POLITECNICO

MILANO 1863

Department of Electronics, Information and Bioengineering
Doctoral Programme In Information Technology

Software Engineering 2 Requirements Analysis and Specification Document

Author(s): **Filippo Balzarini -**

Christian Biffi -

Michele Cavicchioli -

Download Page: https://github.com/filomba01/BalzariniBiffiCavicchioli

Contents

1 | Introduction

1.1. Purpose

The CodeKata is a learning method that takes inspiration from the Kata techniques and is based on continuous practice which became very popular in those years.

CodeKataBattle delineates an innovative platform geared towards enhancing students' software development skills through collaborative learning using CodeKata's fundamentals. Facilitated by educators, CKB provides a dynamic environment where students engage in code kata battles, refining their programming proficiency and embracing best practices such as the test-driven development approach.

Similar to recent initiatives addressing global challenges, CKB empowers educators to orchestrate challenges within competition, fostering healthy competition and cultivating an environment for skill enhancement. The platform enables educators to define battle parameters, set deadlines, and configure scoring criteria, fostering a tailored and effective learning experience.

At its core, a code kata battle presents students with programming challenges within specific language frameworks, coupled with exhaustive test cases. Teams collaboratively tackle these exercises, adhering to a test-first methodology and submitting solutions to the platform upon battle completion.

CKB's automated evaluation system ensures an impartial assessment of student submissions. Automated scrutiny covers mandatory factors, including functional aspects, timeliness, and source code quality, offering an unbiased representation of team performance. Educators can further enhance evaluations with optional manual assessments, providing nuanced insights into student work.

1.1.1.   Goals

The CKB platform aims to provide a collaborative environment for students to practice and refine their software development skills. The platform enables educators to orchestrate challenges within competitions, fostering healthy competition and cultivating an environment for skill enhancement. The platform enables educators to define battle parameters, set deadlines, and configure scoring criteria, fostering a tailored and effective learning experience.

The platform will be used by two types of users: Educators (ED) and Students (ST). The ED will be able to create competitions and battles within competitions. The ST will be able to create teams and join battles as a team or individually. The platform will provide a dashboard for code submission and automated evaluation of the code submitted. The platform will also provide a ranking of the competition and battles.

Below there is the table of goals that the platform will achieve:

#	Goal
G1	Enable ED to manage Competitions
G2	Enable ED to manage Code Battles within Competitions
G3	Enable ST to participate in a Competition
G4	Enable ST to be part of a team within a battle
G5	Send Notifications to STs
G6	Automatically Create GitHub Repositories for Every Battle in a Competition
G7	Synchronize the Submission of Each Candidate with Their GitHub Repository
G8	CKB Provides an evaluation of the code submitted
G9	Allow users to View Rankings in both battles and competitions
G10	Allow to assign badges to the students

Table 1.1: List of goals

1.2.   Scope

1.2.1.   World phenomena

ID	Definitions
WP1	ED wants to create a competitions
WP2	ED wants to create a battle
WP3	ST wants to participate in a competition
WP4	ST wants to participate in a battle
WP5	ST set up GitHub actions

Table 1.2: List of the world phenomena

1.2.2.   Shared phenomena

ID	Definitions
SP1	ST creates an account on the platform
SP2	ED creates an account in the platform
SP3	ST logs in to the platform
SP4	ED logs in to the platform
SP5	ST registers for the competitions before the deadline
SP6	ED creates a badge with certain rules
SP7	ED manually evaluates the code submitted by students
SP8	ED creates a competition
SP9	ED creates a battle within a competition
SP10	ED closes a competition
SP11	ST pushes new commit(s) into their GitHub repository before the deadline

SP12	ST invites other STs to participate in a battle as a team
SP13	ST subscribes as a single/team for an incoming battle before the deadline
SP14	CKB sends a notification that a competition is available to ST
SP15	CKB sends a notification that a battle is created inside a competition to ST
SP16	CKB sends a notification that a competition has ended to ST
SP17	CKB sends a notification that a battle has ended to ST
SP18	CKB sends links to the GitHub repository to all the ST subscribed
SP19	CKB updates scores for each ST
SP20	CKB gives badge to ST
SP21	CKB updates the ranking of the competition
SP22	CKB updates the ranking of the battle

Table 1.3: List of the shared phenomena

1.3. Definitions, Acronyms, Abbreviations

1.3.1. Definitions

User	Anyone interacting with the system, it can be both a Student or an Educator
Manage	Create, supervise and edit a certain element of the application.

Table 1.4: List of definitions

1.3.2.   Acronyms

ST	Student
ED	Educator
CKB	CodaKataBattle
RASD	Requirements Analysis and Specification Document
SAT	Static Analyzer Tool
T	Team

Table 1.5: List of Acronyms

1.3.3.   Abbreviations

WPX	World Phenomena X
SPX	Shared Phenomena X
GX	Goal Number X
DX	Domain Assumption X
UCX	Use Case X

Table 1.6: List of abbreviations

1.4.   Revision history

1.5.   Reference Documents

1.6.   Document Structure

2 | Overall Description

2.1. Product perspective

2.1.1. Class Diagram

Here we provide the class diagram of our system and a brief description on some of the key aspects. The very first thing to explain is its logical split of the *Static Analyzer* from the rest of the system, the reason for this is that we wanted to have a system that could be more *flexible* than usual. The two logical parts can be implemented in the same tier or can also be split physically in terms of devices. What does not change is that when the STs set up github actions on their repository, the server that will be called when a commit is pushed is the **SA_ Server**, which will do the necessary checks and then sends the results to the main server (**Results API** has this purpose), where the score will be computed. Note that the Static Analyzer can be configured from the main server by using the provided APIs in **SA_ Configuration API** (e.g., upload the test cases...). As asked by the assignment we distinguished two types of users, **Students** and **Educators**, in order to differentiate the features the platform offers to the two. The same concept is also applied to the participants of a battle, which does not distinguish a team from a single student; this is very useful when the battle finishes and the results have to be translated into the competition. The **NotificationHandler** has the purpose of handling the notifications of the users and it provides the procedures to do just so.
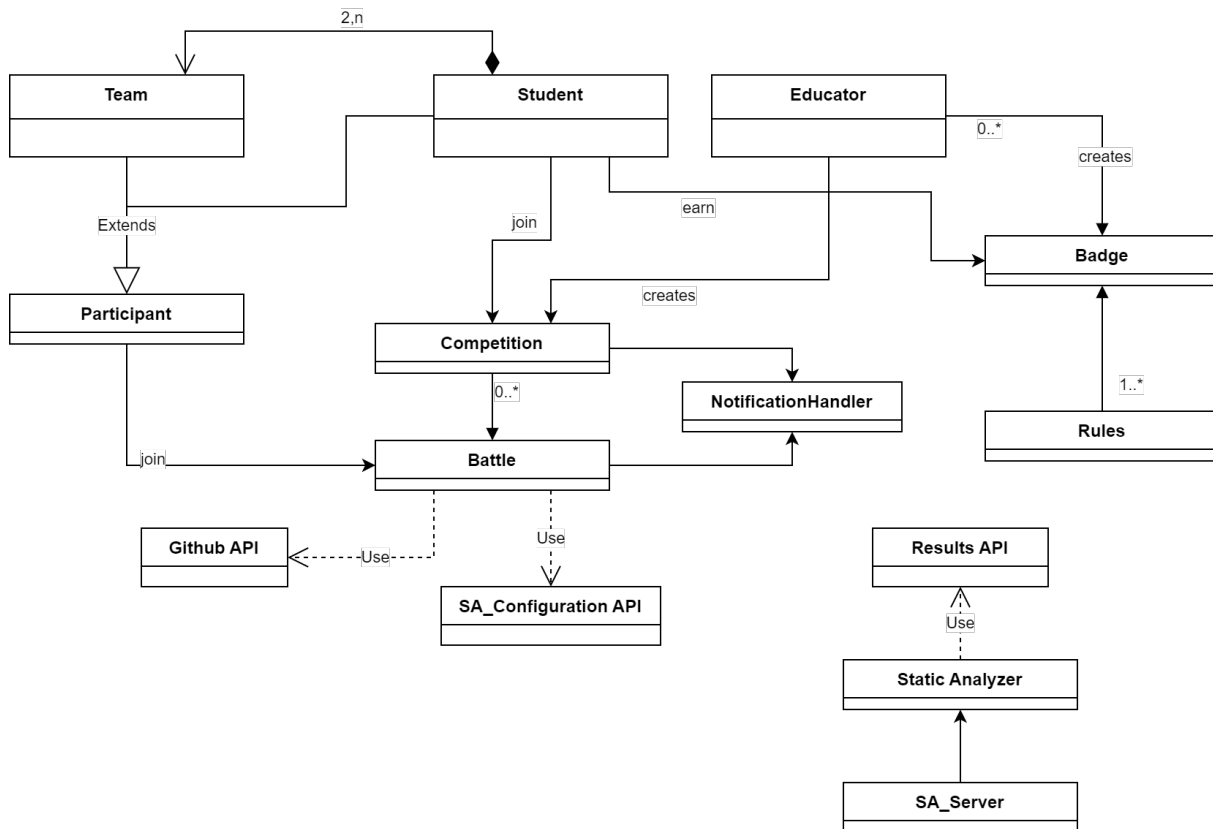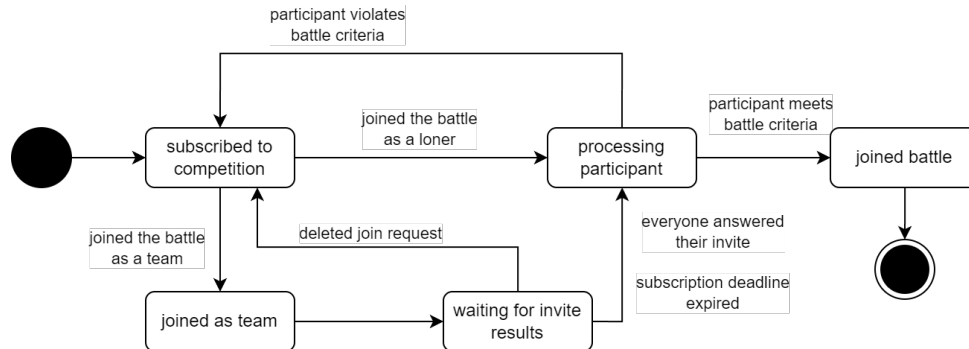
2.1.2. State Diagrams

