

# Requirements Definition *version 2*

*(November 27th, 2022)*

Project Code Defenders - Robo Tournament  
Team Codebenders

# Table of content

- Project background (*refer to Project Plan document for details*)
  - Project Vision, about Code Defenders
- Project requirements
  - User Stories
  - Non functional requirements
- Initial product backlog

# Project vision



- Software quality and testing are at the heart of software engineering, but they may not always get enough attention from software engineering education.
- CodeDefenders (web game) proposes the use of **gamification** to teach **mutation testing** and to strengthen code writing and testing skills.
- The game supports **team play and competition** by having Attackers - Defenders teams whose goal is to inject errors into code or write unit tests to catch them.
- The “**CodeDefenders: RoboTournament**” project aims at enriching the game by adding support for students tournaments and games against bots.

Game 115 (Attacker)

Scoreboard

Timeline

Gradle Export

Feedback

Editor Mode: default

Chat

Existing Mutants

All Alive Killed Claimed Equivalent Equivalent

23 All Mutants			
Mutant 2131	by grant	Modified line 4, line 6	Points: 45
Mutant 2132	by grant	Modified line 7	Points: 0
Mutant 2133	by grant	Modified line 9	Points: 0
Mutant 2134	by grant	Modified line 5	Points: 0
Mutant 2238	by sianico	Modified line 4	Points: 0
Mutant 2239	by sianico	Modified line 4, line 6	Points: 0
Mutant 249	by kJac	Modified line 6	Points: 1
Mutant 251	by akhanfir	Modified line 7	Points: 1

Create a mutant here

Reset Attack

```
1 public class SimpleExamples {
2
3     public static int max(int a, int b, int c){
4         if (a >= b && a >= c)
5             return a;
6         else if (b >= a && b >= c)
7             return b;
8         else
9             return c;
10    }
11 }
12 }
```

Game 115 (Defender)

Scoreboard

Timeline

Gradle Export

Feedback

Editor Mode: default

Chat

Class Under Test

```
1 public class SimpleExamples {
2
3     public static int max(int a, int b, int c){
4         if (a >= b && a >= c)
5             return a;
6         else if (b >= a && b >= c)
7             return b;
8         else
9             return c;
10    }
11 }
12 }
```

Live Killed Claimed Equivalent Equivalent

Mutant restrictions: Moderate

Write a new JUnit test here

Defend

```
1 import org.junit.Test;
2
3 import static org.junit.Assert.*;
4 import static org.hamcrest.MatcherAssert.assertThat;
5 import static org.hamcrest.Matchers.*;
6
7 public class TestSimpleExamples {
8     @Test(timeout = 4000)
9     public void test() throws Throwable {
10         // test here!
11     }
12 }
```

Existing Mutants

All Alive Killed Claimed Equivalent Equivalent

23 All Mutants			
Mutant 2131	by grant	Modified line 4, line 6	Points: 45
Mutant 2132	by grant	Modified line 7	Points: 0

JUnit Tests

45 All Tests

44 max(int, int, int)

# Project requirements

- Implement a **tournament application**. This application must use CodeDefenders as a remote service (through APIs) and must include at least two tournaments modalities.
- Design and implement a set of **OpenAPIs for CodeDefenders** which can be used from the tournament application to manage games and players.
- Implement a **load balancing** mechanism which allows the tournament application to communicate with **multiple CodeDefenders servers** and to always create games on the less loaded server.
- Implement a **streaming** component which allows users to follow in progress games live. This component can optionally include an “overall tournament view” showing schedule, standings and other information for each tournament.
- Design and implement a set of **APIs** which allows users to train **bots** over past games data and to let those bots play CodeDefenders.

# List of requirement

## Tournament App

- Tournament list
- Multiple active tournaments
- Multiple CodeDefenders servers
- Register to tournament app
- Login to tournament app
- New tournament
- Choose tournament type
- Create teams
- Join teams
- Join tournament (single)
- Join tournament (team)
- Schedule tournament matches
- Notify of upcoming match
- Assign teams/players to matches
- Redirect to CodeDefenders
- Redirect to Tournament app
- Restrict CodeDefenders interaction
- Real-time tournament overview
- Match results

## Functional requirements

### Streaming App

- Guests can view streams
- Real-time match overview
- Real-time match updates
- Visual effects
- Visual effects toggle

## Non Functional requirements

### Bots integration

- Bots history
- Bots join matches
- Bots play matches
- Bots get match status

- Load balancing
- Act as a proxy

# User Stories organized in Epics

## Tournament Management

[CDF-32](#) Login/Register  
[CDF-41](#) Display tournaments info  
[CDF-33](#) Create Tournament  
[CDF-34](#) Join tournament  
[CDF-42](#) Matchmaking

## Play tournament games

[CDF-36](#) Starting games with notification  
[CDF-38](#) Return to tournament app from game page

