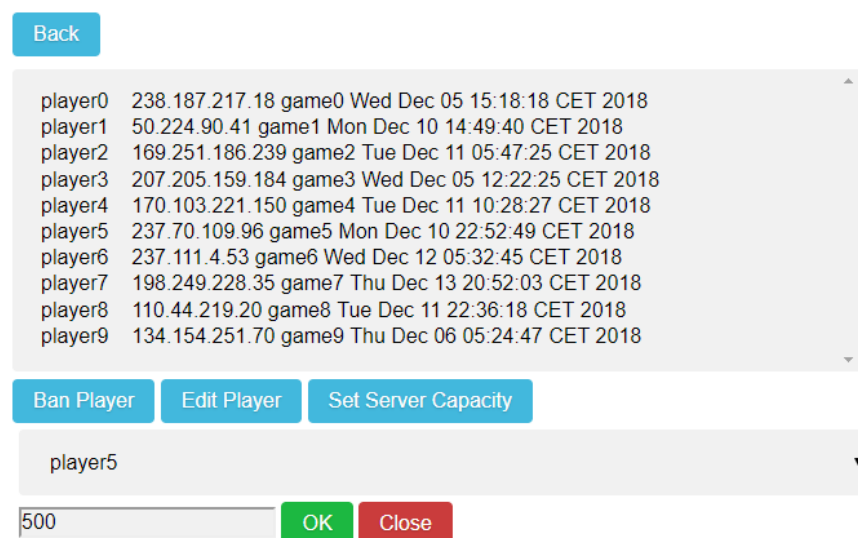
Admin Panel page

On the Admin Panel page, we visualize the currently connected players on the left and currently running game sessions on the right. The top-most button “Back” brings you back to the initial page.



Ban Player

On the left, the button Ban Player summons a form in which you can select any player (even if it's not currently connected) and ban him for the specified number of seconds. The OK button is used to confirm, and close button is used to cancel.



Edit Player

The edit player button, once clicked, shows a form to edit the selected player data. The player is selected by using the select form, which shows all the existing players in the system. You can edit the data then press OK to confirm, or Close to cancel. The username and the email must be unique, if another player shares either of them, the changes are discarded.

Back

player0 169.53.158.57 game0 Fri Dec 07 02:05:48 CET 2018
player1 248.252.226.102 game1 Sat Dec 08 22:08:00 CET 2018
player2 129.215.146.73 game2 Sat Dec 08 01:25:01 CET 2018
player3 60.247.120.69 game3 Tue Dec 11 06:55:08 CET 2018
player4 132.44.152.211 game4 Wed Dec 12 06:37:09 CET 2018
player5 181.89.170.237 game5 Thu Dec 13 04:29:44 CET 2018
player6 150.163.202.40 game6 Thu Dec 06 16:47:59 CET 2018
player7 211.57.80.54 game7 Tue Dec 11 06:05:53 CET 2018
player8 218.34.116.181 game8 Sat Dec 08 06:07:53 CET 2018
player9 74.4.1.149 game9 Tue Dec 11 09:06:39 CET 2018

Ban Player Edit Player Set Server Capacity

player8

player8
player8@gmail.com

OK Close

0 SLOTS player0 Sat Dec 08 12:56:13 CET 2018
1 SLOTS player1 Fri Dec 07 07:58:58 CET 2018
2 BLACKJACK player2 Thu Dec 13 16:30:29 CET 2018
3 BLACKJACK player3 Sun Dec 09 22:06:58 CET 2018
4 BLACKJACK player4 Mon Dec 10 07:17:16 CET 2018
5 SLOTS player5 Mon Dec 10 15:28:22 CET 2018
6 SLOTS player6 Wed Dec 12 21:15:46 CET 2018
7 ROULETTE player7 Thu Dec 06 16:54:01 CET 2018
8 ROULETTE player8 Thu Dec 06 20:56:50 CET 2018
9 BLACKJACK player9 Thu Dec 13 07:11:20 CET 2018

Close Game Set Minimum Bet

Set Server Capacity

The Set Server Capacity button lets you enter a number which will be the maximum number of concurrently connected players allowed. Players are not kicked if the number is lower than the number of currently connected users though. In fact, no one will be allowed to log in until the number of connected players is lower than the number being set. Press OK to confirm or Close to cancel.