Commit in github repository

Here there is the state diagram related to a participant pushing a commit in their repository.
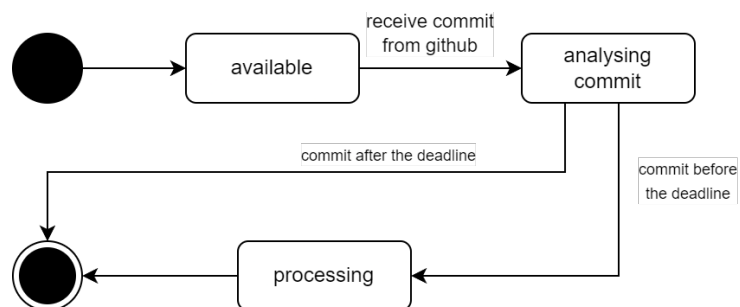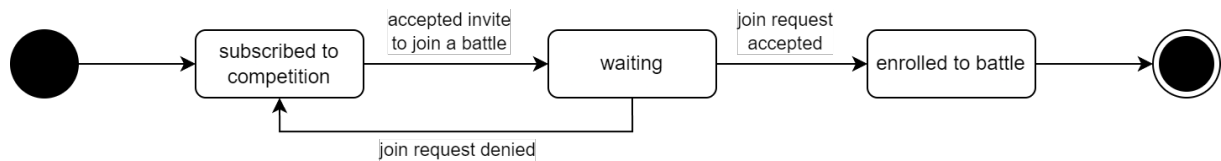
Join battle

Here you can find the state diagram representing the process of joining a battle.

Accept invite to join battle

Below you can find the state diagram representing the process of a ST accepting an invite to join a battle.

2.1.3. Scenarios

Scenario 1: Professor Harry is a professor teaching at Politecnico di Milano together with professor Donald. Harry would like to encourage his students to study during the course, instead of having to study everything a few days before the exam. To do so he came up with the idea to create a challenge where the students can test their preparation and earn some extra points in the exam. While talking with other colleagues, Prof. Harry discovered CKB and he thought it was the perfect fit to implement his idea. The first thing he does is to go to the webpage of CKB and create an account by clicking the ***Sign up*** button and providing some information about himself. Afterwards he is redirected to the home page of the platform where he can click the button ***Create Competition***, and finally he inserts the name of the competition and the subscription deadline. At this point he wants to invite his colleague Donald to manage the competition with him; since he is an ED in the competition he can click on ***Invite Educator*** in the competition page, then provides the email of Donald's account, who will be part of the competition once he accepts the invite.

Scenario 2: Professor Harry is an ED of a competition, within which he wants to create

a battle. To do so he enters the dashboard of the competition, clicks on the button *Create Battle* and provides everything the platform needs: description, test cases, build automation scripts, deadlines, accepted sizes of groups. Marco, a ST who subscribed to the competition, received the notitication about the newly created battle via email. Outside of the platform Marco agreed with a couple of friends to participate in the battle together. Marco then goes on the competition page and finds the newly created battle, here he finds two buttons, *Participate as: 1. Loner 2. Team*; he clicks the second button to participate as a team and invites his friends by providing the platform his friends' account email. Once Marco's friends accepted the invite the subscription to the battle will be automatically finalized by the platform.

Scenario 3: Professor Harry wants to give credit to the hardest working student, so while creating the competition he decided to create a new badge. The hardest working ST is the one that has written the highest amount of code lines among all the battles in the same competition. To implement this badge Prof. Harry must create a new variable *hardest_worker* and provide the code that defines how to compute the value of such variable. Some time after the specification of this new badge, a ST, participating in the competition and in the current battle, called Marco, pushes a commit to his repository. Since all students are supposed to setup *GitHub Actions*, CKB is notified about Marco's commit, so it proceeds to run the required processes to calculate the new score, but also checks if Marco acquired new badges by checking their rules. Assuming that with the last commit Marco has now the most written lines of code, CKB assigns to him the *hardest_worker* badge.

Scenario 4: Marco and his team participated in a battle provided by Prof. Harry, one of the ED of the competition. Since the battle ends the next day, Marco wants to look at the partial rankings of the battle, so he goes on the page related to the battle and clicks on the *Results* section, and sees his team at the bottom of the chart. Understandably, Marco's team resumes to work on the problem and they are able to commit a new version of their solution, which increased their placement in the partial rankings of the battle. The submission deadline now expired and the EDs now want to manually check the work of their STs to assign manually a score to each team; to do this Prof. Harry goes on the battle page and clicks on *Perform Manual check*, which will redirect the ED to another page where he can inspect the source code of each team and give a score to each final work. Once this consolidation phase has been declared finished by an ED, CKB sends to all the STs subscribed to the competition a notification that the battle's results are available and the global scores of the competition have been updated.

2.2. Product functions

The ED can create a new competition
The CKB allows the ED to create a new competition by clicking on the **Create Competition** button on the home page, then providing the name of the competition and the subscription deadline.

The ED inside of the competition have the possibility to create a new battle by clicking on the **Create Battle** button, where he can then insert the description, test cases and the solution. He can also set which aspects of the code he wants that CKB evaluate, such as the quality of the code, the number of lines of code, the security, the readability and the maintainability. Moreover he can select the minimum and maximum number of team components and the deadline for the submission.

Inside of each battle the ED can add badges by clicking on the **Add Badge** button, where he can then insert the name of the badge, the description and the rules to assign the badge.

The ED can also invite other EDs to manage the competition with him by clicking on the **Invite Educator** button and providing the email of the account of the ED he wants to invite. This will allow the colleague to change the settings of the competition and create new battles.

The ST can participate in a competition
The ST after logging in the platform can see the list of the competitions available, and can subscribe to them by clicking on the **Subscribe** button. When the student clicks on the button he can choose to participate as a loner or as a team, in the latter case he has two possibilities:

- Create a new team by clicking on the **Create Team** button, where he can then insert the name of the team and the email of the other STs he wants to invite. Once the other STs accepted the invite the subscription to the competition will be automatically finalized by the platform.

- Join an existing team by clicking on the **Join Team** button. Once the team leader accepted the request,the ST will be added to the team.

After subscribing to a competition, the ST can see the list of the battles inside the competition and can subscribe to them by clicking on the **Subscribe** button. When the student clicks on the button he can choose to participate as a loner or as a team, in the

latter case he can invite other STs to participate in the battle with him by sending them an invite via email.

Inside the competition ST can also see the general ranking of the competition, which is updated after each battle. It is also possible to see the partial ranking of each battle he have partecipated in, which is updated after each submission.

Another feature available to the ST is the possibility to see the list of the badges he earned, and the list of the badges he can earn with the corresponding rules.

The CKB can evaluate the submissions
The CKB is able to automatically evaluate the submissions of the STs by running the test cases provided by the EDs. Each new code submission made on the Github repository of the STs is notified to the CKB, which then runs the test cases on the code.

The test cases are run in a sandbox environment to prevent malicious code from damaging the system. The CKB is also able to run the build automation scripts provided by the EDs to check if the code compiles and if it satisfies the requirements.

The code is evaluated also considering the quality of the sources by running static analysis tools on the code that considers the complexity of the code, the readability and the maintainability.

The CKB after performing all the checks assigns a score to the submission and updates the ranking of the battle and the competition. It also checks if the STs earned new badges by checking their rules and, in case, assigns them to the STs.