## Watch a streamed tournament game

[CDF-39](#) View game stream  
[CDF-40](#) Notification of game stream update

## Team Management

[CDF-35](#) Team creation  
[CDF-54](#) Team management  
[CDF-37](#) Join team

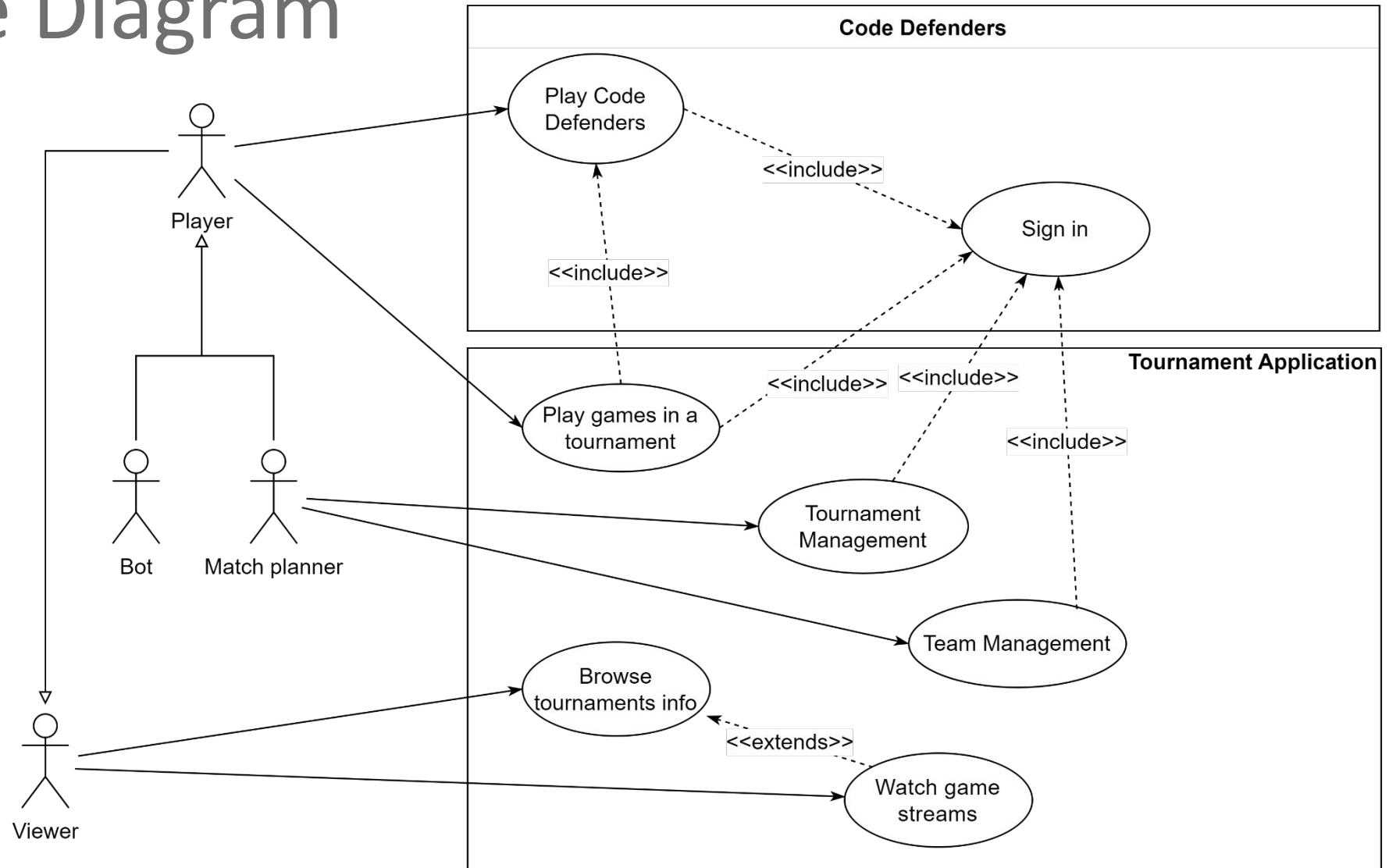
## Play with bots

[CDF-43](#) Bots can play  
[CDF-44](#) Bots can be trained

## Avoid CodeDefenders overload

[CDF-31](#) Low latency  
[CDF-69](#) Efficient flow of updates

# Use Case Diagram



# Epic: Tournament Management

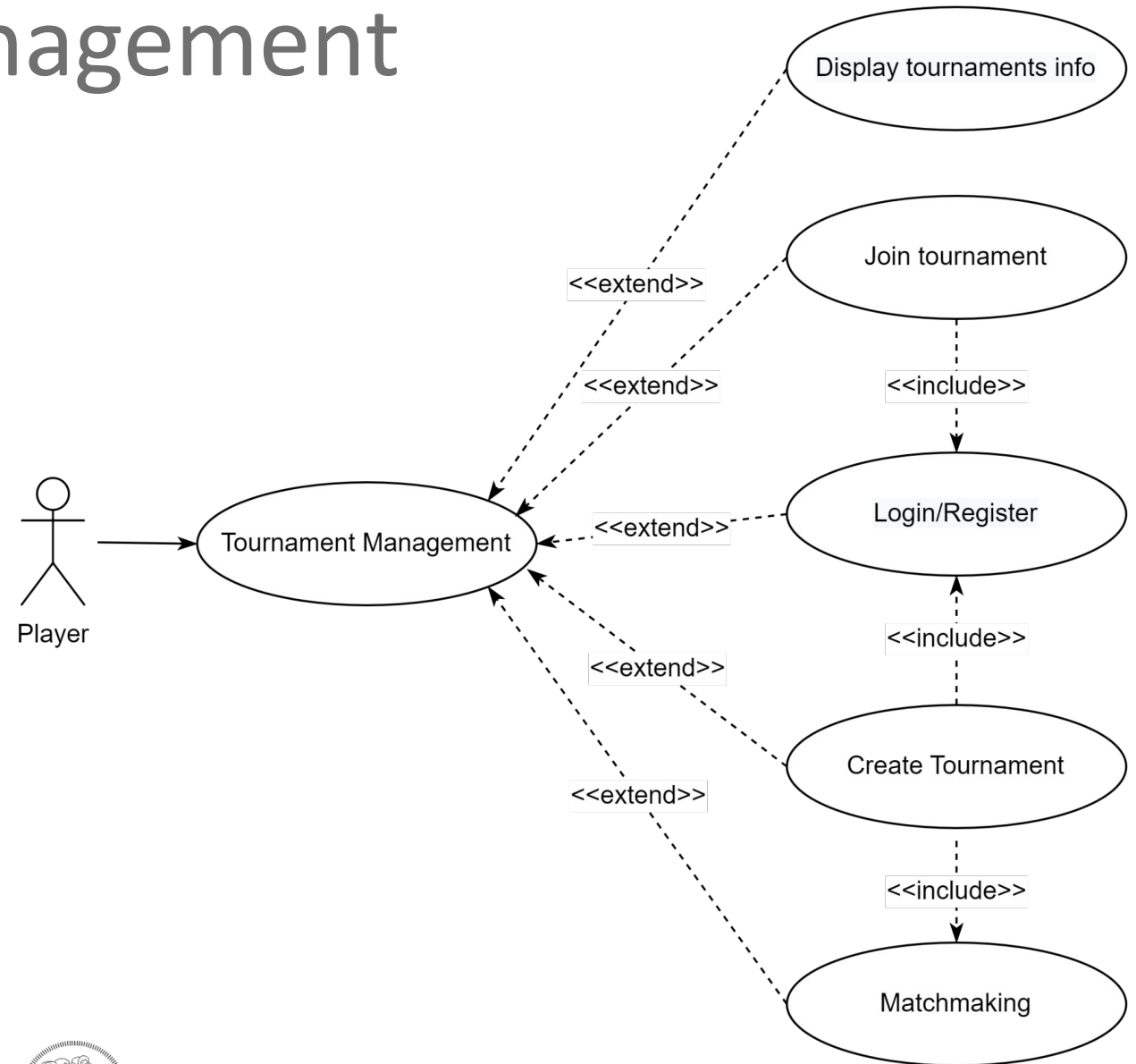
[CDF-32](#) Login/Register

[CDF-41](#) Display tournaments info

[CDF-33](#) Create Tournament

[CDF-34](#) Join tournament

[CDF-42](#) Matchmaking





## CDF-32 Login/Register

Entry condition: homepage

A non-authenticated user wants to authenticate/register to the tournament application to be able to join or create new tournaments.

Exit condition: homepage

Validation: User is authenticated by CodeDefenders and can access pages that require validation

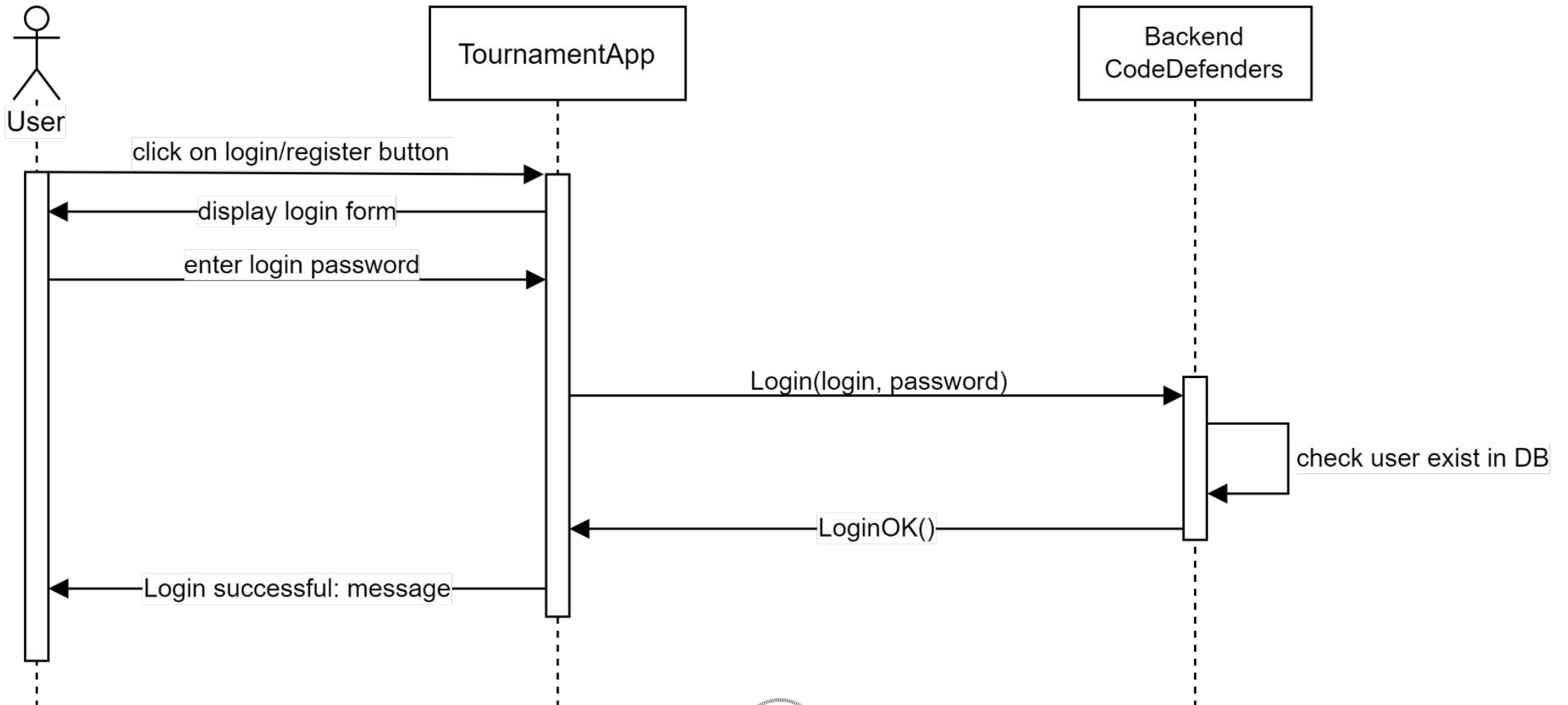
Motivation:

- Only authenticated players can be referenced (who is in which team, who participates in which tournament)

Source: Identified by the team

- 1) In the home page the user clicks on login/register button
- 2) The tournament application shows a login/register page with a form
- 3) The user inserts authentication/registration data
- 4) The tournament app verifies the inserted data through CodeDefenders and sends a response with the outcome to the user, if successful, redirect to homepage

# Login

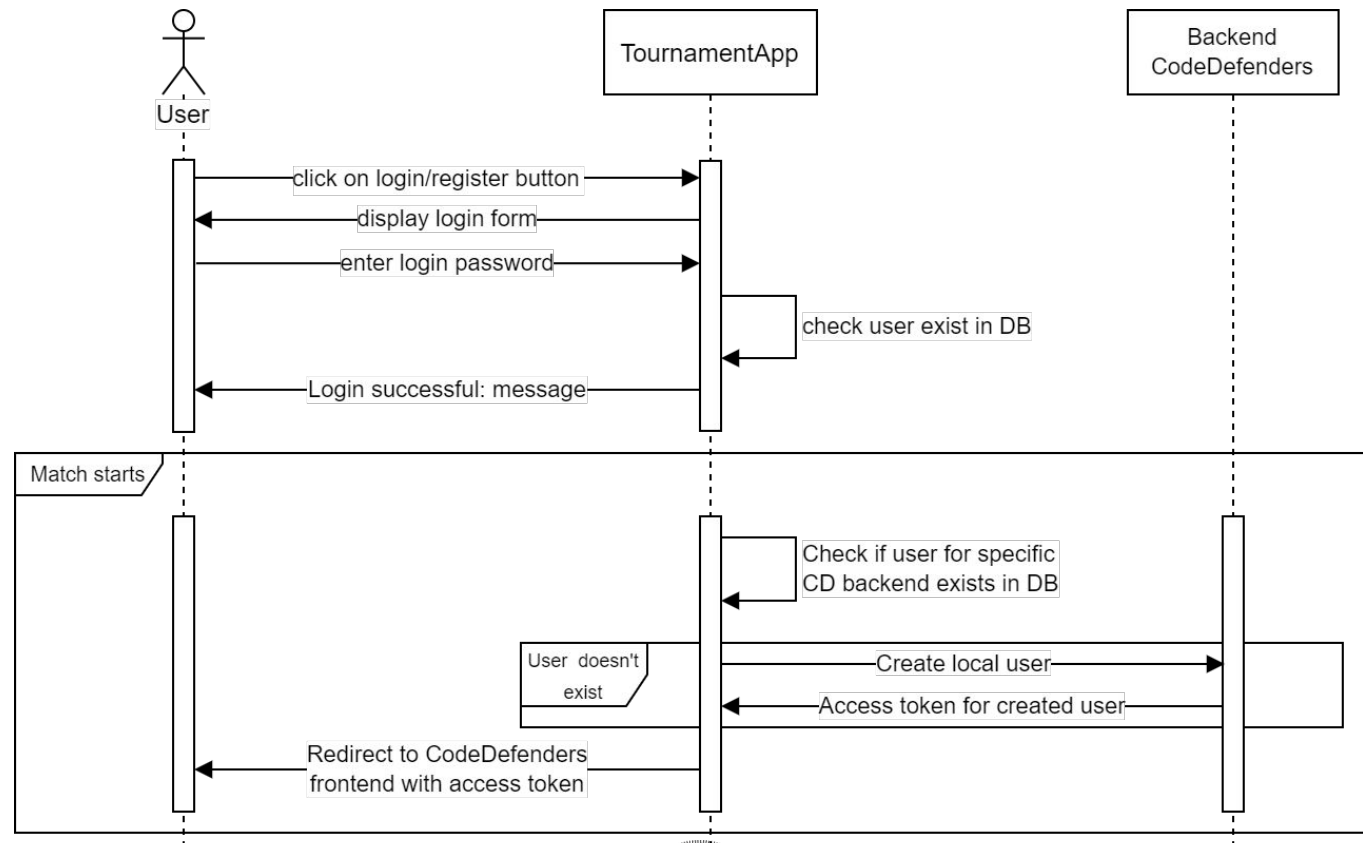


# Authentication sequence

The users register and login on our TournamentApp, with no interaction with the CodeDefenders backend.

When a match starts, we check if our user has a corresponding user on the CodeDefenders instance hosting the game. If it doesn't, we create it and get the access token; otherwise it already exists and we already have the token.

We redirect the user to the target game page on CodeDefenders specifying the authentication token, so that the CodeDefenders login page is skipped



## CDF-41 Display tournaments info

Entry condition: homepage

Players want a way to inspect list of tournaments, filter it by specific players/team, type, date, labels, current status of tournament tree,...

Exit condition: other view (tournament/team creation, FAQ...)

Validation: Any user (authenticated or not) can view all tournaments info, filter and sort the list.

Motivation:

- There needs to be a billboard for tournaments

Source: Customer request

- 1) User sees and possibly reorders and filters the list with all tournaments
- 2) User possibly selects one tournament to see more detailed information

## CDF-33 Create Tournament

Entry condition: User is authenticated, enters create tournament view

Players want to create tournaments of different types, with constraints for joining, conditions for start...

Exit condition: confirm creation, return to homepage

Validation: In the Home page, the created tournament is displayed in the list of tournaments

Motivation:

- There needs to be a way of creating new tournaments with different configurations

Source: Customer request

- 1) In the tournament app home page the user clicks on create Tournament button
- 2) A form for the tournament creation is displayed
- 3) The user selects all tournaments options (type of tournament, game type, number of team, the condition to join/start the tournament)
- 5) The user clicks the button to confirm the creation

# Changes introduced in version 2

CDF-34 Join Tournament, CDF-37 Join team, CDF-35 Team creation

After better analysis of user stories and how they could introduce problems to each other, we decided to set a few constraints:

- a) Players can only be members of one team at a time
- b) Teams can participate in only one tournament at a time
- c) Teams cannot be changed while participating in ongoing tournament

Problems solved:

- 1) Two teams with same players playing the same match on opposite teams
- 2) No need to reject a team's request to join a tournament if there is already accepted team with the same player/s
- 3) Teams being called to play two different matches of different tournaments at the same time, or having dedicated time schedulers for each team

## CDF-34 Join Tournament

Entry condition: User is authenticated and is the leader of the team, homepage

Players want to join already existing tournaments with their teams

Exit condition: confirm registration, return to homepage

Validation: In the Home page, the joined tournament now shows also the new team among the participants

Motivation:

- There needs to be a way to join already existing tournaments so that teams can compete

Source: Customer request

\*) User can only join his/hers team to one tournament at a time

1) In the tournament app home page the user chooses a tournament to join and clicks the join button

2) Pop-out message to confirm registration to the tournament

3) Response message if team entered the tournament or not

## CDF-42 Matchmaking

Entry condition: Tournament view, after the list of all competitors is full

Once new tournament is filled with all the competitors, players want fair pairings of opposing teams.

