RASD-v0.5

Francesco Spangaro - Tosetti Luca - Francesco Riccardi

October 2023

POLITECNICO
MILANO 1863

Contents

1 Introduction

1.1 Purpose

The purpose of the CodeKataBattle platform is to create a friendly and enjoyable environment for students to acquire new skills and improve the ones already owned in Software Development. This is done by allowing students to train and compete with each others by writing code in order to resolve problems. All of this under the supervision of educators who can challenge their students to take part to these competitions.

1.1.1 Goals

G1: ***Allow educators to create new tournaments:***

Educators have the possibility to create new tournaments. When creating a tournament, educators have the opportunity to create new badges. Badges have corresponding achievements, called "Rules", which are defined on badge creation. Badges are obtained by users on achievement completion. Obtained badges will be then displayed on the user's profile page.

G2: ***Allow educators to create new battles:***

Educators have the possibility to define new battles within tournament they crated or in tournaments they have been granted permission to do so. When creating a new battle educators have to set different parameters:

- upload project description;
- specify the programming language and build tool to utilize, including test cases and build automation scripts;
- set minimum and maximum number of students per group;
- set a registration deadline;
- set a final submission deadline;
- set additional configuration for scoring.

G3: ***Allow educators to administer different tournaments:***

Educators can grant other colleagues permission to create new battles in their tournaments. Educators have the possibility to close their tournaments, thus, not letting students submit new answers to any battle defined in the closed tournament, nor letting their colleagues create new battles in that tournament.

G4: ***Allow educators to administer different battles:***

Educators have the possibility, once a battle has expired, to manually evaluate through the platform each student's work, and then assign a corresponding score to each one of them, ranging from 0 to 100.

G5: ***Allow students to subscribe in tournaments:***

Students subscribed to the platform have the possibility to subscribe to different tournaments, in which they plan to participate in.

G6: ***Allow students to participate in battles:***

Students can join battles within a set deadline. They can do so by themselves, by inviting somebody else or by accepting someone's else invite.

G6.1: ***Allow students to form groups to participate with:***

Students have the possibility to send out invitations to other students, so that they can form a group to participate with. Groups need to follow the guidelines specified by the battle creator for it to be accepted.

G6.2: ***Allow students to submit their answers:***

When students have developed a solution to the battle, they can submit their answer to the platform. Groups are requested to send only one answer. Students can change their answer as they proceed, when uploading a new solution the older one is overwritten, since there can only be one answer for each battle.

G6.3: ***Allow students to see their scores:***

After each answer submission, a new score is assigned to the students. The score can be manually created by an educator or automatically assigned to the students by the platform. Students can see the scores obtained

G7: ***Let the students be notified on important events:***

When a new tournament is created, all students subscribed to the platform are notified. A different notification will be sent when a new battle is created in a tournament they are subscribed to.

1.2 Scope

CodeKataBattle (CKB) is an easy-to-use platform, which aims to allow educators to propose homework and/or lessons in a new and fresh way to involve students more in acquiring and improving software developing skills. To do that, CKB offers educators the possibility to open several tournaments. Each tournament is composed by several battles in which students can compete with each other, individually or in groups. In order to offer all of this CKB relies

on the external platform GitHub. GitHub will take the role of a "bridge" between CKB platform and students, allowing them to upload their solutions on it. These solutions will be then taken by the CKB platform from GitHub and used to evaluate student's score in the battle for which they uploaded a specific solution.

1.2.1 Phenomena

Events that take place either in the real world, in machine world or in both. Used to describe respectively what cannot be observed by the machine, real world and event that connects the two.

1.2.1.1 World phenomena

Phenomena events that take place in the real world and are not observable by the machine

WP1: Students fork the GitHub repository of which they received a link by the platform.

WP2: Student write code on his personal device.

WP3: Students choose which tournament to join

WP4: Students choose which battle to join in a tournament precedently joined.

WP5: Student choose his teammates for a battle.

WP6: Educator choose whether and which colleagues to allow access to one of his tournaments

WP7: Student subscribed to a battle wait for its start (registration deadline expiration)

WP8: Educator decide to close a tournament

1.2.1.2 Shared phenomena

- Phenomena controlled by the world and observed by the machine

  ➤ Student related phenomena

  SP1: Student registration to the platform

  SP2: Student log in to the platform

  SP3: Student subscribe to a tournament within a deadline

  SP4: Student invite other students to form a team

  SP5: Student accept an invite from another student and join its group.

  SP6: Student or a group of students join a battle in a tournament they are subscribed to within a deadline.

SP7: Student upload a new software solution for the battle's problem, in which he's partecipating

SP8: Student sees its, and others badges visualizing its or others profile page

SP9: Student or a group of it, push a new commit on GitHub repository

➤ Educator related phenomena

SP10: Educator create a new tournament

SP11: Educator grant access to his other collegues to create new battle within a tournament he created

SP12: Educator create a new battle

SP13: Educator set battle's setting while creating one of them

SP14: Educator manually evaluate the work done by students in a certain battle of a certain tournament during battle's consolidation phase

SP15: Educator closes a tournament

SP16: Educator defines new badges achievable in a tournament by students while creating it

SP17: Educator sees collected badges of a student by visualizing its profile page

- Phenomena controlled by the machine and observed by the World.

➤ Student related phenomena

SP18: Student registered to the platform gets notified when a new tournament is created

SP19: Student subscribed to a tournament gets notified of upcoming battle created in that tournament

SP20: Student receive from the platform a invite notification in order to join a group of students to join a battle.

SP21: Platform, when a battle's registration deadline expires, send every student that joined the battle a link to the GitHub repository created by the platform itself.

SP22: Platform at the end of each battle updates students' score in the tournament in which battle took place allowing all students and educators to see them

SP23: Students get notified when a tournament is closed

➤ Educator related phenomena

SP24: Educator receive a notification when allowed by a collegue to access its collegue's tournament.

SP25: Educator gets notified when submission deadline of solution for a battle expires, and start the consolidation phase.

1.2.1.3 Machine phenomena

Phenomena events that take place in the machine world and are not observable from the real world

MP1: The platform creates a GitHub repository containing the code kata when registration deadline of a battle expires.

MP2: The platform when notified by GitHub API pulls the latest sources of the repository of a battle

MP3: The platform analyzes the sources, by running tests on them.