It is also possible for the ED managing the battle to manually evaluate the submissions by clicking on the ***Perform Manual Check*** button on the battle page. This will redirect the ED to another page where he can inspect the source code of each team and give a score to each final work. Once this consolidation phase has been declared finished by an ED, CKB sends to all the STs subscribed to the competition a notification that the battle's results are available and the global scores of the competition have been updated.

2.3. User characteristics

The actors that are going to use the CKB system are:

- **Educator (ED)**: an educator is a user that can create competitions and battles within competitions. He can set the parameters of the battles and the deadlines. He can also invite other educators to manage the competition with him.

- **Student (ST)**: a student is a user that can create teams and join battles as a team or individually. He can earn points and badges by partecipating in the competitions and battles.

2.4. Assumptions, dependencies and constraints

ID	Description
DA1	ST owns a device able to connect to the internet
DA2	ST owns a GitHub account
DA3	ST has installed Git on his computer
DA4	ST knows how to use Git
DA5	ED knows how to use Git
DA6	ED owns a device able to connect to the internet
DA7	ED writes correct tests
DA8	ED correctly evaluates the final source code of a T
DA9	GitHub permits automatic push to a repository
DA10	GitHub permits automatically pull from a repository
DA11	ST knows the usernames of other STs they want to invite to a T
DA12	STs has an internet connection
DA13	ED has an internet connection
DA14	ST writes code only with languages that are treatable by the platform

| DA15 | ED knows the email of the other EDs he wants to invite to manage a competition |
| DA16 | ED writes the correct badge' rules |

Table 2.1: List of the domain assumption

3 | Specific Requirements

3.1. External Interface Requirements

3.1.1. User Interfaces

The CKB user interface will be a web page that will be accessed through a web browser. The web page will be designed to be simple and easy to use with the support for multiple screen sizes and devices.

3.1.2. Hardware Interfaces

The platform requires a computer with a web browser and an internet connection to access the CKB web page.

3.1.3. Software Interfaces

CKB will be using some external interfaces in order to provide the service. The external interfaces are listed below:

- **Github API:** CKB will use Github as source control system for the projects. The Github API will be used to create the repositories for each team in the battle and share it with the team members.

- **Static Analyzer API:** CKB will use a static analysis tool to analyze the code of each team after every new commit and will use the result of the analysis to assign points to each team. The purpose of this interface is to give the possibility to the system to configure the analyzer as the educator needs, in terms of test cases, languages supported and other parameters.

- **Results API:** The *SA_Server* will use this API to send the results of the analysis to the main server.

- **Github Actions:** Github Actions will notify the system when a participant has

pushed a new commit to its repository. The *SA_Server* will be listening to this notifications and will trigger the analysis of the source code.

3.1.4. Communication Interfaces

All the communication between CKB, the external interfaces and the user will be done using HTTPS protocol.

3.1.5. List of Requirements

Requirement	Description
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R3	CKB shall allow ST to connect their Github account
R4	CKB shall allow ED to create competition
R5	CKB shall allow ED to create battle within a competition
R6	CKB shall allow ED to invite other EDs to create battles in a competition
R7	CKB shall allow ED to upload the code kata
R8	CKB shall allow ED to set a registration deadline to the battle
R9	CKB shall allow ED to set a minimum number of STs per group in a battle
R10	CKB shall allow ED to set the maximum number of STs per group in a battle
R11	CKB shall allow ED to set a final submission deadline
R12	CKB shall allow ED to set how to perform static analysis
R13	CKB shall allow ST to subscribe to a competition
R14	CKB shall send notifications about a new competition to ST

R15	CKB shall send notification about battle created within a competition ST are subscribed in
R16	CKB shall allow ST to join a battle on his own
R17	CKB shall allow ST to invite other ST in a T for a battle
R18	CKB shall create a GitHub repository containing the code kata
R19	CKB shall send the Github repository link to ST member of a T competing in the competition
R20	CKB shall supply API to call with Github actions
R21	CKB shall be able to pull sources from GitHub
R22	CKB shall be able to send the ST source code to the correct SAT
R23	CKB shall be able to receive the evaluation given by SAT on a source code
R24	CKB shall be able to run tests on code
R25	CKB shall evaluate the code in terms of test cases passed
R26	CKB shall evaluate the code in terms of timeliness
R27	CKB shall allow ED to assign a score to codes
R28	CKB shall update the score of a T (as soon as new push actions are performed)
R29	CKB shall allow ED to go through sources produced by Ts
R30	CKB shall notify ST when final battle ranks are available
R31	CKB shall update the personal competition score of an ST at the end of each battle
R32	CKB shall create a rank with students' performances in a competition
R33	CKB shall allow ST to see all ST's rank in battle where is enrolled

R34	CKB shall allow ED to see all ST's ranks in the battle that he/she manages
R35	CKB shall allow EDs and STs to see all ST's rank in competitions
R36	CKB shall update the personal competition score of an ST at the end of each battle
R37	CKB shall allow ST to see the list of ongoing competitions
R38	CKB shall allow ED to close a competition
R39	CKB shall allow ED to define badges in the context of a competition
R40	CKB shall assign badges to students at the end of the competition
R41	CKB shall allow ED to define new rules for badges
R42	CKB shall allow ED to define new variables for badges
R43	CKB shall allow users to visualize badges obtained by an ST
R44	CKB shall allow users to visualize an ST profile
R45	CKB shall allow ST to join a T for which is invited
R46	CKB shall allow ST to join a public T
R47	CKB shall allow ST to create a T
R48	CKB shall allow ST to set a T to public or private
R49	CKB can distinguish between an ED user and a ST user
R50	CKB shall not allow ST/ED to see the rankings of battles in competitions they are not enrolled in
R51	CKB shall have the environments for all the programming language it supports

3.1.6.   Mapping on Goals

Goal	Requirements	Domain Assumptions
G1	R1 R2 R49 R3 R9 R38 R39 R41 R42 R50	DA4 DA13 DA15 DA16
G2	R1 R2 R49 R4 R5 R6 R34 R7 R8 R9 R10 R50	DA5 DA6 DA8 DA13
G3	R1 R2 R13 R49 R2 R3 R13 R14 R15 R33 R37 R50	DA1 DA2 DA3 DA4
G4	R1 R2 R49 R13 R15 R45 R46 R47 R48	DA1 DA2 DA3 DA4 DA9 DA10 DA11
G5	R1 R2 R49 R12 R13 R30	DA1 DA11
G6	R18 R19 R20	DA9
G7	R2 R19 R20 R21	DA1 DA2 DA3 DA4 DA11
G8	R22 R23 R24 R25 R26 R27 R28 R29 R31 R32 R51 R36	DA5 DA6 DA12 DA13 DA14
G9	R1 R2 R13 R15 R29 R30 R34 R33 R35 R44	DA1 DA11 DA5 DA12
G10	R39 R40 R43 R49	DA1 DA5 DA11 DA12 DA15

Table 3.2: Mapping between goals, requirements, and domain assumptions

G1	**Enable ED to manage competitions**
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R3	CKB shall allow ST to connect their Github account
R9	CKB shall allow ED to set a minimum number of STs per group in a battle
R38	CKB shall allow ED to close a competition
R39	CKB shall allow ED to define badges in the context of a competition
R41	CKB shall allow ED to define new rules for badges
R42	CKB shall allow ED to define new variables for badges
R49	CKB can distinguish between an ED user and an ST user
R50	CKB shall not allow ST/ED to see the rankings of battles in competitions they are not enrolled in
DA4	ST knows how to use Git
DA13	ED has an internet connection
DA15	ED knows the email of the other EDs he wants to invite to manage a competition
DA16	ED writes the correct badge' rules

Table 3.3: Specific mapping on G1

G2	Enable ED to manage Code Battles within Competitions
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R3	CKB shall allow ST to connect their Github account
R4	CKB shall allow ED to create competition
R5	CKB shall allow ED to create battle within a competition
R6	CKB shall allow ED to invite other EDs to create battles in a competition
R7	CKB shall allow ED to upload the code kata
R8	CKB shall allow ED to set a registration deadline to the battle
R9	CKB shall allow ED to set a minimum number of STs per group in a battle
R10	CKB shall allow ED to set the maximum number of STs per group in a battle
R34	CKB shall allow ED to see all ST's ranks in the battle that he/she manages
R38	CKB shall allow ED to close a competition
R39	CKB shall allow ED to define badges in the context of a competition
R41	CKB shall allow ED to define new rules for badges
R42	CKB shall allow ED to define new variables for badges
R49	CKB can distinguish between an ED user and an ST user
R50	CKB shall not allow ST/ED to see the rankings of battles in competitions they are not enrolled in
DA5	ED knows how to use Git
DA6	ED owns a device able to connect to the internet
DA8	ED correctly evaluates the final source code of a T
DA13	ED has an internet connection