Exit condition: tournament is completely defined

Validation: Tournaments with mentioned labels can't be edited by creator after the automatic distribution, manual check of formula for distribution based on rating

### Motivation:

- We want to avoid teams with too different skill levels to compete against each others

### Source: Identified by the team

- 1) Tournament creator chooses the matchmaking option for its tournament (random/rating distributed)
- 2) Once the tournament is complete with all the participants, the tournament app estimates rating of players based on their score history
- 3) A math formula is used for computing the pairings and filling the tournament structure
- 4) Tournament gets labelled as "Random distribution" or "Rating distribution"

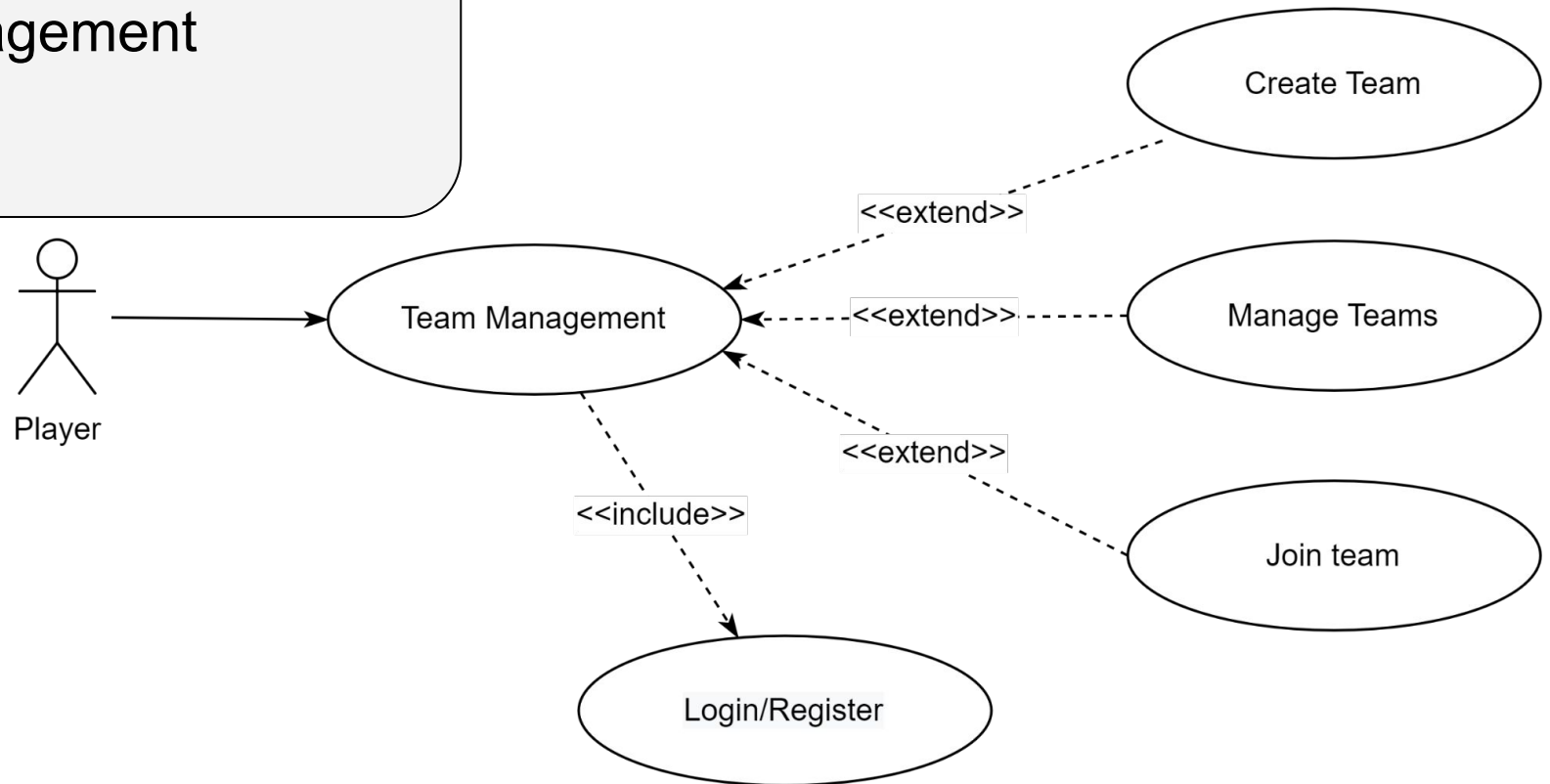


# Epic: Team Management

[CDF-35](#) Team creation

[CDF-54](#) Team management

[CDF-37](#) Join team



## CDF-35 Team creation

Entry condition: User is authenticated, enters create team view, user is not a member of other team

A player wants to create a team and choose its team members

Exit condition: Confirm creation, return to homepage

Validation: The new team is visible in the team management view

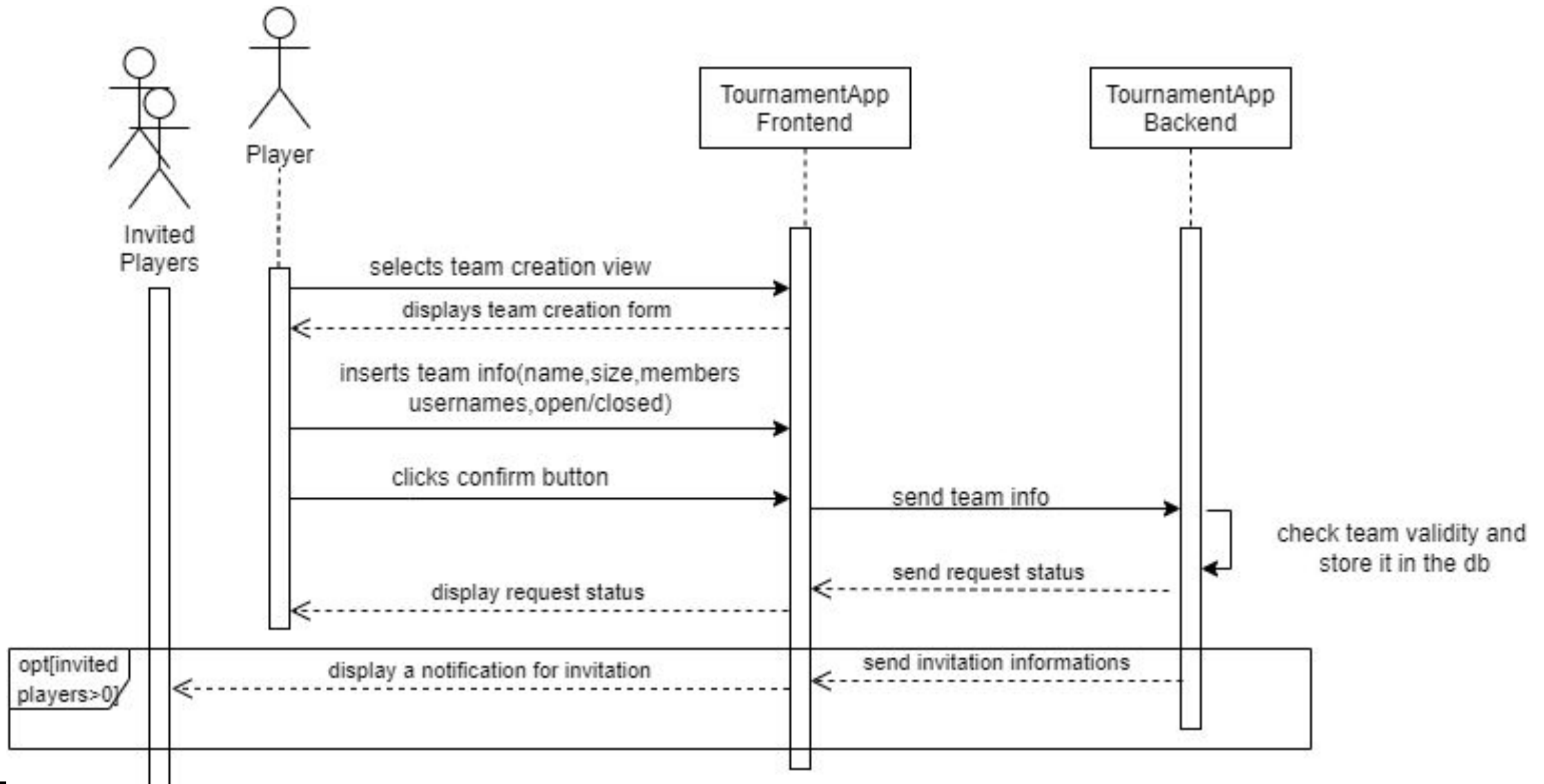
Motivation:

- We want to give a player the possibility to choose the members of its team and to join multiple tournaments with the same team

Source: Identified by the team

- 1) The player enters team creation view
- 2) The player adds team information (name, size, type)
- 3) The player confirms team creation

# Team Creation



## CDF-54 Team management

Entry condition: User is authenticated, is a team leader, enters manage team view and team is currently not participating in a tournament

A team leader wants to change the team settings, add and remove members

Exit condition: Confirm changes, return to homepage

Validation: The changes applied to the team are visible in the team management view

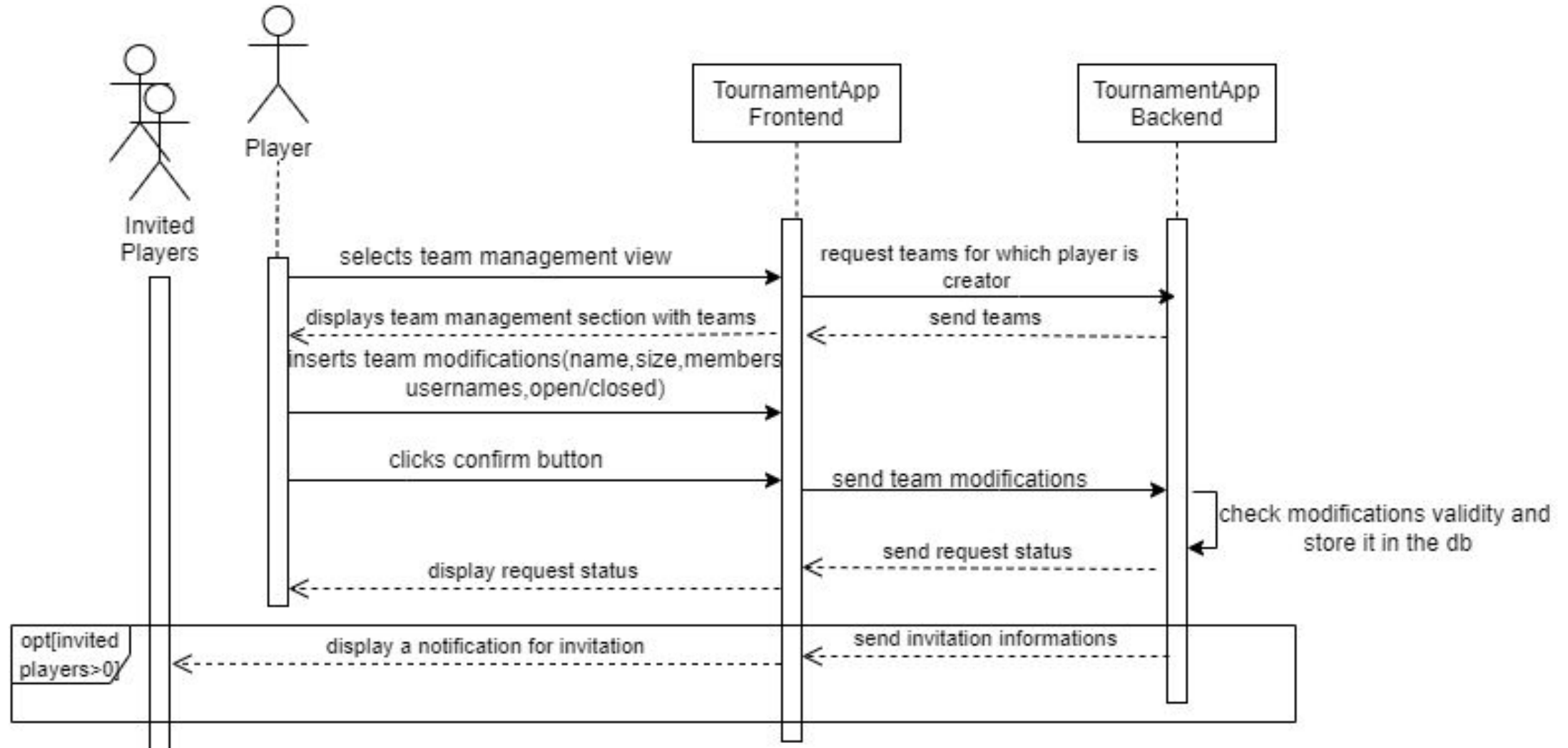
Motivation:

- We want to give a player the possibility to modify settings and members of a team previously created

Source: Identified by the team

- 1) The team leader enters team management view and selects one of its teams
- 2) The team leader possibly changes the team settings (name, size, ...)
- 3) The team leader possibly adds or removes team members
- 4) The team leader confirms the changes

# Team management



## CDF-37 Join team

Entry condition: User is authenticated, enters join team view or receives an invitation

A player wants to join an already existing team

Exit condition: Confirm the team to join or accept the invitation, return to homepage

Exception: If user is already in a team, they are asked if they want to leave that team since one can only be in one team at a time

Validation: The user can see the new team it joined in the list of its teams

Motivation:

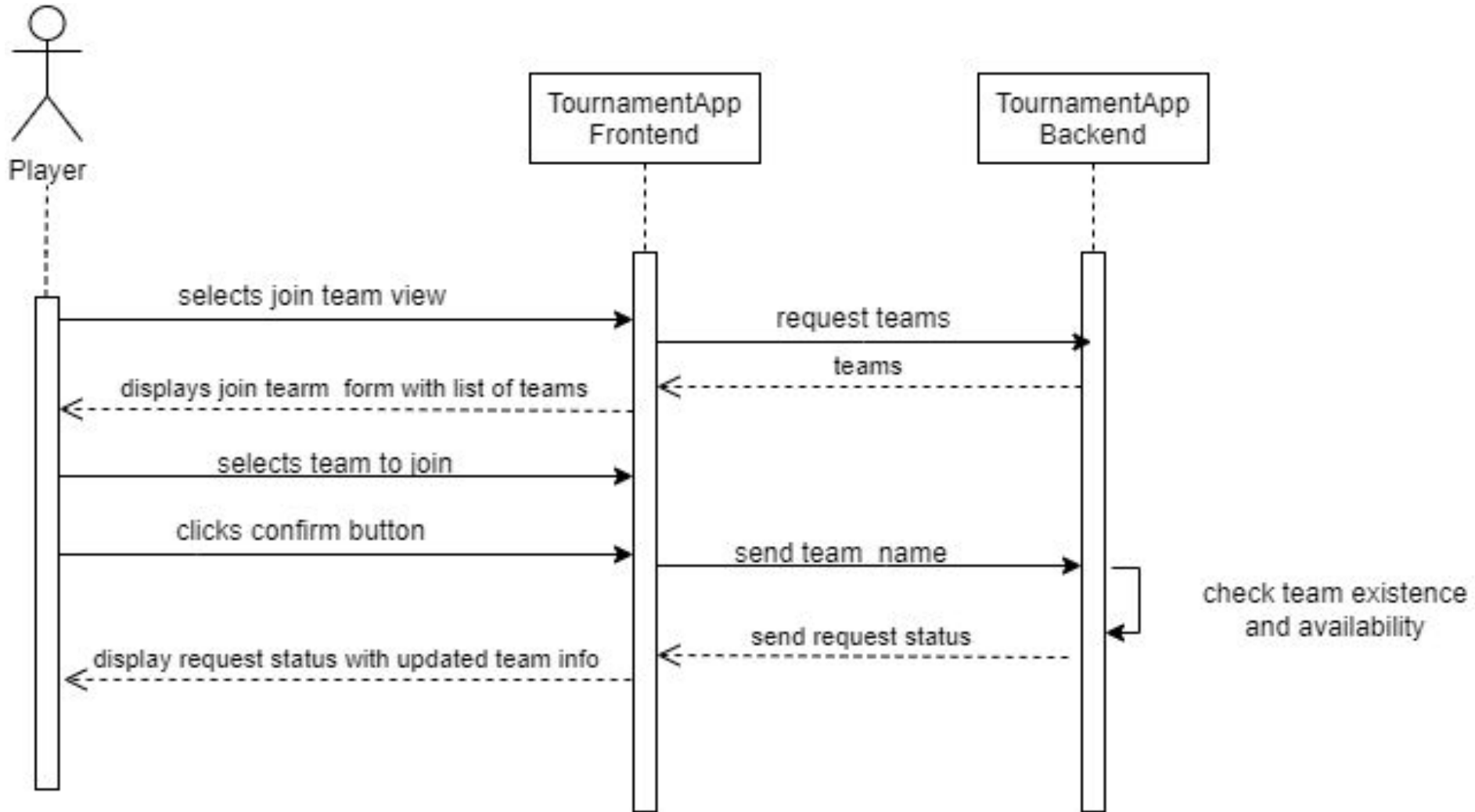
- The player should be able to join a team it likes or a team it has been invited to