MP4: The platform calculate the battle score of a team based on the executables uploaded by students for a battle. Score is automatically updated when the platform receive notification from GitHub about new push actions.

MP5: The platform at the end of each battle of a tournament, compute the tournament rank of each student in that tournament.

MP6: The platform automatically register badges acquirement from a student, when that student satisfy the rules to obtain them.

1.3 Definitions, acronyms, abbreviations

1.3.1 Definitions

Term	Definition
GitHub Repository	→ A place on the GitHub platform where a user can store code, files and each file's revision history.
Registration deadline	→ Maximum time within which a student can subscribe to a battle or to a tournament.
Submission deadline	→ Maximum time within which a student, or a group of student, can upload their solution to a battle problem

Term	Definition
Code Kata	→ The word kata refers to a karate exercise in which a form gets repeated many time, making little improvements in each tentative. In this context it's used to express the fact that the code need to be developed multiple times to reach a optimal solution to the battle problem.
Consolidation phase	→ Phase started at the end of a battle, used to consolidate the score of each student in the battle by eventually a manual evaluation of the students' code by an Educator.
View-only mode	→ An abstract modality (not an implementation detail) in which educator can be if he access to a tournament in which they have no rights to create new battles
Modify-enabled mode	→ An abstract modality (not an implementation detail) in which educator can be if he access to a tournament in which he go granted the access by tournament's creator, and thus can create new battles.

1.3.2 Acronyms

Acronym	Meaning
API	→ Application Programming Interface, indicates on demand procedure which supply a specific task
CKB	→ CodeKataBattle, the name of the platform described in this document
IT	→ Used as acronym for Information Technology which identify something, generally a computing or communication hardware, with information storage capability and closely related to the informatic world.

Acronym	Meaning
UML	→ Unified Modeling Language, a standard notation for modeling real world object, in an high level diagram representing OO components.
BPMN	→ Stands for Business Process Modeling Notation, standard notation to represent processes through diagrams.
OO	→ Object-Oriented is a programming paradigm based on the concept of object which can contain data and code, and represent usually real world object.

1.3.3 Abbreviations

Abbreviation	Meaning
G#	→ Goal number #
WP#	→ World phenomena number #
SP#	→ Shared phenomena number #
MP#	→ Machine phenomena number #
D#	→ Domain assumption number #
c.s.	→ Computer science
e.g.	→ Exempli gratia, latin phrase corresponding to "for example"

1.4 Revision history

- **Placeholder data**: version 1.0
- **Placeholder data**: version 1.1
- **Placeholder data**: version 1.2
- **Placeholder data**: version 2.0
- **Placeholder data**: version 2.1

1.5 Reference documents

GitHub references:

- Official documentation to get started with GitHub: → `https://docs.github.com/en/get-started/quickstart`
- Official documentation about fork process → `https://docs.github.com/en/get-started/quickstart/fork-a-repo`
- Official documentation about GitHub actions → `https://docs.github.com/en/actions`

UML official specification → `https://www.omg.org/spec/UML`

BPMN official specification → `https://www.omg.org/spec/BPMN/2.0`

1.6 Document structure

- ***Section 1: Introduction***
  This section introduces the problem and the platform/application to be developed in order to resolve it. It describes the major purpose of the project, every one of his goals, the analysis of the its domain and every world, shared and machine phenomena associated with it. In addition in this section are inserted the definitions, acronysm and abbreviations used in this ddocument, including even it's revision history and refereced documents or web pages.
- ***Section 2: Overall description***
  This section gives an overall description of the project and all the interactions that could occur between the platform and the final users (Students and educators). To do this, it will include different possibile scenarios that could happen, the different actors involved in the platform usage and all the assumptions, dependencies and eventual constraints that have to be considered in the development of the platform
- ***Section 3: Specific Requirements***
  This section is the most technical one, it contains more precise descriptions of each scenario called use cases. It describe the several functional and performance requirements of the project and their map to the goals of the project. Finally it contains also all the design constraint and system attributes that must be followed/guaranteeded while developing the platform.
- ***Section 4: Formal analysis using allo***y
  In this setion can be found a formal description of the platform/application. The formal description is done using the formal language Alloy (referenced in section 1)

- ***Section 5: Effor spent***
  A simple section in which are included all the information about the time requested by each group member to complete this document, and it's division by each section of the document.

2 Overall description

2.1 Product perspective

The following section contains the UML diagram of the platform and a list of meaningful scenarios in which the platform can be used, how and when users can interact with it.

2.1.1 Scenarios

2.1.1.1 Student sign up to the platform

Peter is a volenterous student of an informatic class, that want to improve his software development skills. He comes to know about the platform CKB one day, when his c.s. professor propose to his class, a software development tournament in substitution to the normal, boring and limited tests organized through the academic year. Peter is extremely interested, not only in the tournament, but also in the service offered by the platform itself, thus the same day he decide to try to register to it. Peter open his personal browser and go to the CKB site's homepage. Here he goes to the student dedicated page and clicks the "Sign up" button in order to register. He inserts all the required information in the mandatory fields of the page that showed up (e.g. name, surname, attended school, email, nickname, password, ...) and clicks on "Confirm" button. Peter now wait for the notification, through his email, about the correctness of his registration, and can now access all the CBK platform's features after loggin in with it's credentials.

2.1.1.2 Educator sign up to the platform

Vittorio is an innovative teacher who teaches c.s. in a school. He discovered, while searching the web for new ideas on how testing his students' abilities in software developing, the CKB platform and its services. Vittorio decide to try to register to the platform, as he's very intrigued about it. To do that Vittorio goes on the CKB Homepage and then on the page dedicated to educators and clicks the "Sign up" button. At this point he compiles all the fields in the form that showed up with the new page (e.g. name, surname, school in which he teaches, istitutional email, password, ...), especially the fields related to his profession, and in the end clicks on "Confirm" button. Finally Vittorio wait for the registration confirm email, and start using CKB platform's features after loggin in with it's credentials.