Back

player0 169.53.158.57 game0 Fri Dec 07 02:05:48 CET 2018
player1 248.252.226.102 game1 Sat Dec 08 22:08:00 CET 2018
player2 129.215.146.73 game2 Sat Dec 08 01:25:01 CET 2018
player3 60.247.120.69 game3 Tue Dec 11 06:55:08 CET 2018
player4 132.44.152.211 game4 Wed Dec 12 06:37:09 CET 2018
player5 181.89.170.237 game5 Thu Dec 13 04:29:44 CET 2018
player6 150.163.202.40 game6 Thu Dec 06 16:47:59 CET 2018
player7 211.57.80.54 game7 Tue Dec 11 06:05:53 CET 2018
player8 218.34.116.181 game8 Sat Dec 08 06:07:53 CET 2018
player9 74.4.1.149 game9 Tue Dec 11 09:06:39 CET 2018

Ban Player Edit Player Set Server Capacity

5

OK Close

0 SLOTS player0 Sat Dec 08 12:56:13 CET 2018
1 SLOTS player1 Fri Dec 07 07:58:58 CET 2018
2 BLACKJACK player2 Thu Dec 13 16:30:29 CET 2018
3 BLACKJACK player3 Sun Dec 09 22:06:58 CET 2018
4 BLACKJACK player4 Mon Dec 10 07:17:16 CET 2018
5 SLOTS player5 Mon Dec 10 15:28:22 CET 2018
6 SLOTS player6 Wed Dec 12 21:15:46 CET 2018
7 ROULETTE player7 Thu Dec 06 16:54:01 CET 2018
8 ROULETTE player8 Thu Dec 06 20:56:50 CET 2018
9 BLACKJACK player9 Thu Dec 13 07:11:20 CET 2018

Close Game Set Minimum Bet

Close game

On the right, the Close Game button closes the games that are selected from the above form. You can close more than one game at the same time, by selecting multiple games by dragging or holding CTRL. CTRL + Click on a game to deselect.

0 SLOTS player0 Sat Dec 08 12:56:13 CET 2018
1 SLOTS player1 Fri Dec 07 07:58:58 CET 2018
2 BLACKJACK player2 Thu Dec 13 16:30:29 CET 2018
3 BLACKJACK player3 Sun Dec 09 22:06:58 CET 2018
4 BLACKJACK player4 Mon Dec 10 07:17:16 CET 2018
5 SLOTS player5 Mon Dec 10 15:28:22 CET 2018
6 SLOTS player6 Wed Dec 12 21:15:46 CET 2018
7 ROULETTE player7 Thu Dec 06 16:54:01 CET 2018
8 ROULETTE player8 Thu Dec 06 20:56:50 CET 2018
9 BLACKJACK player9 Thu Dec 13 07:11:20 CET 2018

Close Game

Set Minimum Bet

Set Minimum Bet

The Set Minimum Bet button lets you set the minimum bet for each game. Press OK to confirm or close to Cancel.

Back

player0 169.53.158.57 game0 Fri Dec 07 02:05:48 CET 2018
player1 248.252.226.102 game1 Sat Dec 08 22:08:00 CET 2018
player2 129.215.146.73 game2 Sat Dec 08 01:25:01 CET 2018
player3 60.247.120.69 game3 Tue Dec 11 06:55:08 CET 2018
player4 132.44.152.211 game4 Wed Dec 12 06:37:09 CET 2018
player5 181.89.170.237 game5 Thu Dec 13 04:29:44 CET 2018
player6 150.163.202.40 game6 Thu Dec 06 16:47:59 CET 2018
player7 211.57.80.54 game7 Tue Dec 11 06:05:53 CET 2018
player8 218.34.116.181 game8 Sat Dec 08 06:07:53 CET 2018
player9 74.4.1.149 game9 Tue Dec 11 09:06:39 CET 2018

