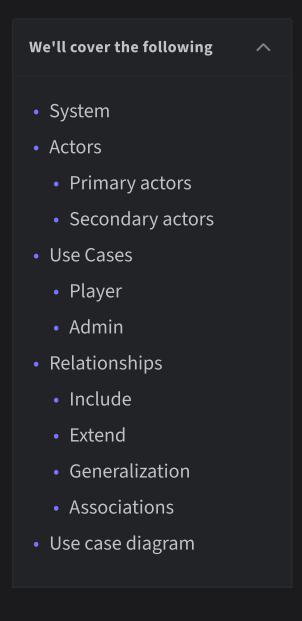
Use Case Diagram for the Chess Game

Learn how to define use cases and create the corresponding use case diagram for the chess game.



between its different components.

First, let's define the different elements of our chess game, followed by the complete use case diagram of the system.

Let's build the use case diagram of the chess game and understand the relationship

System
Our system is a "Chess game."

Actors

Here are the main actors of our chess game.

Primary actors

• Player: This is the primary user and is responsible for playing the chess game. It

can start a new game, make move or resign/forfeit a game.

shared among different actors in the system.

update account information of an existing account

Create a new game: To create a new game to start playing

• **Admin:** This can add, remove, or update a player's account and membership, view open games, and validate player moves.

Secondary actors

Use Cases
In this section, we'll define the use cases for the chess game. We have listed down the

use cases according to their respective interactions with a particular actor.

Player ______

• Create/Update account: To create a new account to play a chess game or to

Note: You will see some use cases occurring multiple times because they are

Login/Logout: To log in to or log out from an account
View open games: View available games that are waiting for players to join

• Join a game: Join an open game

Admin

game

- Make move: To make a move in the game
 Resign or forfeit a game: To resign or forfeit a game so that it ends
- Block/unblock member: To block or unblock a member from playing the chess

• Login/Logout: To log in to or log out from an account

• Cancel/Update membership: To cancel the membership or to update the membership of an account

Add/modify member: To add a new member or update member information

cases.

- View open games: View available games that are waiting for players to join
 Validate Moves: To validate player move
- Relationships

This section describes the relationships between and among actors and their use

Declare results: To declare the result of the game when the game is over

Include

• The "Resign or forfeit a game" use case has an include relationshop with the "Declare result" use case because the game will be over, and results will be

the admin has to validate if the move was as per the rules set.

declared when a player either resigns or forfeit from the game.

• The "Block/unblock member" has an extend relationship with the

member, its membership might be canceled.

• The "Make move" has an include relationship with the "Validate move" because

Extend

make any of these six moves.

Login/Logout

Here is the use case diagram of the chess game:

Use case diagram

Associations

result" use case as there is a chance that the game will be over and results will be declared when a player makes a validate move and checkmate another player.

Generalization

The "Make move" has a generalization relationship with the "Play pawn", "Play

bishop", "Play king", "Play queen", "Play knight", and "Play rook", since a player can

The table below shows the association relationship between actors and their use

• The "Validate move" use case has an extend relationshop with the "Declare

"Cancel/Update membership" since when the admin unblocks a member, there

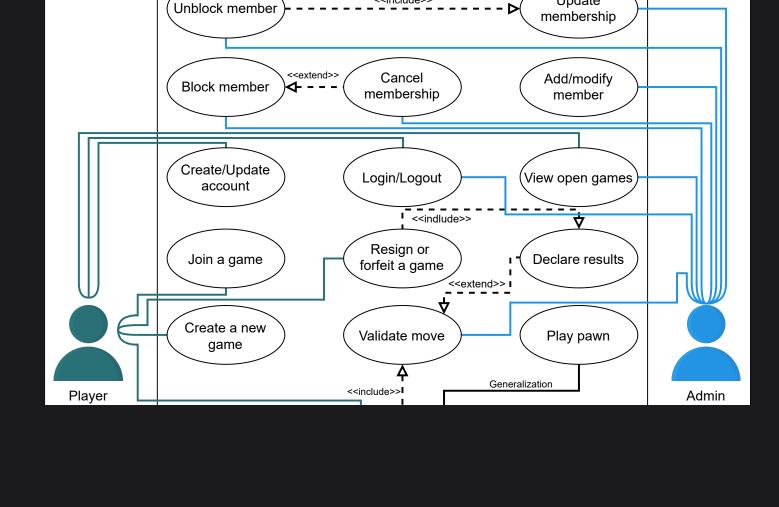
is a chance that their membership will be updated. When the admin blocks a

Cases.

Player Admin

Create/Update account	Block/unblock member
Join a game	Cancel/Update membership
Create a new game	Add/modify member
Make move	Login/Logout
Resign or forfeit a game	Validate Moves
View open games	View open games

Declare results



Chess game <<include>>

Update