2.1.1.3 Educator creates a new tournament (and badges)

Laura is an c.s. educator who registered to CKB platform. She decide to create a new tournament to let her student compete with each other and improve their software developing skills. In order to do so, Laura log in her account on the platform and try to create a new tournament. While doing so, she decide which programming languages can be used to develop the solution to the battles that will be contained in the tournament, the name of the tournament, the method through which students can access it, eventually its maximum duration, and finally she decide if the tournament will contains some new or default badges, obtainable by the students by doing some achievements. Laura wants to create new particular badges, in order to encourage her students to participate more actively to the tournament. To do so, Laura access the appopriate section during tournament creations, and starts creating the badges, specifing their title, rules to obtain them, their icon and eventually their score, that will be added to the student's score that obtained them. At the end of the process Laura, confirm her choices and the tournament starts.

2.1.1.4 Educator creates a new battle

Laura is an c.s educator who registered to the CKB platform. After she created a new tournament, Laura wants to immediately create a new battle within it, to let her students compete with each other. To do so Laura, log in her account on the platform, access the tournament in which she wants to create a new battle through an appropriate web page. At this point Laura try to create a new battle within the tournament. While doing so, she decide the battle's name, the programming language allowed in the battle, maximum number of partecipant, dimension of group of students that can partecipate to it, a registration and submission deadline and eventually sets some personalized rules to calculate students' score in the battle. At the end of all of this process, Laura waits for the students of her class, subscribed to the tournament in which the battle has been created, to join the battle. At the end of the registration deadline, partecipant can start competing with each other, while Laura can supervise their work. At the end of the battle, which automaticcaly close itself at submission deadline, students' score gets calculated and added to their tournament score.

2.1.1.5 Educator close a tournament

Marco is an c.s educator who registered to the CKB platform. He has created a new tournament one mounth ago, today, after this mounth he decide to close it. In order to do so, Marco log in his account on the platform and access the tournament which he wants to close through an appropriate web page, at this point Marco control il there are battles that aren't altready closed and sees that Laura has a tournament that is still open, so he now control when is the submission deadline of this battle. After discovering that the submission deadline is in two days he close the possibility of other educators to creates

new battles. After three days Marco, after log in to the platform, control that there are'n still open tournament or tournament that are waiting for the manual evaluation. Then he proceed to close the tournament

2.1.1.6 Educator manually evaluate a battle after the end

Luca is an c.s educator who registered to the CKB platform. He has created a new battle in a tournament two weeks ago, choosing as the submission deadline yesterday and choosing to include a Manual evaluation in the total evaluation of the battle, according to the requirments of shared balaced partecipation, optimization and writing comments related to the code and the implementative choice. Today, Luca log in his account on the platform, access to the tournament in which he created the battle and see that this battle in closed and is waiting for a manual evaluation and decide to examine the codes that have been delivered in the battle by the different groups. The first group he analyzes is made un of three students, and sees that they have a score of 90/100 in the automated evaluation. After entering on the resource made available by the group, Luca check if the number of commits and work ha ìs been done equally among the students in the group and not noticing large differences between the number of lines of code written and the number of commit made by each student he decide to go next to the comment section. Opening the code in the resource, he relizes that the group has limited himself to adding simple comments, without going to much in the details of the work done and choices made; so he decide to modify the grade assigned befor, by reducing it from 90/100 to 80/100, writing a little explanation of the reason for this change. It then continues to control all the others group and at the end he can close formally the battle.

2.1.1.7 Student forms a group then joins a battle

Francesca singns in the platform with her credentials and sees that a new tournament has opened. Shoe would like to participate in it, and so decides to form a group with two of her friends to compete in the tournament's battles. She clicks on the apposite menu and chooses the users she would like to invite to join her group. After both her friends accept her invite, she enters the tournament by signing up her group as participant. She then looks at all of the proposed battles they can join in, and chooses one. Once the registration deadline for the groups expires, Francesca's group forks the automatically created GitHub repository and sets up an automated workflow through GitHub Actions, to notify the platform of each of their commits.

2.1.1.8 Student uploads a solution

Piero is currently competing with his group in a tournament, they are working on a solution for a battle and they are confident that they are right with what they coded. Piero decides to upload a solution for the battle to the GitHub repository, so that it can be evaluated and they can see how much their score

is for this first draft solution. Piero uploads the current solution to the forked GitHub repository. After the push, the GitHub Actions workflow is started, letting the platform know that Piero has uploaded a new solution for the battle he's currently competing in. The automated evaluation system integrated in the platform now starts, tests are automatically ran and no human evaluation is deemed needed, so the platform gives Piero's solution a score of 75/100. Piero sees the score on his profile on the platform and understands that his solution is found correct, but not the best that can possibly be done. He decides to continue working on it, until his solution will obtain a score of at least 95/100.

2.1.1.9 Student uploads a solution after deadline expiration time

Frank has been competing in a tournament for very long, and has finally found a solution for the last battle he struggled to solve. After having uploaded his solution on the forked repository, the GitHub Actions workflow starts and notifies the platform that Frank's has uploaded a new solution for his final battle. The CodeKataBattle platform then sees that the deadline for Frank's battle has expired. The platform notifies Frank that his last uploaded solution will not be considered on the final score he will get, and no further action is taken.

2.1.1.10 Students visualize his tournament's results and his badges

Paolo, Lucia e Alessandro are three friends that are subscribed in the CKB platform, one evening they decide to compare their tournament's result on the CKB platform. First, they want to compare their result in the last tournament, which ended the day before. Paolo, after logging in the site and entering his profile page, sees that he achived a score of 70/100. Lucia in last tournament was in the same group as Paolo, so she achived the same score. Alessandro, who was in a different group, sees that he has a score of 90/100, thus becoming the best of the three. Lucia now wants to see the best score they have ever achieved, so after logging in the site and searching for her best result, found out that she participated in a tournament two months prior, where she achived a score of 95/100. Alessandro wants to see his best result and after logging in the site, he sees that in his badges there is the "perfect score" badge, which is assigned to a student who got a score of 100/100 in at least one tournament. Paolo then wants to know who out of the three has the most badges. After accessing his profile page, he sees that he has obtained a total of 57 badges, obtained in all of the tournaments in which he participated; in this badges Paolo sees that have achived badges for being the most committer of a tournament, one for having partecipated at 50 tournaments and one for having write the most number of code row in a tournament. Alessandro checks that too, and sees that he obtained only 13 badges.