Source: Identified by the team

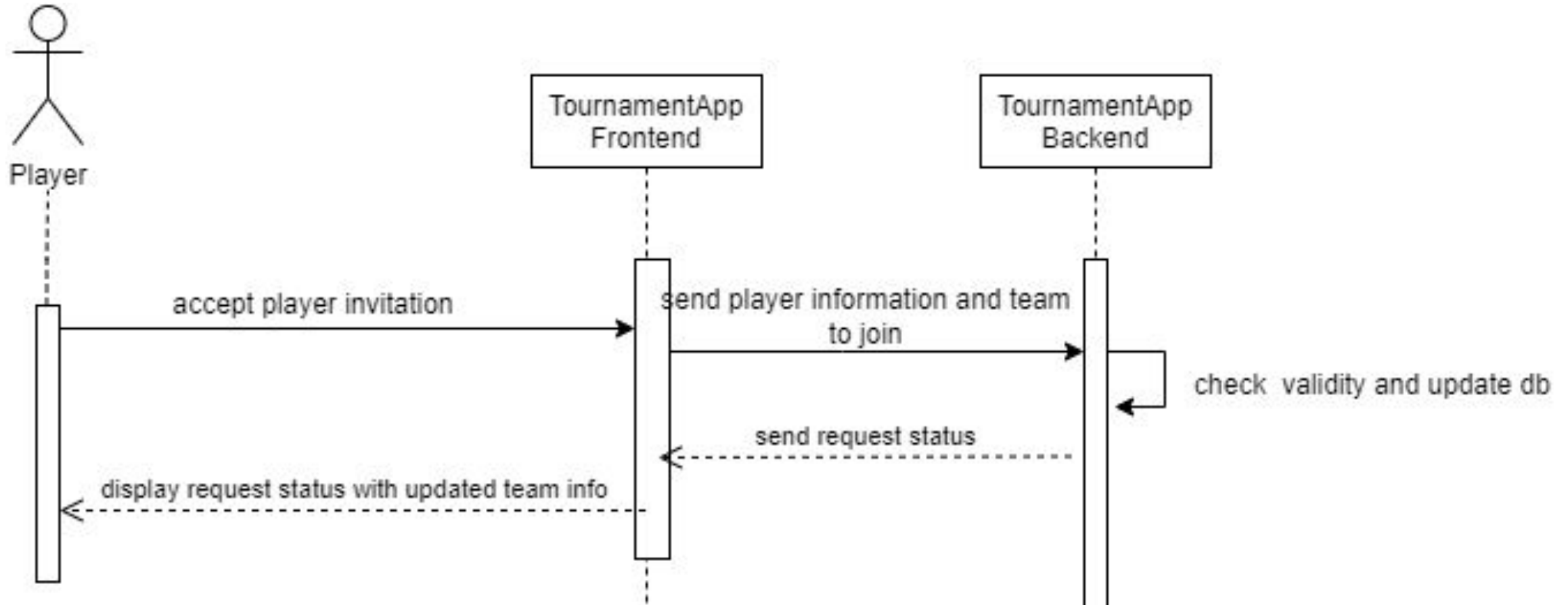
- 1.a) The player enters join team view and sees the list of available teams
- 2.a) The player can reorder the teams (by names, date, ...) and see teams details
- 3.a) The player selects an open team to join and confirms

- 
- 1.b) The player receives an invitation from a team
  - 2.b) The player can see the team details
  - 3.b) The player accepts (or refuses) the invitation

# Join Team



# Accept /Refuse invitation

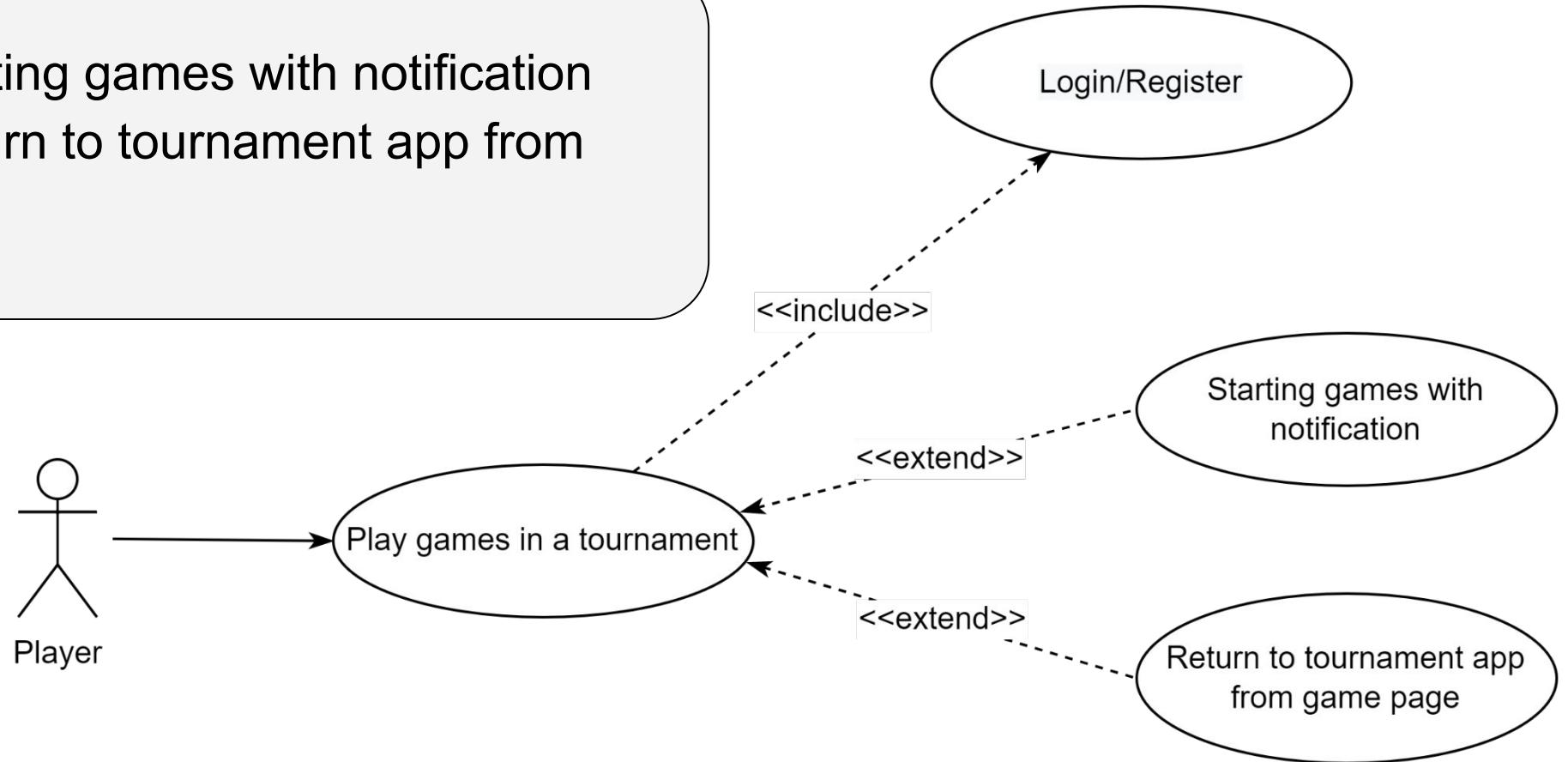




# Epic: Play Tournament games in CodeDefenders

CDF-36 Starting games with notification

CDF-38 Return to tournament app from game page



## CDF-36 Starting games with notification

Entry condition: the tournament is ready to start (all players joined and the matchmaking has been performed)

Players want to play games in a tournament and get notified when a game in which they are participating starts

Exit condition: homepage

Validation: After a tournament starts the involved players must see a notification of the game, a button to access the game on CodeDefenders and a button to access the streaming.