Ban Player

Edit Player

Set Server Capacity

0 SLOTS player0 Sat Dec 08 12:56:13 CET 2018
1 SLOTS player1 Fri Dec 07 07:58:58 CET 2018
2 BLACKJACK player2 Thu Dec 13 16:30:29 CET 2018
3 BLACKJACK player3 Sun Dec 09 22:06:58 CET 2018
4 BLACKJACK player4 Mon Dec 10 07:17:16 CET 2018
5 SLOTS player5 Mon Dec 10 15:28:22 CET 2018
6 SLOTS player6 Wed Dec 12 21:15:46 CET 2018
7 ROULETTE player7 Thu Dec 06 16:54:01 CET 2018
8 ROULETTE player8 Thu Dec 06 20:56:50 CET 2018
9 BLACKJACK player9 Thu Dec 13 07:11:20 CET 2018

Close Game

Set Minimum Bet

Blackjack

OK

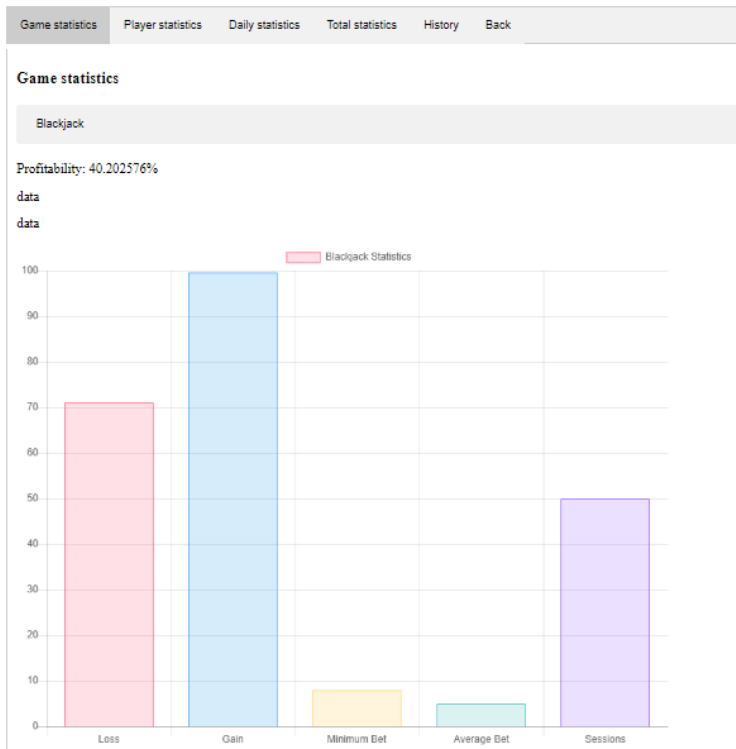
Close

Statistics Page

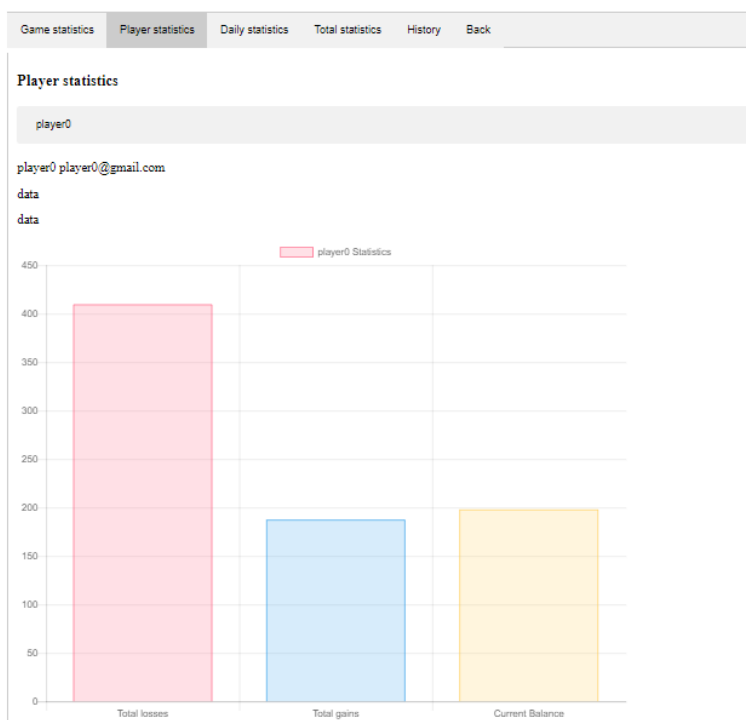
On the Statistics page, there are 5 different tabs to show different statistics, plus a final Back tab that brings you back to the initial page.