2.1.2 Class diagram

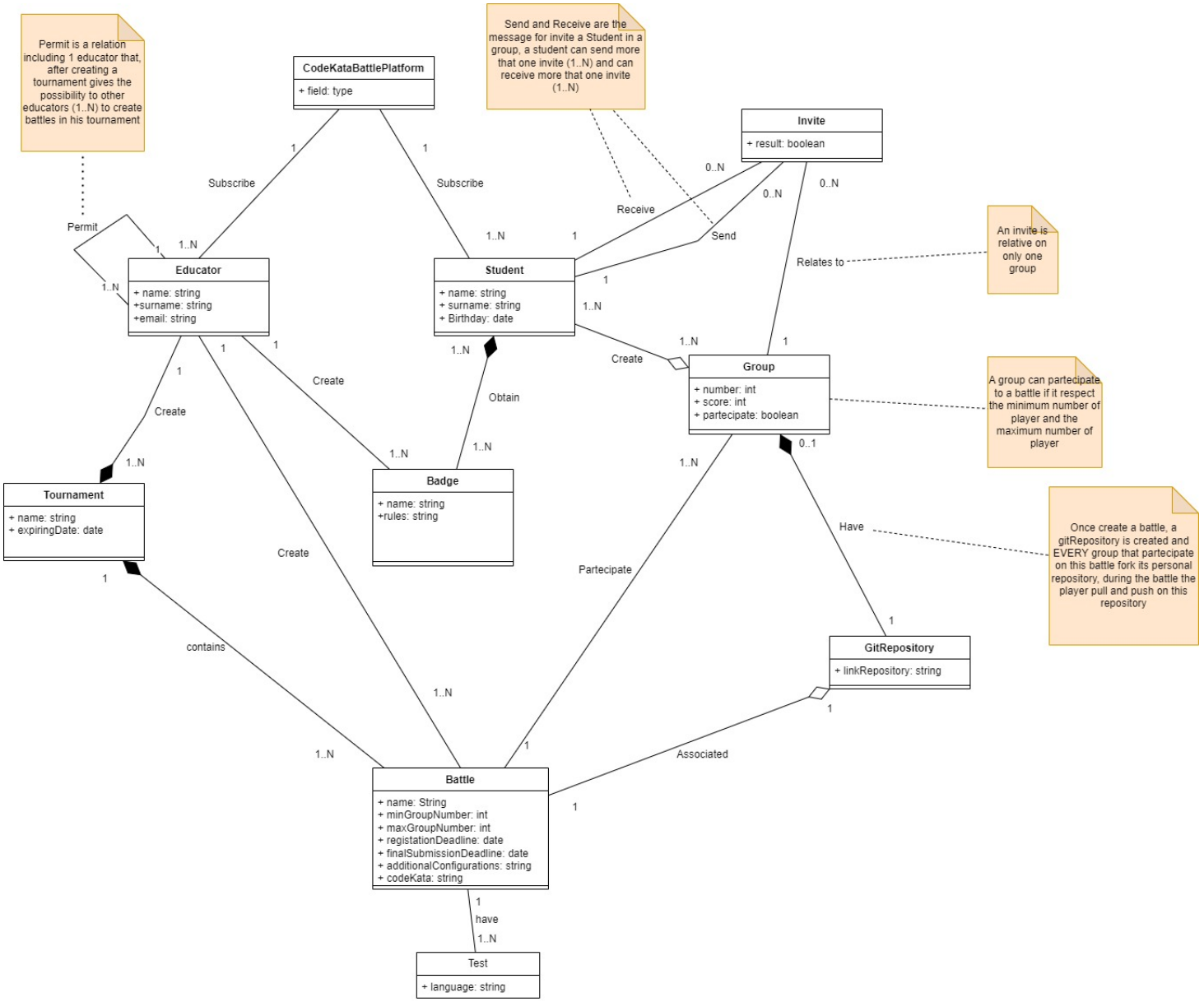

Figure 1: Class Diagram

2.2 Product functions

2.2.1 Shared functions

- **Sign-up:** Let the user (either students or educators) sign-up to the platform.

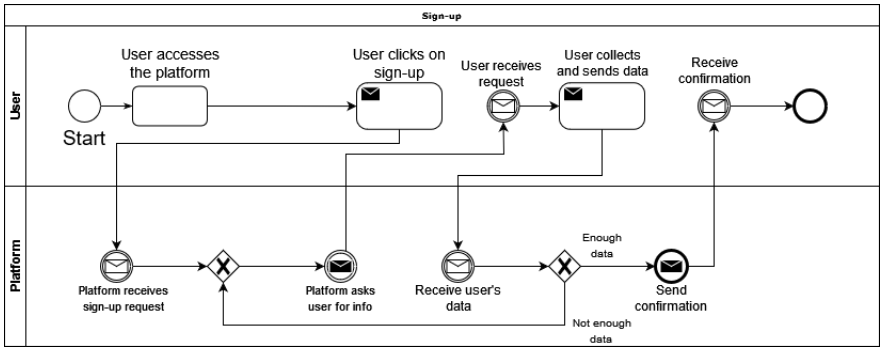

Figure 2: Sign-up BPMN

- **Visualize student's profile:** Let an user (either a student or an educator) to visualize the profile page of a specific student.

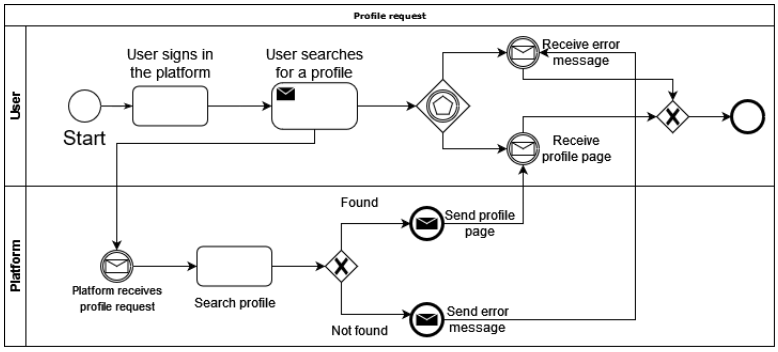

Figure 3: Profile visualization BPMN

2.2.2 Student functions

- **Student subscribes to a tournament:** Let a student search, according to some parameters (most used programming languages, creation date, number of partecipant, etc...) and subscribe to a tournament, after loggin in the platform.