Motivation:

- Players must be able to play games on CodeDefenders and to receive notifications of their next games.

Source: Customer request (notifications requirement identified by the team)

- 1) The tournament application creates and starts the games on CodeDefenders
- 2) The players are notified by the tournament application that a game started
- 3) The clients receives from the tournament application links to play the game on CodeDefenders and watch the game live

## CDF-38 Return to tournament app from game page

Entry condition: Any Tournament App page that allows viewing a game on CodeDefenders

A user watches a game and wants to return to the Tournament App

Exit condition: The same Tournament App page

Validation: The final page corresponds exactly to the initial page

Motivation:

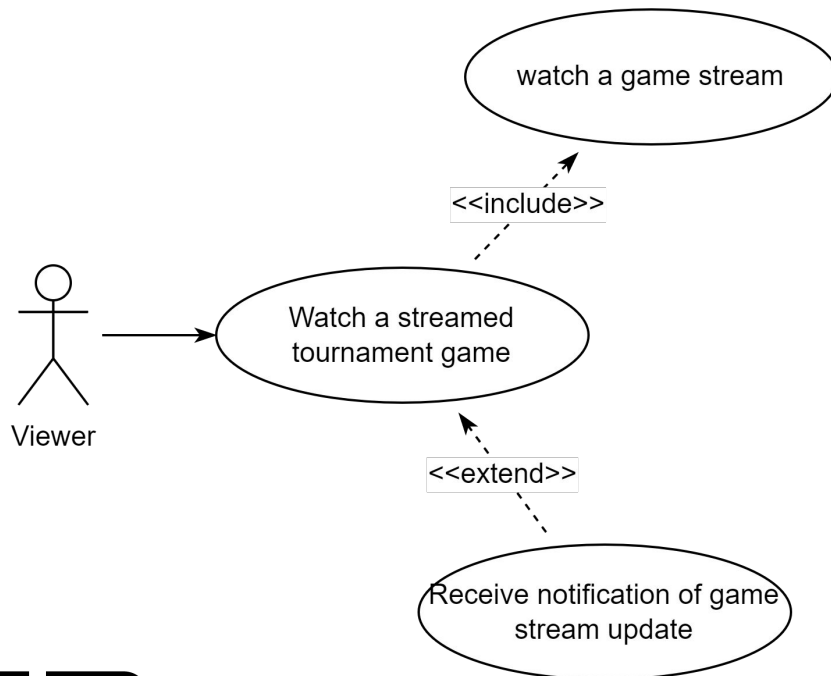
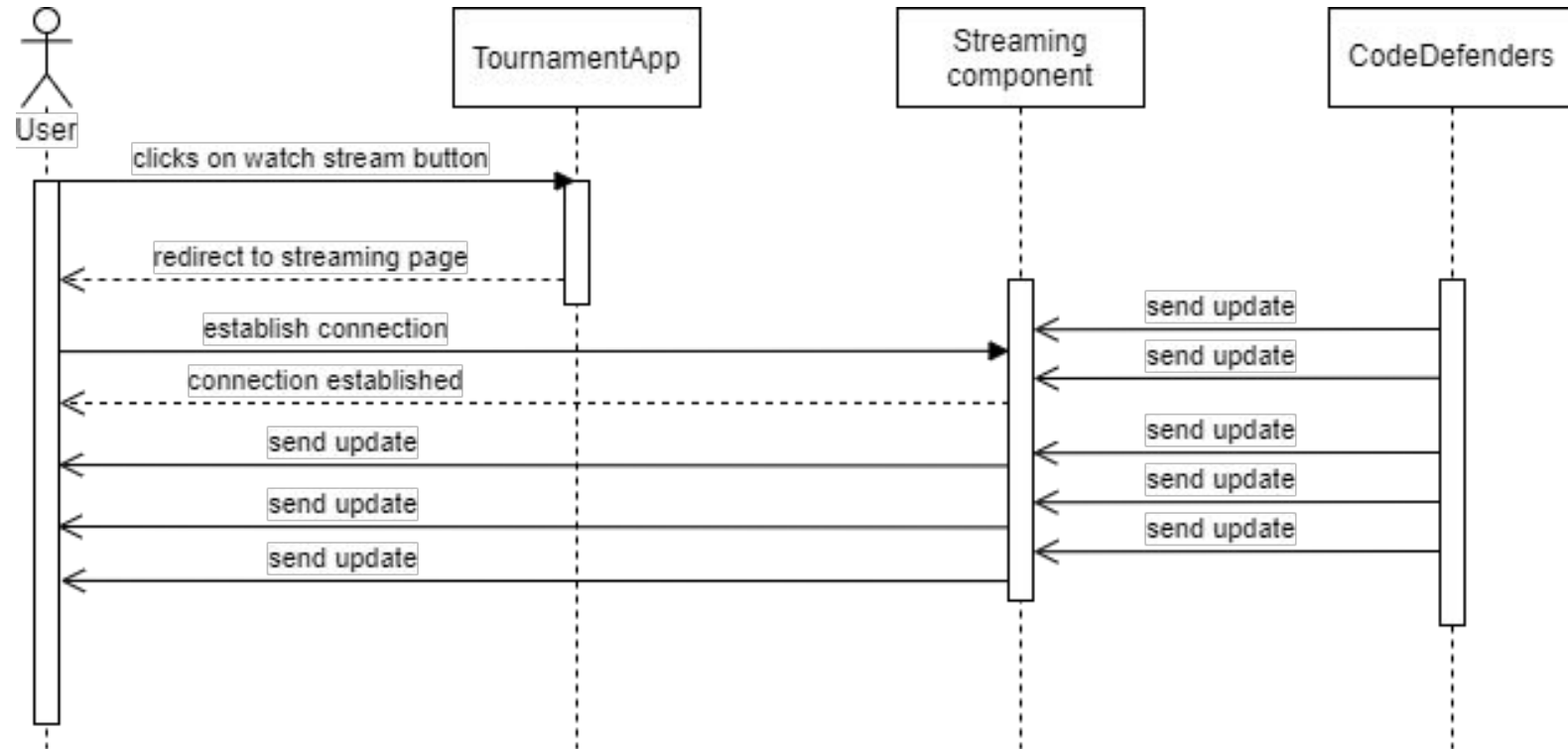
- Since the Tournament App and the CodeDefenders game page are on different servers, the user needs a convenient way to switch between them

Source: Identified by the team

- 1) From a Tournament App page a user decides to watch a game
- 2) The user is redirected to the CodeDefenders game page
- 3) The user clicks the button to return to the previous page
- 4) The user is redirected to the previous Tournament App page

# Epic: Watch a streamed tournament game

[CDF-39](#) view game stream  
[CDF-40](#) notification of game stream update



## CDF-39 View game stream

Entry condition: Home Page

A user wants to watch an ongoing live game

Exit condition: Game streaming page

Validation: A connection with the streaming component is established and the user starts receiving updates about the game selected.

Motivation:

- Guests and users must be allowed to follow the games of their preferred players live.

Source: Customer request

- 1) In the homepage the user clicks on the button corresponding to the live game he is interested to
- 2) A connection is established with the streaming component
- 3) The tournament application shows a dedicated page where updates from the game selected are shown

## CDF-40 Notification of game stream update

Entry condition: Game streaming page

A user has joined a live stream of an ongoing game and he wants to receive updates of that game.

Exit condition: The game ends

Validation: the user can see live updates of the game with visual effects.