Table 3.4: Specific mapping on G2

G3	**Enable ST to participate in a Competition**
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R3	CKB shall allow ST to connect their Github account
R13	CKB shall allow ST to subscribe to a competition
R14	CKB shall send notifications about a new competition to ST
R15	CKB shall send notification about battle created within a competition ST is subscribed in
R33	CKB shall allow ST to see all ST's rank in a battle where is enrolled
R37	CKB shall allow ST to see the list of ongoing competitions
R49	CKB can distinguish between an ED user and an ST user
R50	CKB shall not allow ST/ED to see the rankings of battles in competitions they are not enrolled in
DA1	ST owns a device able to connect to the internet
DA2	ST owns a GitHub account
DA3	ST has installed Git on his computer
DA4	ST knows how to use Git

Table 3.5: Specific mapping on G3

G4	**Enable ST to be part of a team within a battle**
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R49	CKB can distinguish between an ED user and an ST user
R13	CKB shall allow ST to subscribe to a competition
R15	CKB shall send notification about battle created within a competition ST is subscribed in
R45	CKB shall allow ST to join a T for which is invited
R46	CKB shall allow ST to join a public T
R47	CKB shall allow ST to create a T
R48	CKB shall allow ST to set a T to public or private
DA1	ST owns a device able to connect to the internet
DA2	ST owns a GitHub account
DA3	ST has installed Git on his computer
DA4	ST knows how to use Git
DA9	GitHub permits automatic push to a repository
DA10	GitHub permits automatically pull from a repository
DA11	ST knows the usernames of other STs they want to invite to a T

Table 3.7: Specific mapping on G4

G5	**Send Notifications to STs**
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R49	CKB can distinguish between an ED user and an ST user
R12	CKB shall allow ED to set how to perform static analysis
R13	CKB shall allow ST to subscribe to a competition
R30	CKB shall notify ST when final battle ranks are available
DA1	ST owns a device able to connect to the internet
DA11	ST knows the usernames of other STs they want to invite to a T

Table 3.9: Specific mapping on G5

G6	**Automatically Create GitHub Repositories for Every Battle in a Competition**
R18	CKB shall create a GitHub repository containing the code kata
R19	CKB shall send the Github repository link to ST member of a T competing in the competition
R20	CKB shall supply API to call with Github actions
DA9	GitHub permits automatic push to a repository

Table 3.11: Specific mapping on G6

G7	Synchronize the Submission of Each Candidate with Their GitHub Repository
R2	CKB shall allow users to log in
R19	CKB shall send the Github repository link to ST member of a T competing in the competition
R20	CKB shall supply API to call with Github actions
R21	CKB shall be able to pull sources from GitHub
DA1	ST owns a device able to connect to the internet
DA2	ST owns a GitHub account
DA3	ST has installed Git on his computer
DA4	ST knows how to use Git
DA11	ST knows the usernames of other STs they want to invite to a T

Table 3.13: Specific mapping on G7

G8	CKB Provides an evaluation of the code submitted
R22	CKB shall be able to send the ST source code to the correct SAT
R23	CKB shall be able to receive the evaluation given by SAT on a source code
R24	CKB shall be able to run tests on code
R25	CKB shall evaluate the code in terms of test cases passed
R26	CKB shall evaluate the code in terms of timeliness
R27	CKB shall allow ED to assign a score to codes
R28	CKB shall update the score of a T (as soon as new push actions are performed)
R29	CKB shall allow ED to go through sources produced by Ts
R31	CKB shall update the personal competition score of an ST at the end of each battle
R32	CKB shall create a rank with students' performances in a competition
R51	CKB shall have the environments for all the programming language it supports
R36	CKB shall update the personal competition score of an ST at the end of each battle
DA5	ED knows how to use Git
DA6	ED owns a device able to connect to the internet
DA12	GitHub permits automatic push to a repository
DA13	ED has an internet connection
DA14	ST writes code only with languages that are treatable by the platform

Table 3.15: Specific mapping on G8

G9	**Allow users to View Rankings in both battles and competitions**
R1	CKB shall allow an unregistered user to create an account
R2	CKB shall allow users to log in
R13	CKB shall allow ST to subscribe to a competition
R15	CKB shall send notification about battle created within a competition ST is subscribed in
R29	CKB shall allow ED to go through sources produced by Ts
R30	CKB shall notify ST when final battle ranks are available
R34	CKB shall allow ED to see all ST's ranks in the battle that he/she manages
R33	CKB shall allow ST to see all ST's rank in a battle where is enrolled
R35	CKB shall allow EDs and STs to see all ST's rank in competitions
R44	CKB shall allow users to visualize badges obtained by an ST
DA1	ST owns a device able to connect to the internet
DA11	ST knows the usernames of other STs they want to invite to a T
DA5	ED knows how to use Git
DA12	GitHub permits automatically pull from a repository

Table 3.17: Specific mapping on G9

G10	**Allow to assign badges to the students**
R39	CKB shall allow ED to define badges in the context of a competition
R40	CKB shall assign badges to students at the end of the competition
R43	CKB shall allow users to visualize badges obtained by an ST
R49	CKB can distinguish between an ED user and an ST user
DA1	ST owns a device able to connect to the internet
DA5	ED knows how to use Git
DA11	ST knows the usernames of other STs they want to invite to a T
DA12	GitHub permits automatically pull from a repository
DA15	ED knows the email of the other EDs he wants to invite to manage a competition

Table 3.19: Specific mapping on G10

3.1.7. Use Cases

Use case diagram

Here you can find the use case diagram of the platform.
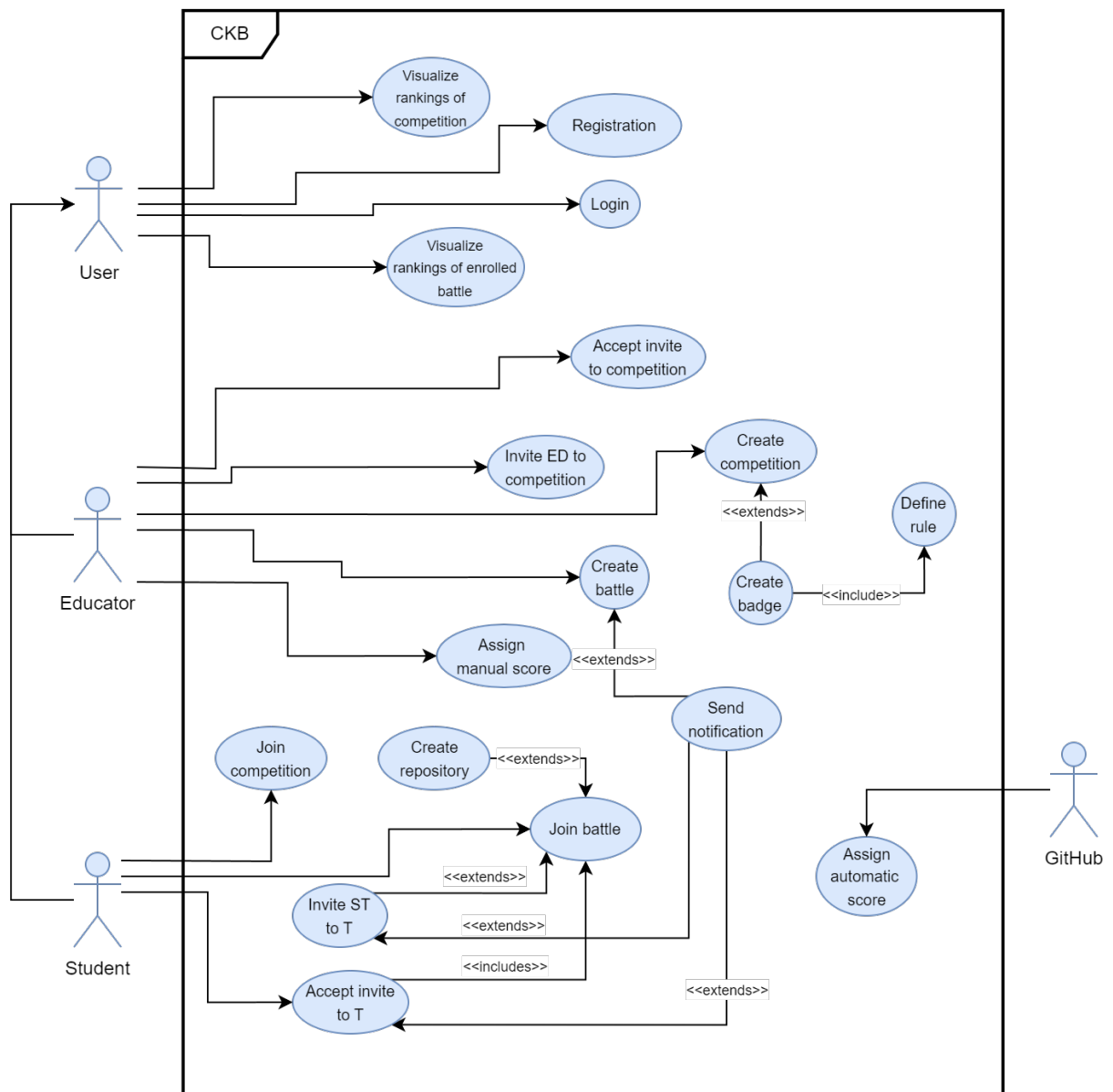
Figure 3.1: Use case diagram of the platform.