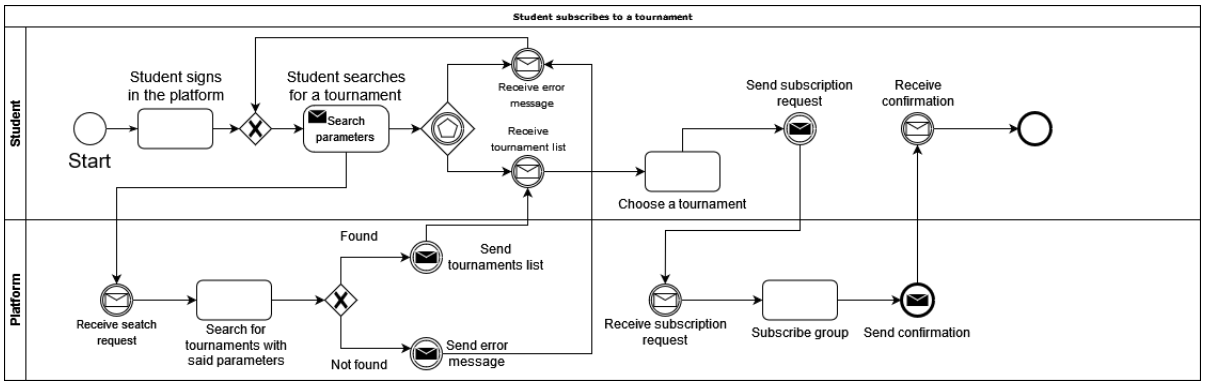

Figure 4: Student's tournament subscription BPMN

- **Student join a battle:** Let a student subscribed to a tournament search a battle in it according to some parameters (programming language requested, expiration date, difficulty, etc...) and join it alone or in a group.

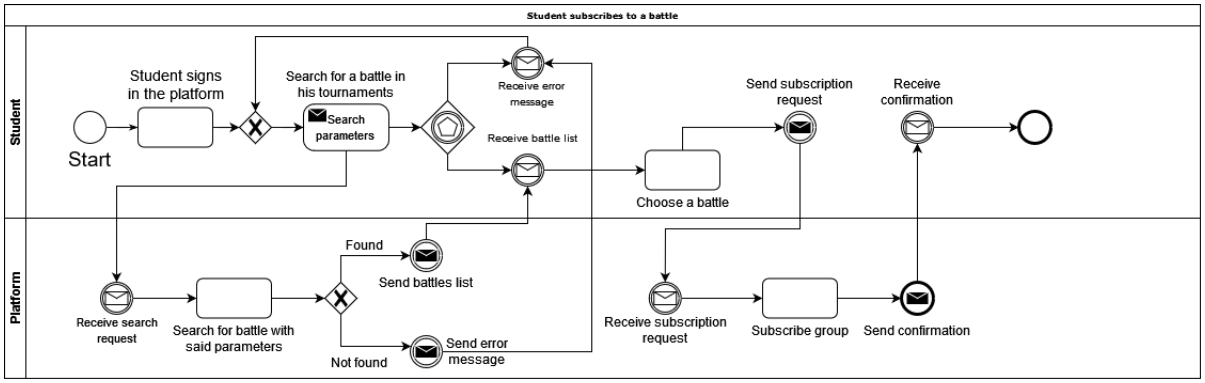

Figure 5: Student's battle joining BPMN

- **Student gets notified of new events:** Notify the student about new tournaments created, if signed to the platform, and new upcoming battles that take place in tournaments in which he has joined.

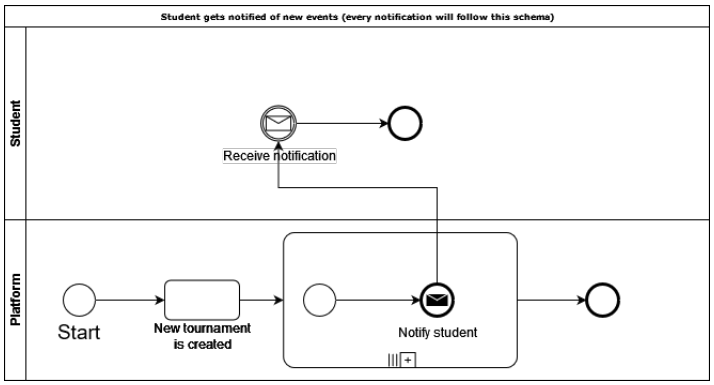

Figure 6: Student notification BPMN

- **Student's solutions get evaluated:** Let a student, who connects to his forked repository on GitHub, upload his solution and evaluate it according to some parameters.

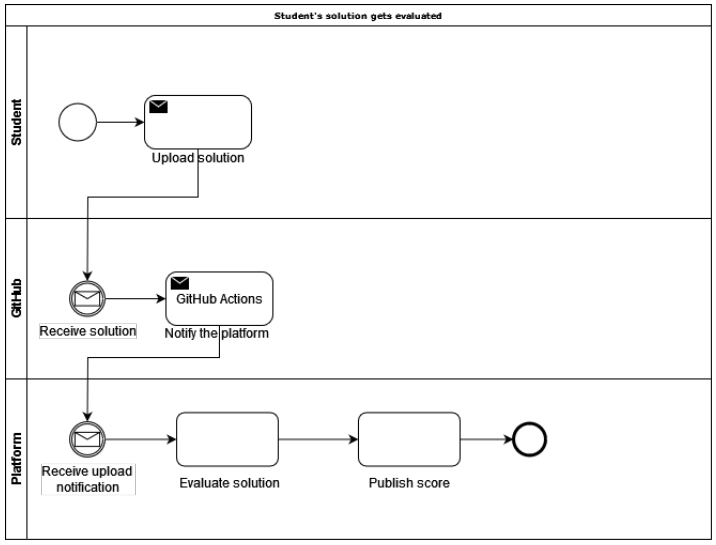

Figure 7: Student's solutions evaluation BPMN

- **Student form a group for a battle:** Let a student form a group with other students in order to face a battle, through invite messages.