Game statistics
Player statistics
Daily statistics
Total statistics
History
Back

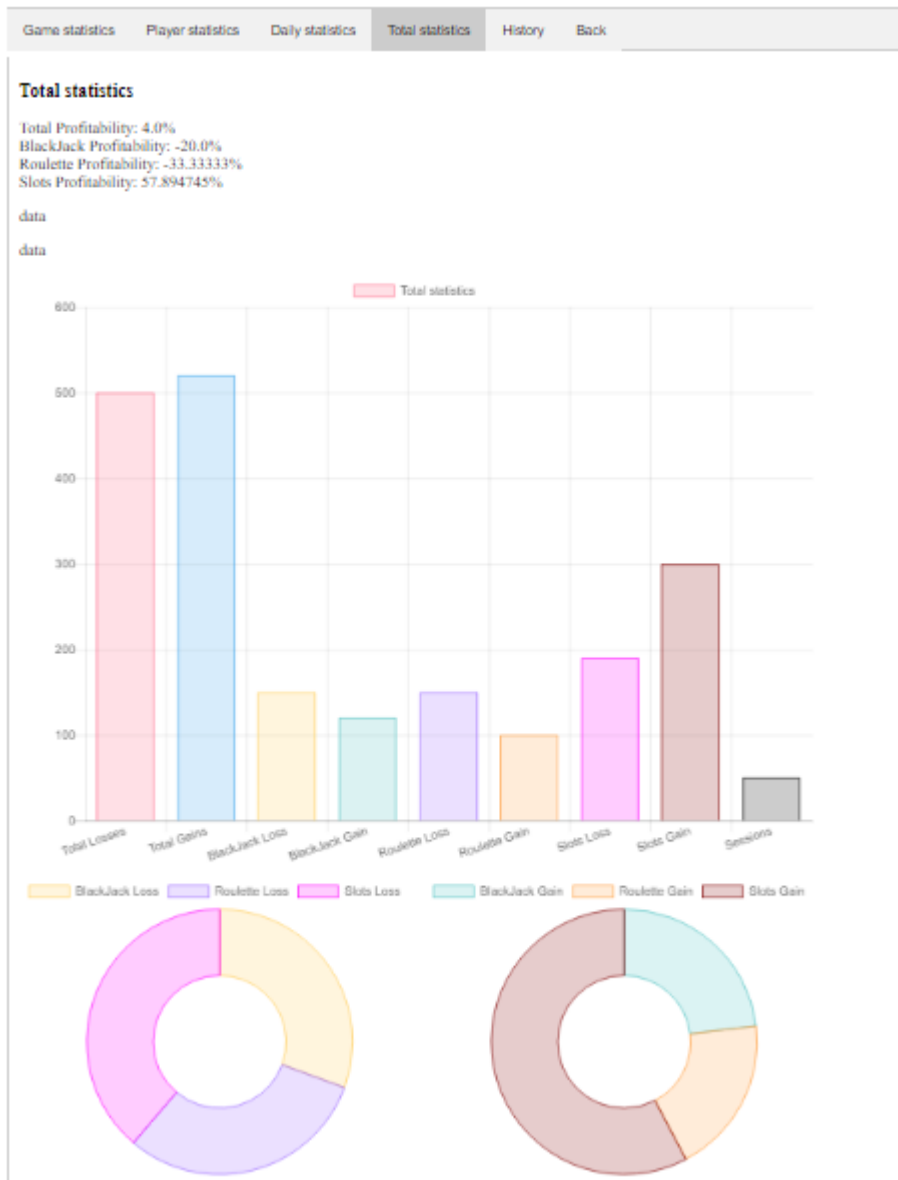
Clicking on the Game Statistics tab opens a view which shows statistics for each game type, selectable with the select form.