For simplicity we decided not to include all the *«include»* relationship related to the ***Login*** use case, since almost every use case requires the user to be logged in, this was to explain that it was not forgotten.

UC1: Unregistered ST creates an account

Actor	Unregistered ST
Entry conditions	The ST isn't already registered in the CKB platform and he clicks on the sign-up button
Event Flow	1. CKB asks the ST to insert the personal information (i.e. name, surname, nickname, e-mail and password)
	2. The ST fills out the form with the requested informations and accepts the "Terms & Conditions" and "Privacy Policy"
	3. CKB validate the inserted information
	4. CKB sends an account activation link to the email inserted by the ST
	5. ST clicks on the link received in the email
	6. CKB confirms the account creation
Exit condition	The ST account is created
Exceptions	3.1 CKB isn't able to validate the information inserted (i.e. duplicate email, duplicate username and wrong email address
	5.1 The link is expired
	In all the cases the ST is notified with an error message

Table 3.20: ST signs up case.
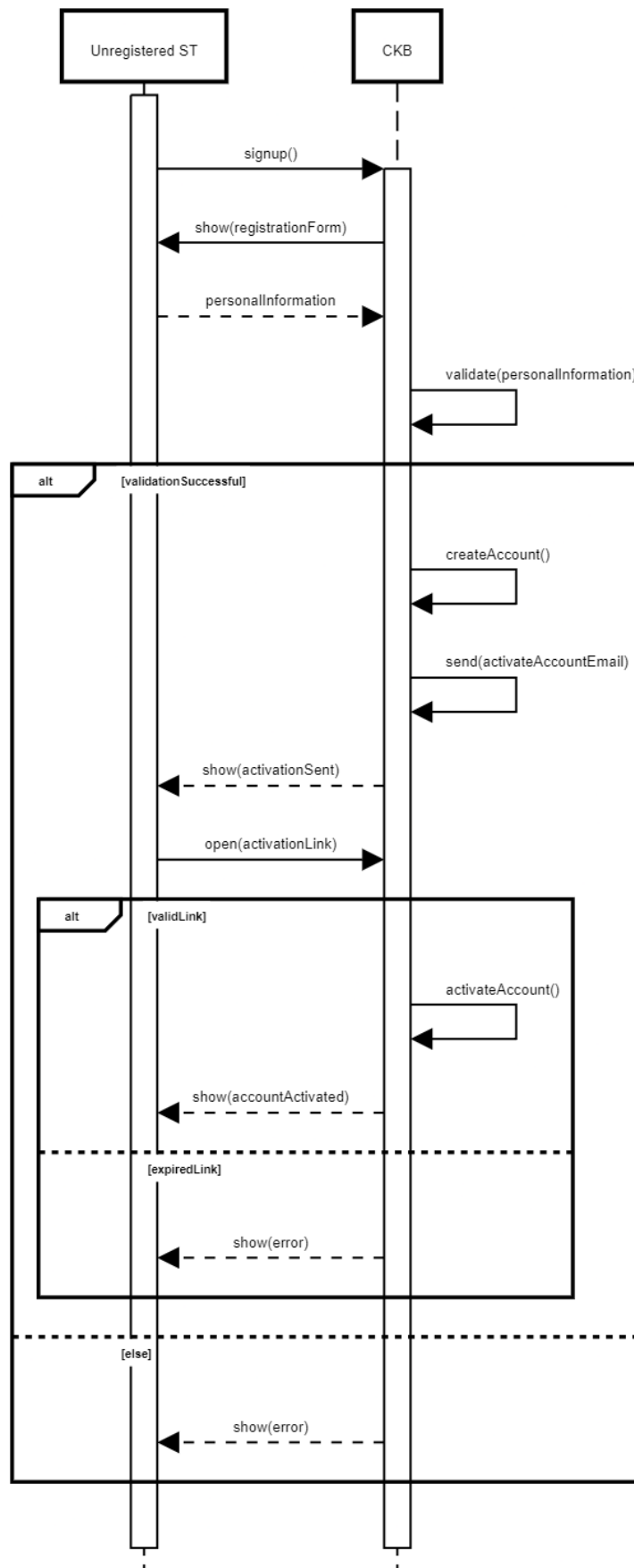
Figure 3.2: UC1 sequence diagram.

UC2: Unregistered ED creates an account

Actor	Unregistered ED
Entry conditions	The ED isn't already registered in the CKB platform and he clicks on the sign-up button
Event Flow	1. CKB asks the ED to insert the personal information (i.e. name, surname, nickname, e-mail and password)
	2. The ED fills out the form with the requested informations and accepts the "Terms & Conditions" and "Privacy Policy"
	3. CKB validate the inserted information
	4. CKB asks the ED to insert the information about the institution (i.e. name, address, city, country, website)
	5. The ED fills out the form with the requested informations
	6. CKB validate the information about the institute
	7. CKB sends an account activation link to the email inserted by the ED
	8. ED clicks on the link received in the email
	9. CKB confirms the account creation
Exit condition	The ED account is created
Exceptions	3.1 CKB isn't able to validate the information inserted (i.e. duplicate email, duplicate username and wrong email address
	6.1 CKB isn't able to validate the information about the institute
	8.1 The link is expired
	In all the cases the ST is notified with an error message

Table 3.21: ED signs up case.
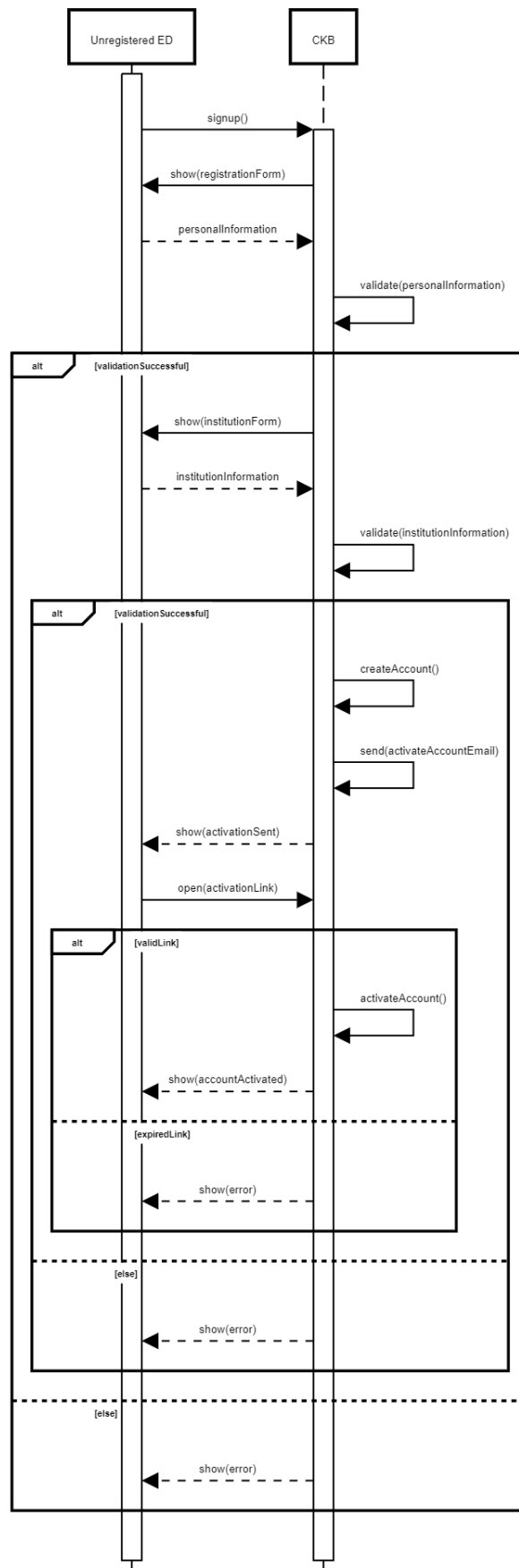
Figure 3.3: UC2 sequence diagram.