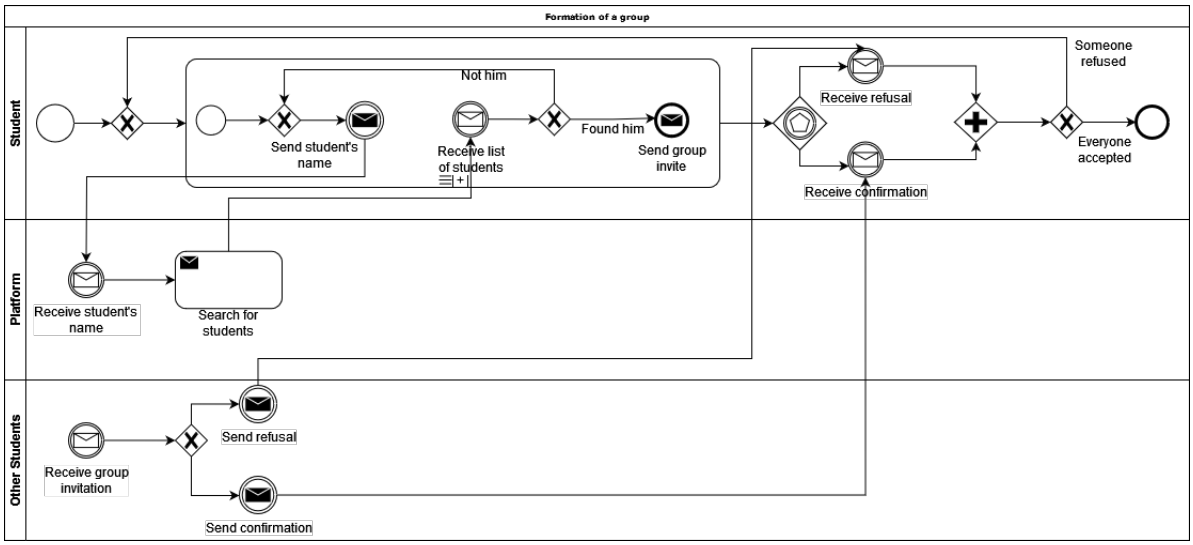

Figure 8: Formation of a group BPMN

2.2.3 Educator functions

- **Educator create a tournament:** Let an educator create a new tournament and set its parameters.

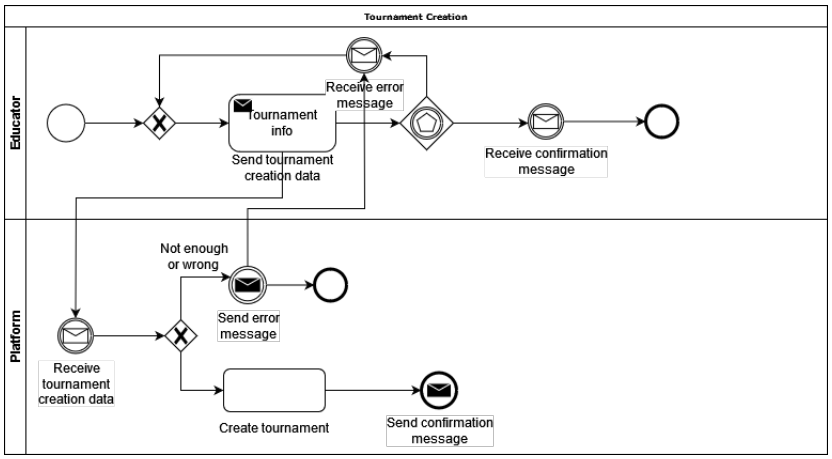

Figure 9: Tournament's creation BPMN

- **Educator close a tournament:** Let an educator close one of his tournaments whenever he wants.

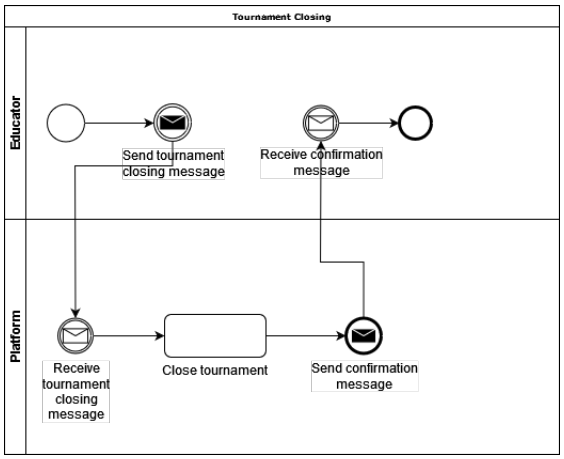

Figure 10: Tournament's closing BPMN

- **Educator grant access to a tournament:** Give an educator the possibility to grant the access to one of his tournaments to a colleague.

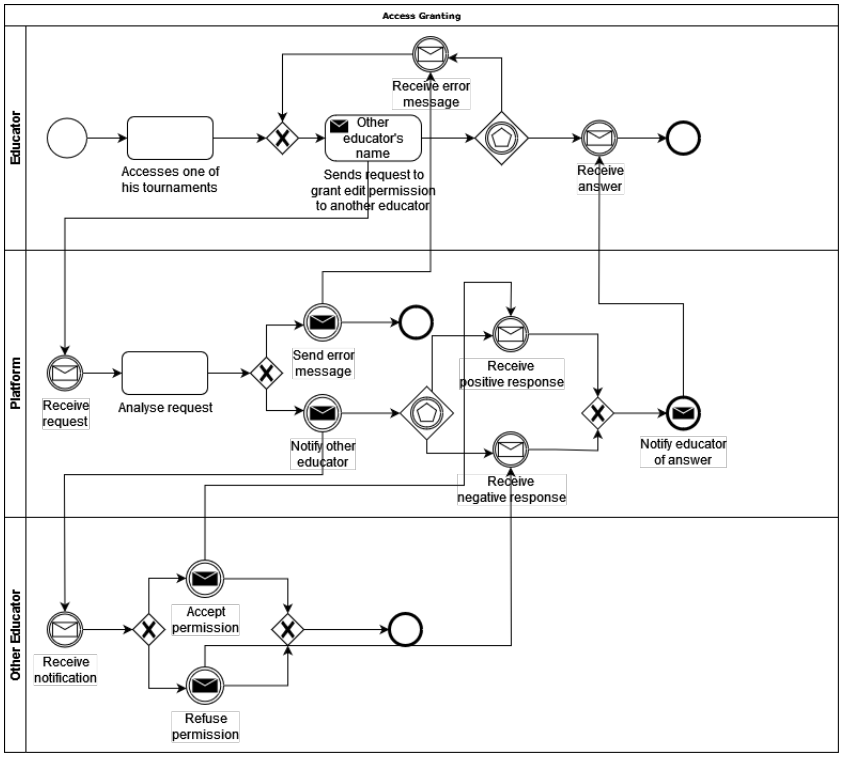

Figure 11: Grant access to tournament BPMN

- **Educator creates a battle:** Grant an educator the possibility of creating a battle in tournaments where he has permission to.