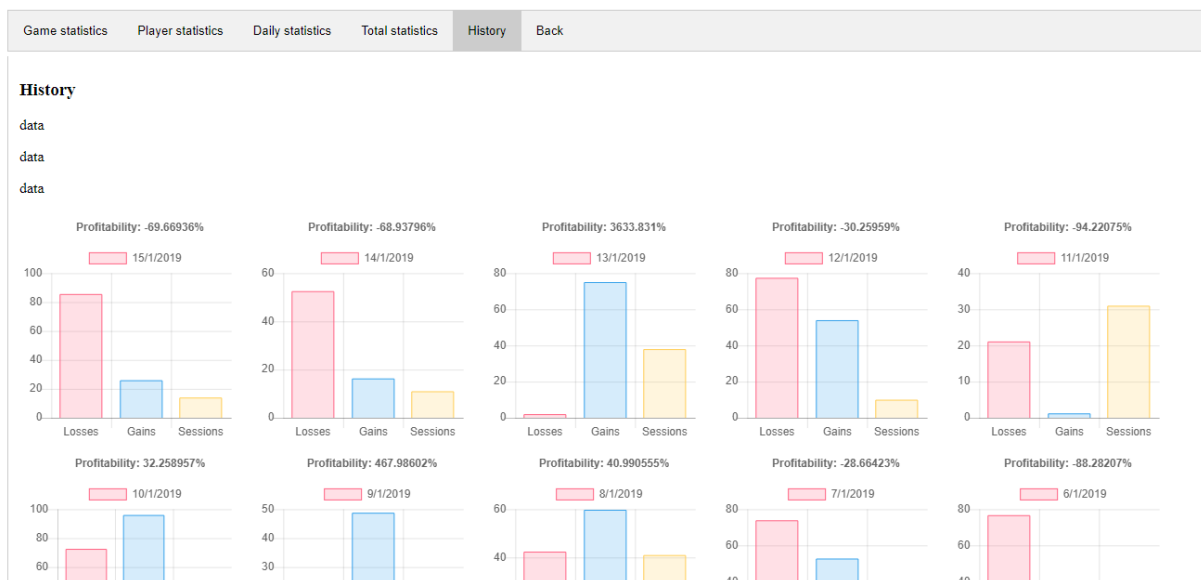
Clicking on the Player Statistics tab opens a view which shows statistics for each player in the system, selectable with the select form.



Clicking on Daily statistics or Total statistics shows either the data for today or the data for all the values present in the application database.



Lastly, the History tab shows an overview day by day of all the days the application has been running in the last month.



Complete description

The component uses a very broad data format. Messages sent to the server are decoded with having a type and then all the data contained in a List of String arrays, which we resolved to be the most flexible data container. Messages from server to client are instead JSON arrays with a type in first position and a variable number of data positions. For this reason the data format must be respected very precisely for it to work.

The AdminPanel page can ban players, close game sessions, edit players, set the minimum bet and set the maximum number of connected players. It also shows live the players connected and the games running.

The statistics page shows the Player statistics, with the anagraphic data, the total gains and total losses. The game statistics show for each game gains and losses for the house, minimum and average bet, the total number of sessions and the profitability. Daily and total statistics pages show total losses, total gains, loss and gains for each game type and number of sessions. Additionally donut charts show the distribution of loss and gains per game type. Also total and per game profitability is shown. The difference between daily and total statistics is that the former is about the current day while the latter queries all the data in the database. The history page shows a basic gains, losses and number of session chart for everyday present in the database, along with a profitability value.

Profitability is calculated as

$$P = \frac{gains}{losses} \cdot 100 - 100$$

and represents how much was gained or lost after repaying the losses.

Installation description

In order to be installed it is necessary to connect the actual implemented interfaces in place of UserManagementPuppet and GameManagementPuppet. Furthermore, return values of statistics method must match exactly those of the puppet classes because of the message formatting issues described in the previous section.