UC3: ST or ED logs in

Actor	Registered ST
	Registered ED
Entry conditions	The ED isn already registered in the CKB platform
Event Flow	1. The user clicks on the login button
	2. CKB asks for the username and the password
	3. The user inserts the username and the password
	4. CKB validates the information
	5. CKB redirects the user to the home page
Exit condition	The user is logged in
Exceptions	4.1 The username or the password are wrong
	In this case the user is notified with an error message

Table 3.22: ST or ED logs in.
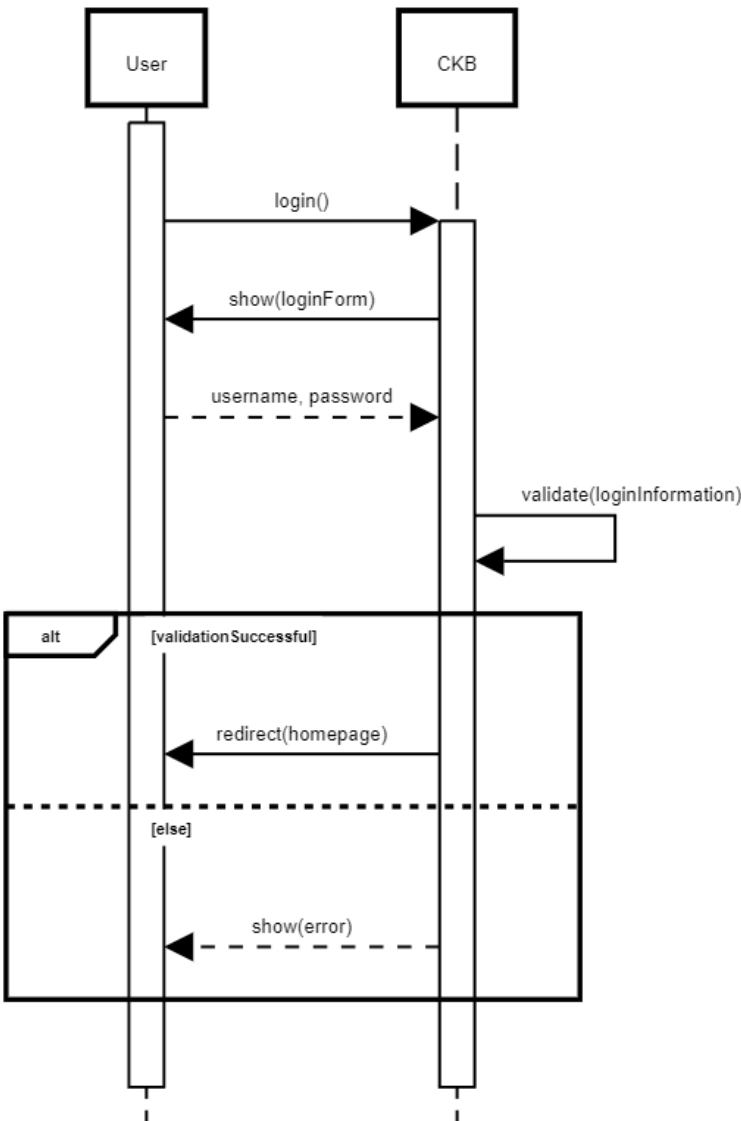
Figure 3.4: UC3 sequence diagram.

UC4: ED creates a new competition

Actor	Registered ED
Entry conditions	The ED is already registered and logged in
	The ED clicks on the "Create Competition" button
Event Flow	1. CKB asks for a name for the competition
	2. The ED inserts the name

	3. CKB checks if the name is available
	4. CKB asks for the information of the competition (i.e. description, start date, end date, programming languages allowed)
	5. CKB validates the information
	6. CKB creates the competition
Exit condition	The competition is correctly created
Exceptions	3.1 The name is already used by another competition
	5.1 CKB isn't able to validate the information
	In the first case the ED is notified with an error message and the flow restarts from the step 1
	In the other case the ED is notified with an error message

Table 3.23: ED creates a competiton.
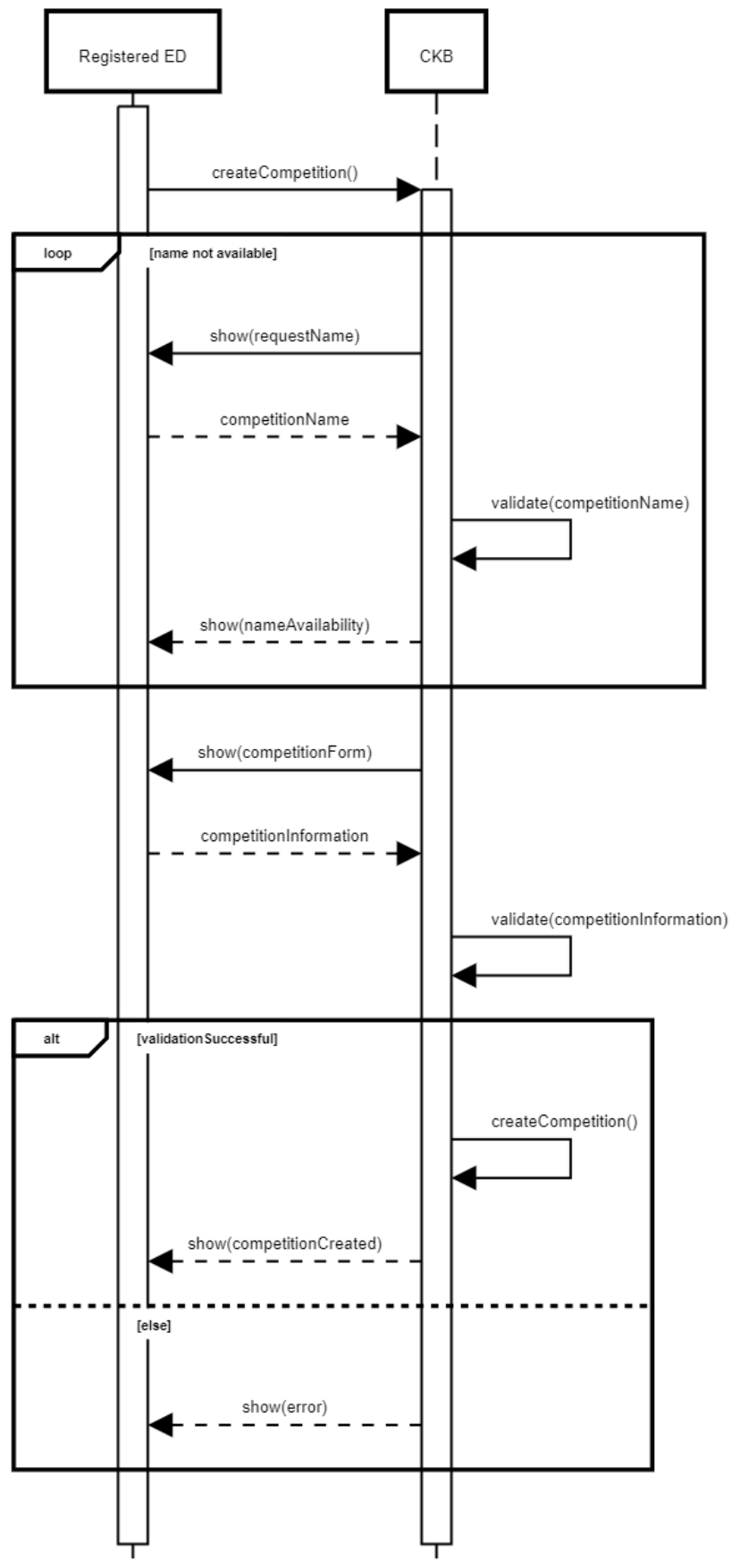
Figure 3.5: UC4 sequence diagram.

UC5: ST joins a competition

Actor	Registered ST
Entry conditions	The ST is already registered and logged in
Event Flow	1. CKB shows the list of the available competitions
	2. The ST selects the competition
	3. CKB shows the information about the competition
	4. The ST clicks on the "Join" button
	5. CKB adds the ST to the competition
	6. ST can now see the competition in the "My Competitions" section
Exit condition	The ST has joined the competition
Exceptions	2.1 There are no competitions available
	In this case the ST visualizes a message that there are no competition available

Table 3.24: ST joins a competition.
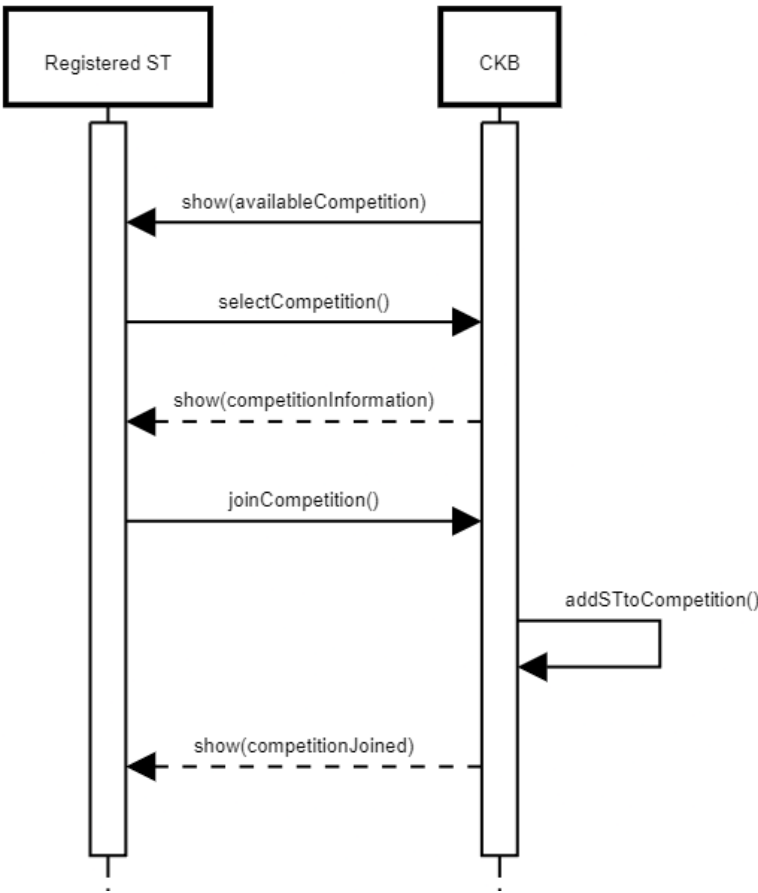
Figure 3.6: UC5 sequence diagram.