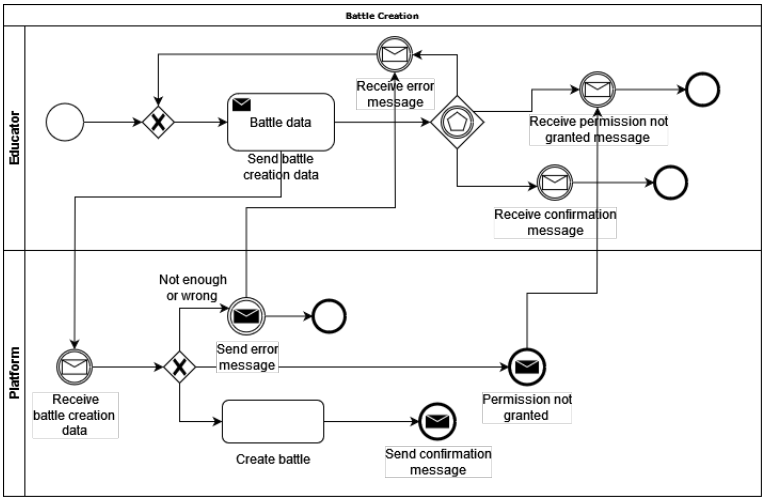

Figure 12: Create battle BPMN

- **Educator notified of battle's end:** Educator gets notified about the end of a battle in one of his tournaments, give him the possibility to manually evaluate the students' solutions.

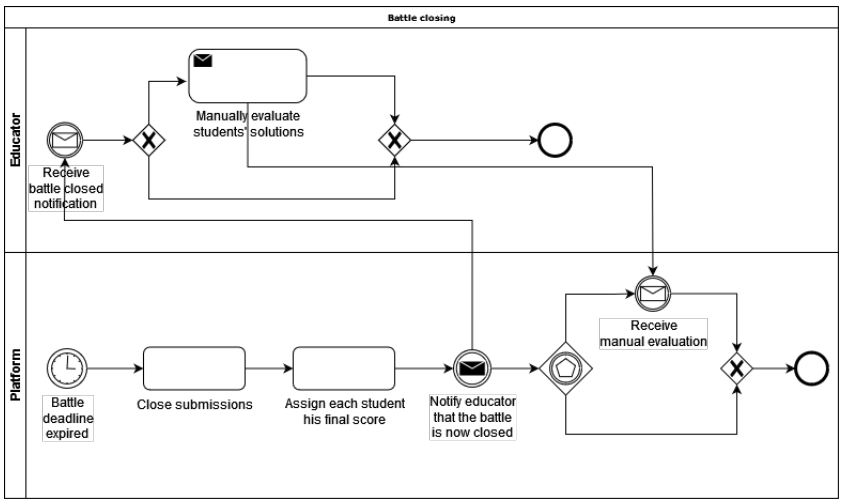

Figure 13: Ending of a battle BPMN

2.3 User characteristics

2.4 Assumptions, dependencies and constraints

D1: Students and educators have access to internet while using the platform

D2: Students and educators have their own IT device to connect to the application

D3: Students and educators have to be subscribed to the platform in order to use its features

D4: Students know how to use GitHub actions

D5: Students know how to fork a repository in GitHub

D6: A student can join a battle only if subscribed to the tournament in which that battle take place

D7: GitHub platform offers reliable services through its API allowing to CKB platform to always get notified when new code is uploaded by students.

D8: Educator know how to create new badges, and new rules to obtain them, for the tournaments

D9: Time information about registration and submission deadlines for tournaments and battles are always correct.

D10: Code written by students can not make the platform crash while testing it.

D11: Educators upload ,when creating a battle, some correct, meaningful and faultproof test cases and automation scripts.

D12: Students score is always correctly calculated and meaningful.

D13: Educators access always access a tournament either in view-only mode, if not invited by the tournament's creator, or in modify-enabled mode if the tournament's creator has granted him the access. **(??? Verificare se potrebbe essere un goal piuttosto che una assumption ???)**

3 Specific requirements

3.1 External interface requirements

3.2 Functional requirements

3.2.1 Requirements

3.2.2 Use case diagrams

- **Student**

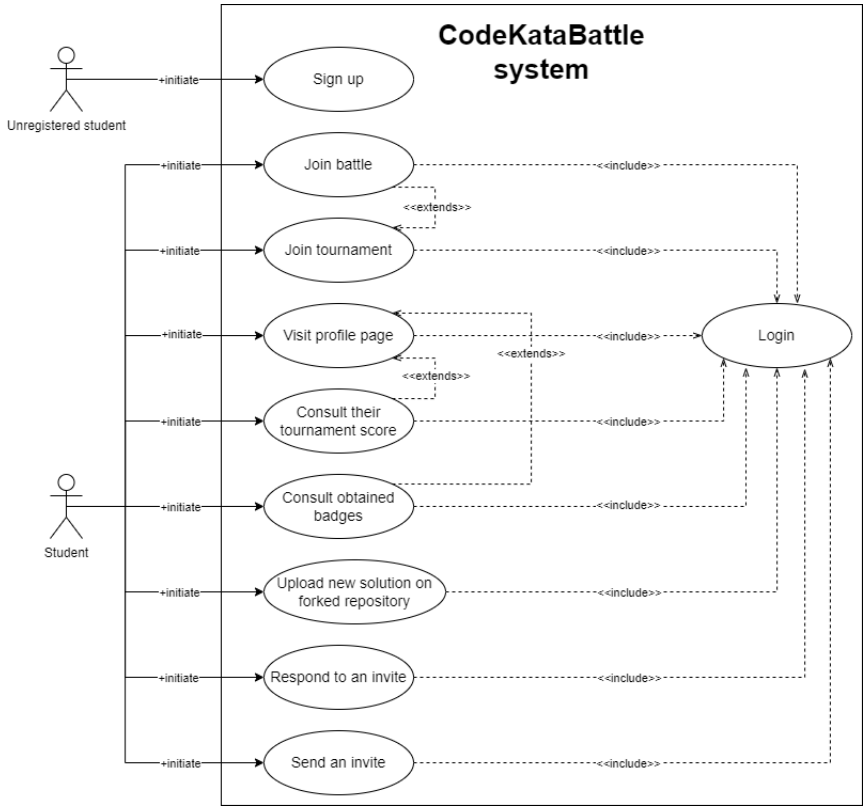

Figure 14: Student use case diagram

N.B: The repository citated in this diagram, on which the student upload their solution, it's not part of the platform. The system, in fact, relies on a third party service (GitHub) to handle the repository. This use case

was specified in order to better clarify the tight interaction between the system and the student trought the GitHub service.