Motivation:

- Guests and users must be allowed to receive live updates about the match they are interested in

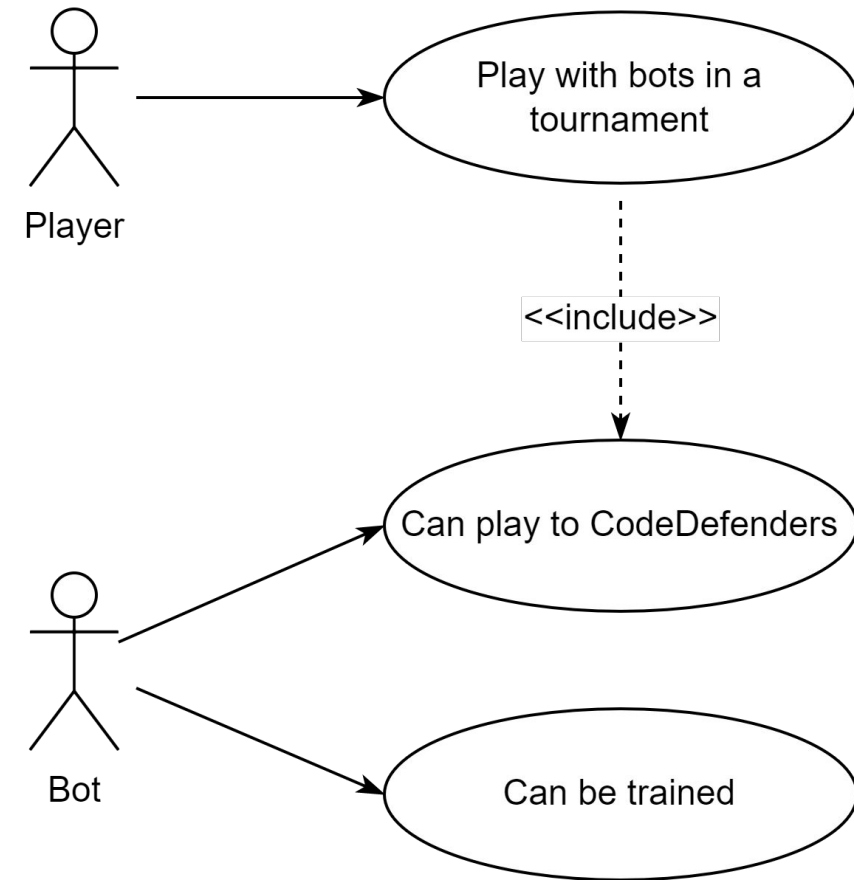
Source: Customer request

- 1) The user receives updates about the ongoing game
- 2) Updates are shown on the dedicated page with visual effects for points, scores and main actions.

# Epic: Play with Bots

CDF-43 Bots can play

CDF-44 Bots can be trained



## CDF-43 Bots can play

Entry condition: A fully developed bot is ready to play

Users wants to make their developed bots play CodeDefenders

Exit condition: The game ends

Validation: A dummy bot can complete an entire game in CodeDefenders

### Motivation:

- We need to allow users to have their programmed and trained bots and let them play on CodeDefenders

### Source: Customer request

- 1) An authenticated user obtains an identification token to verify its bot
- 2) The bot joins a CodeDefenders game
- 3) The bot plays the game by calling CodeDefenders API
- 4) The game ends and the bot stops playing



## CDF-44 Bots can be trained

Entry condition: A user obtained an identification token

Users wants to download historical data about previous games in order to train their bots.

Exit condition: All historical data are received

Validation: A dummy bot can complete an entire game in CodeDefenders

### Motivation:

- We need to allow users to access historical data so that they have data to train their bots as they prefer.

### Source: Customer request

- 1) An authenticated user obtains an identification token to verify its bot
- 2) The user requests the historical data to train the bot using the appropriate CodeDefenders API
- 3) All the historical data are sent to the user

# Epic: Avoid CodeDefenders overload

CDF-31 Low latency

CDF-69 Efficient flow of updates

CDF-31 and CDF-69 are **non-functional requirements**

## CDF-31 Low latency

Entry condition: Every page allowing a client to send requests to CodeDefenders server

CodeDefenders must support the dispatch of requests among multiple CodeDefenders instances.

Exit condition: user's request is served by one of the CodeDefenders instances

Validation: 5 different instances of CodeDefenders servers are running and receiving requests from the clients (easily scalable to n).

### Motivation:

- Players of CodeDefenders don't want to have overloaded servers and slow response of the web game.

### Source: Customer request

- 1) The user sends a request to CodeDefenders
- 2) The request is dispatched to the server instance with the minimum number of requests
- 3) The request is served and the response is returned to the user

## CDF-69 Efficient flow of updates

Entry condition: -

Updates about streamed games must flow from CodeDefenders to the streaming component and then be dispatched to the interested users.

Exit condition: the user receives updates about the game he is watching

Validation: Clients are able to see live updated games by just interacting with the streaming component (and without contacting CodeDefenders)

Motivation:




































- We want to avoid overloading CodeDefenders servers with continuous requests from users watching live games.

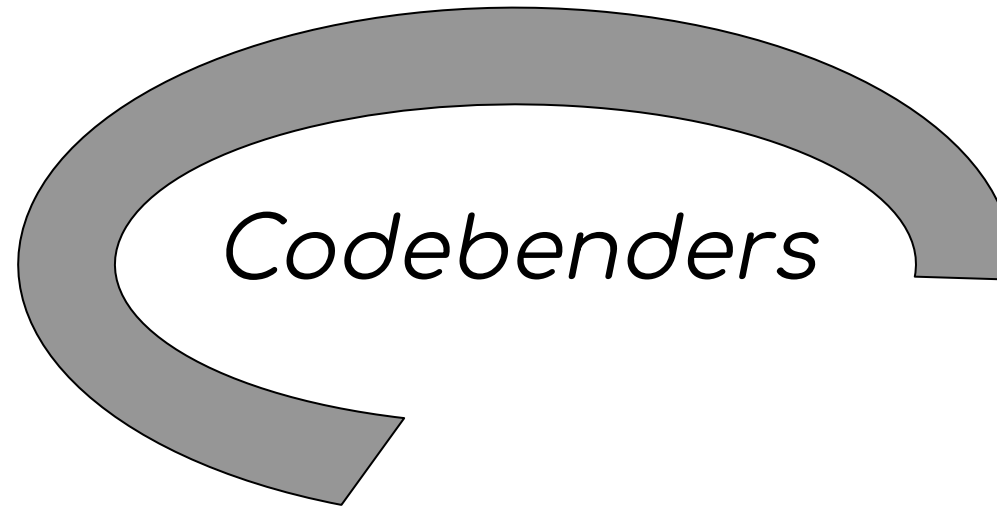
Source: Customer request

- 1) The user joins a live stream on the streaming component
- 2) CodeDefenders sends updates about the ongoing games to the streaming component
- 3) The user receives updates about the game he is watching by the streaming component

# Product Backlog

- User stories are sorted in priority order
- CDF-35, CDF-54 and CDF-32 have been added to sprint 2 backlog and are already in progress
- More information about subtasks, estimated work effort and work remaining is available on Jira

 CDF-35 Team creation	 IN PROGRESS ▾	
 CDF-54 Team management	 IN PROGRESS ▾	
 CDF-32 Login/Register	 IN PROGRESS ▾	
 CDF-37 Join team	TO DO ▾	
 CDF-33 Create Tournament	TO DO ▾	
 CDF-41 Display tournaments info	TO DO ▾	
 CDF-34 Join Tournament	TO DO ▾	
 CDF-36 Starting games with notification	TO DO ▾	
 CDF-38 Return to tournament app on game end	TO DO ▾	
 CDF-39 view game stream	TO DO ▾	
 CDF-40 notifications of game stream update	TO DO ▾	
 CDF-69 Efficient flow of updates	TO DO ▾	
 CDF-43 Bots can play	TO DO ▾	
 CDF-44 Bots can be trained	TO DO ▾	
 CDF-31 Low latency	TO DO ▾	
 CDF-42 Matchmaking	TO DO ▾	



## contact info:

[fanny.delnondedieu@fer.hr](mailto:fanny.delnondedieu@fer.hr)

[dominik.brdar@fer.hr](mailto:dominik.brdar@fer.hr)

[hrvoje.rom@fer.hr](mailto:hrvoje.rom@fer.hr)

[simone.mezzaro@mail.polimi.it](mailto:simone.mezzaro@mail.polimi.it)

[fabio.patella@mail.polimi.it](mailto:fabio.patella@mail.polimi.it)

[andrea2.restelli@mail.polimi.it](mailto:andrea2.restelli@mail.polimi.it)