UC6: ED creates a new battle inside a competition

Actor	Registered ED
Entry conditions	The ED is already registered and logged in
	The ED is the creator or a collaborator of the competition
	The ED is in the competition page
	The ED clicks on the "Create Battle" button
Event Flow	1. CKB asks for a name for the battle
	2. CKB checks if the name is not already used for another battle inside the competition

	3. CKB asks for the information of the battle (i.e. description, start date, end date, programming languages allowed, number of teams, number of members per team)
	4. ED inserts the information
	5. CKB asks for the configuration of the static analyzer
	6. CKB asks to upload the test cases and the solution
	7. ED uploads the files and insert all the information requested
	8. CKB validates the information
	9. CKB creates the battle
Exit condition	The battle is created correctly
Exceptions	6.1 The upload of the files fails
	7.1 CKB isn't able to validate the information
	In all the cases the ED is notified with an error message

Table 3.25: ED creates a battle inside a competition.
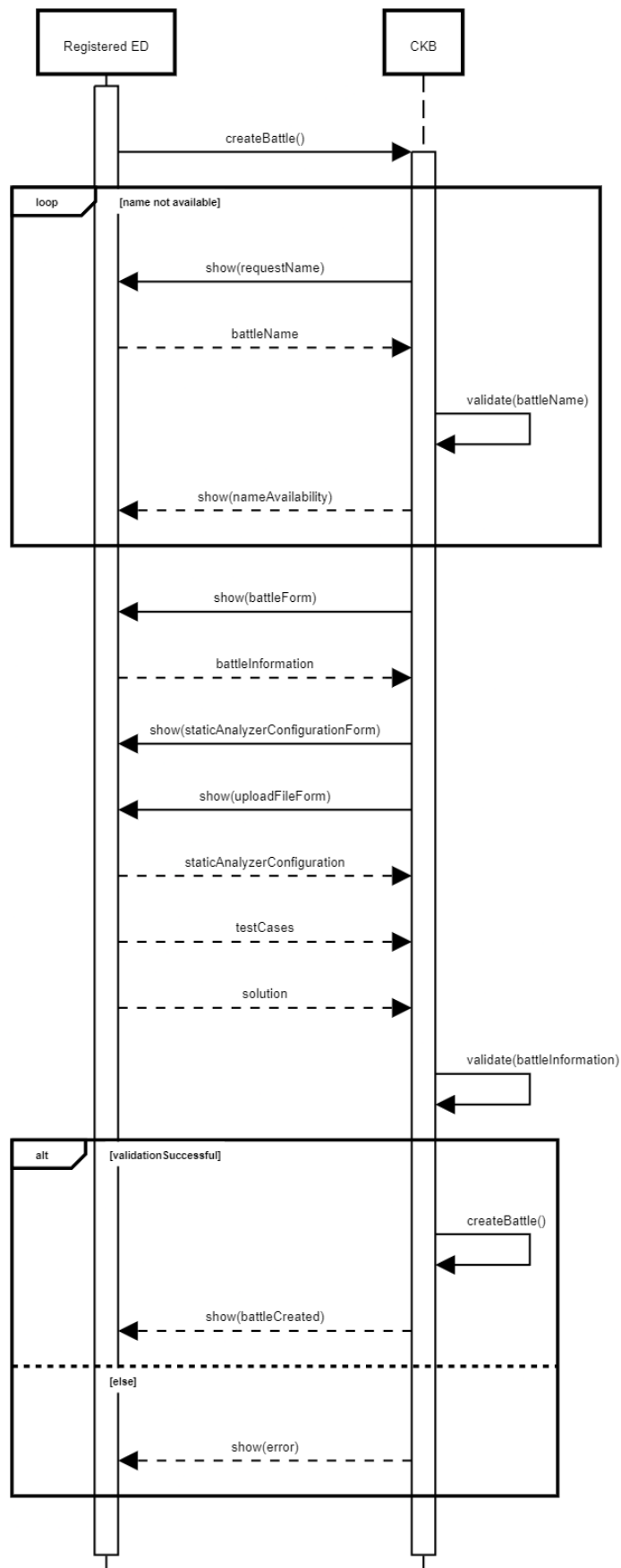
Figure 3.7: UC6 sequence diagram.

UC7: ED invites other EDs to a competition

Actor	Registered ED
Entry conditions	The ED is already registered and logged in
	The ED is the creator or a collaborator of the competition
	The ED is in the competition page
Event Flow	1. ED clicks on the "Settings" button inside the competition
	2. ED scrolls down to the "Collaborators" section
	3. ED clicks on the "Invite" button
	4. CKB asks for the email or the username of the ED to invite
	5. ED inserts the email or the username
	6. CKB validates the information
	7. CKB sends an invitation email to the ED
	8. The invited ED clicks on the link received in the email
	9. CKB confirms the invitation
Exit condition	The invited ED is a collaborator of the competition
Exceptions	6.1 The email or the username is not valid
	8.1 The link is expired
	In all the cases the ED is notified with an error message

Table 3.26: ED invites other EDs to a competition.
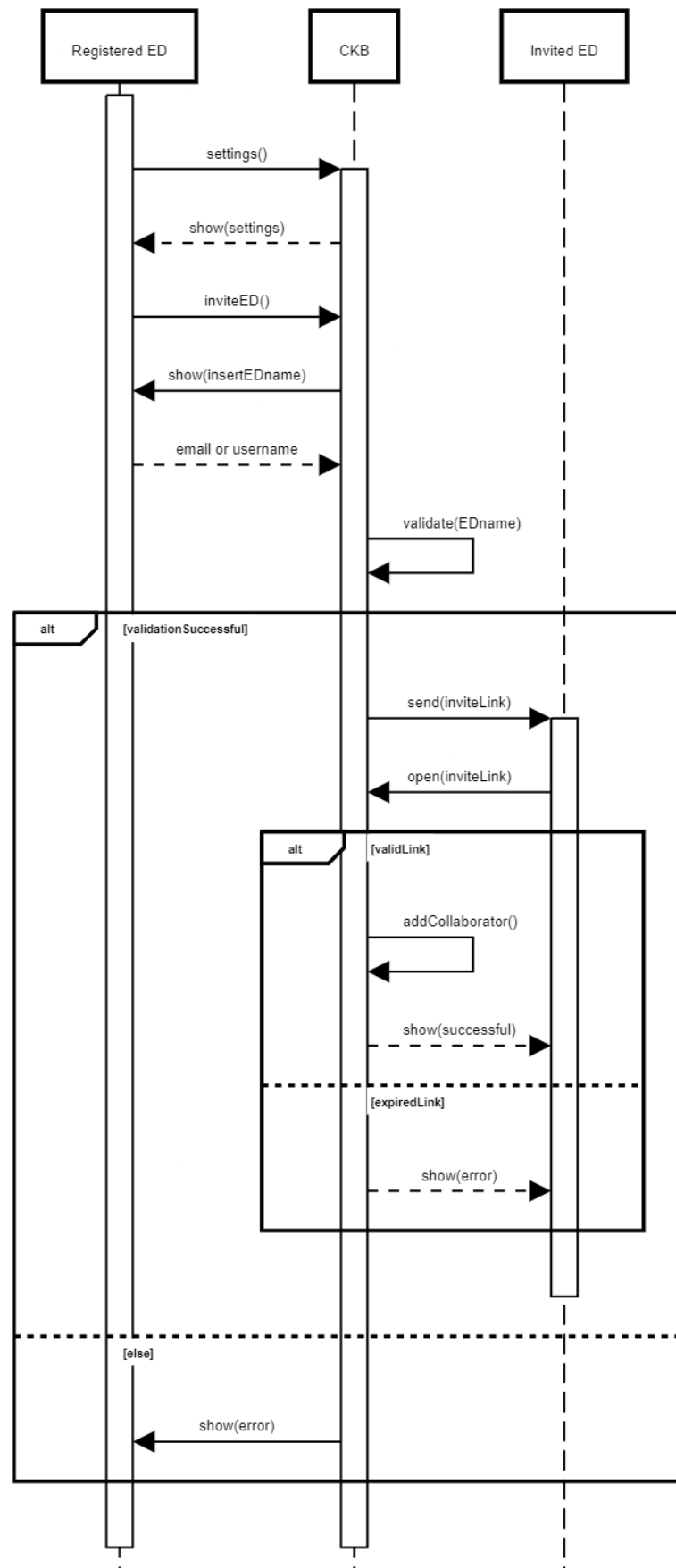
Figure 3.8: UC7 sequence diagram.