- **Educator**

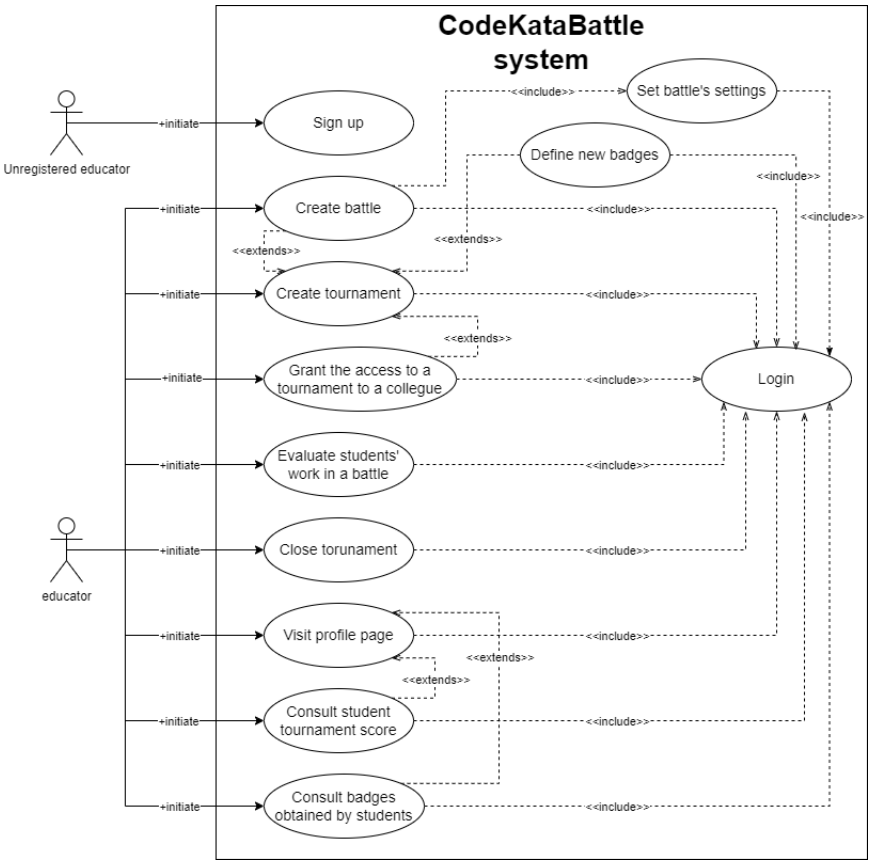

Figure 15: Educator use case diagram

- **GitHub**

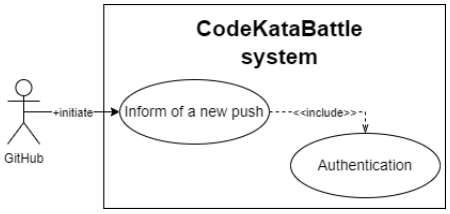

Figure 16: GitHub use case diagram

3.2.3 Use cases w/ sequence diagrams

1. Student sign up to the platform

Name	Student sign up
ID	UC.1
Actors	Unregistered student
Entry condition	Student want to register to the platform
Flow of events	1. Student opens CKB platform 2. Student presses the sign-up button 3. Student fills the form with all the required informations (name, surname, nickname attended school, email, password, ...), accepts the "Terms & Conditions" and finally clicks on a button to confirm. 4. CKB platform validates the personal information inserted by the student. 5. CBK platform display a confirmation message. 6. CKB platform send an email notification to the student regarding the registration outcome
Exit condition	Student's account is created, and its data are saved into the system
Exceptions	4.1 CKB platform isn't able to validate student registration data (mail already taken, attended school non existings, ...) → Unregistered student gets notified of the registration failure throught an error message

Figure 17: Student sign up sequence diagram

2. Educator sign up to the platform

Name	Educator sign up
ID	UC.2
Actors	Unregistered educator
Entry condition	Educator want to register to the platform
Flow of events	1. Educator opens CKB platform 2. Educator presses the sign-up button 3. Educator fills the form with all the required informations (name, surname, nickname school in which teaches, istitutional email, password, ...), accepts the "Terms & Conditions" and finally clicks on a button to confirm. 4. CKB platform validates the personal information inserted by the student. 5. CBK platform display a confirmation message. 6. CKB platform send an email notification to the educator regarding the registration outcome
Exit condition	Student's account is created, and its data are saved into the system
Exceptions	4.1 CKB platform isn't able to validate educator registration data (mail already taken, school's details not correct, ...) → Unregistered educator gets notified of the registration failure throught an error message

Figure 18: Educator sign up sequence diagram

3. Educator creates a new tournament

Name	
ID	
Actors	
Entry condition	
Flow of events	1. 2. 3. 4. 5. 6.
Exit condition	
Exceptions	N.N $\rightarrow$

Figure 19: ...

4. Educator creates new badges

Name	
ID	
Actors	
Entry condition	

Flow of events	1. 2. 3. 4. 5. 6.
Exit condition	
Exceptions	N.N →

Figure 20: ...

5. Educator creates a new battle

Name	
ID	
Actors	
Entry condition	
Flow of events	1. 2. 3. 4. 5. 6.
Exit condition	
Exceptions	N.N →

Figure 21: ...

3.2.4 Traceability matrix

3.3 Performance requirements

3.4 Design constraints

3.5 Software system attributes

4 Formal analysis using Alloy

5 Effort spent

6 References