UC8: ED creates a new badge inside a competition

Actor	Registered ED
Entry conditions	The ED is already registered and logged in
	The ED is the creator or a collaborator of the competition
	The ED is in the competition page
Event Flow	1. ED clicks on the "Settings" button inside the competition
	2. ED scrolls down to the "Badges" section
	3. ED clicks on the "Create new Badge" button
	4. CKB asks for the information (i.e. name, description) of the badge
	5. ED inserts the information
	6. CKB asks for the rules of the badge
	7. ED inserts the rules
	8. CKB validates the information
	9. CKB creates the badge
Exit condition	The badge is added to the competition
Exceptions	8.1 CKB isn't able to validate the information
	The ED is notified with an error message

Table 3.27: ED creates a badge inside a competition.
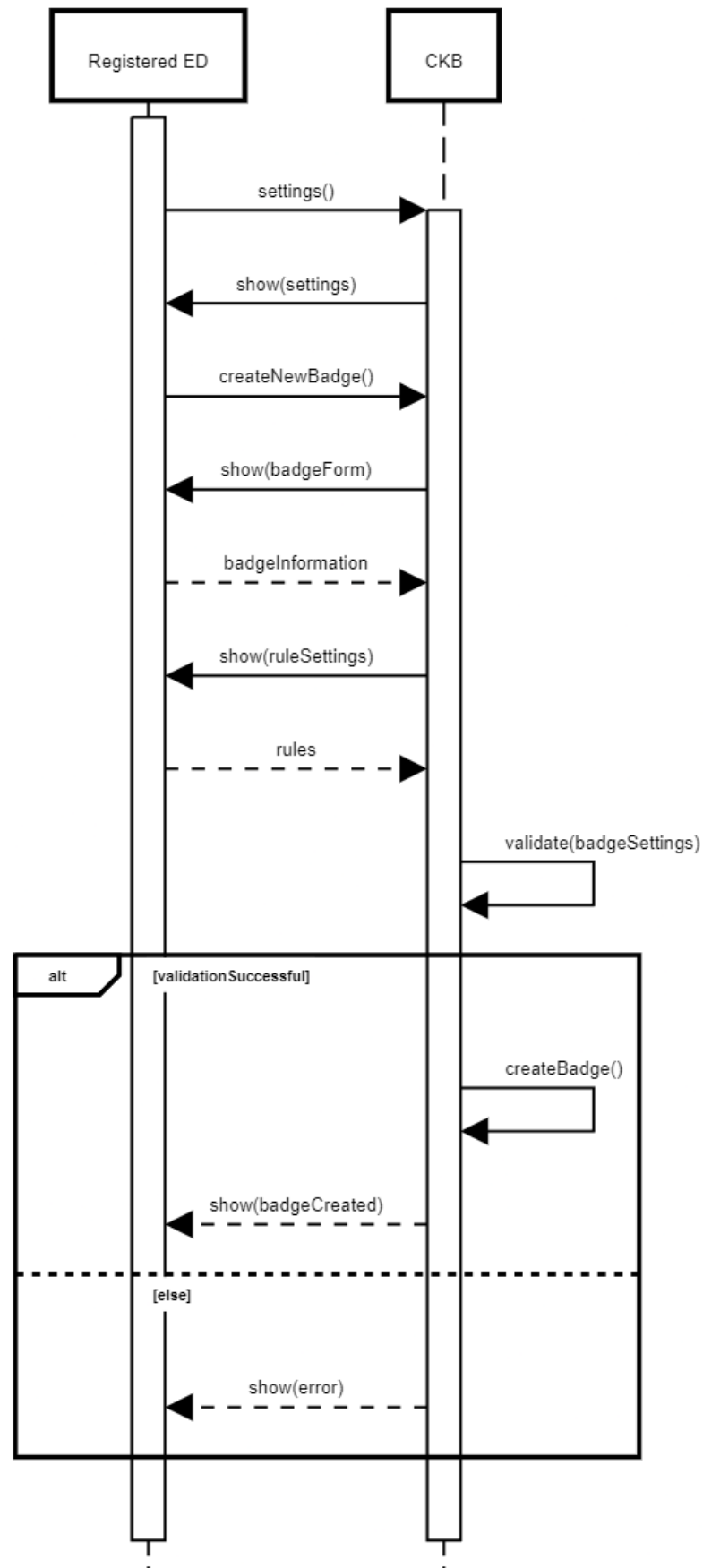
Figure 3.9: UC8 sequence diagram.

UC9: ST receives notification from CKB

Actor	Registered ST
Entry conditions	The ST is already registered
	A competition is created
	A battle inside a competiton in which the ST is enrolled is created
	A battle in which the ST has partecipated has ended
	The rank of a competition in which the ST has partecipated is available
Event Flow	1. CKB sends a notification via email to the ST
	2. The ST clicks on the link received in the email
	3. CKB shows the information about the notification
Exit condition	The ST received the notification
Exceptions	2.1 The link is wrong
	In this case the ST is notified with an error message

Table 3.28: ST receives notification from CKB.
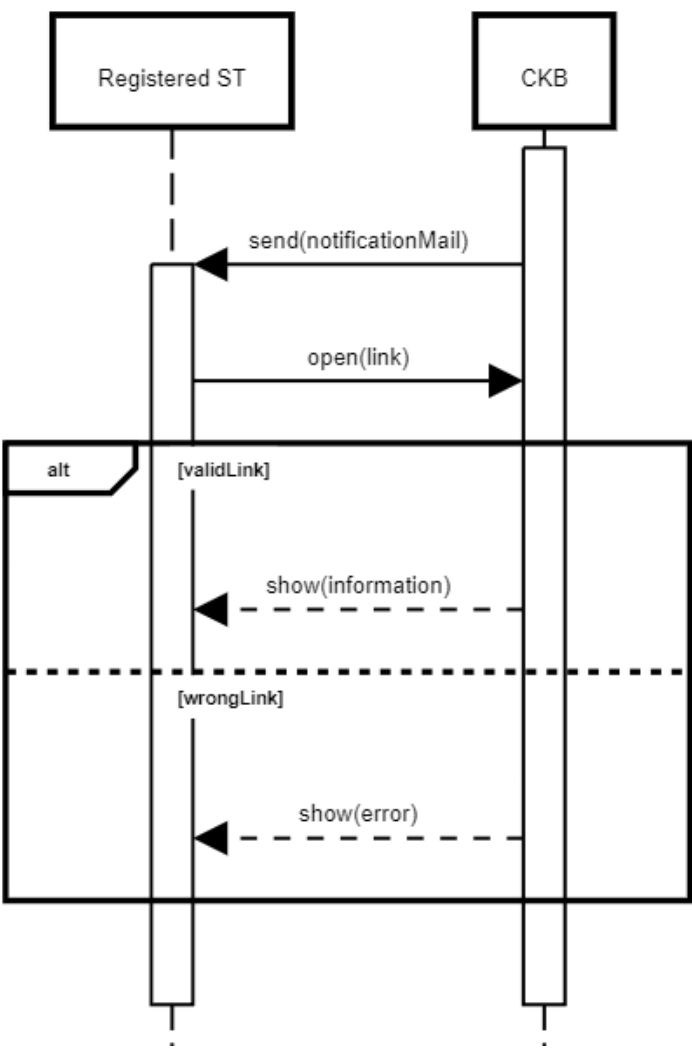
Figure 3.10: UC9 sequence diagram.

UC10: ST visualizes the ranking of their T

Actor	Registered ST
Entry conditions	The ST is already registered and logged in
	The ST is enrolled in a competition
	The ST is enrolled in a battle
	The ST is in the competition page
Event Flow	1. CKB shows the list of the battles in which the ST is enrolled

	2. The ST selects the battle
	3. The ST goes to the "Ranking" section
	4. CKB shows the ranking of the battle
Exit condition	The ST visualize the ranking of their team
Exceptions	2.1 There are no battles available
	In this case the ST visualizes a message that there are no battles available

Table 3.29: ST visualizes the ranking of their T.
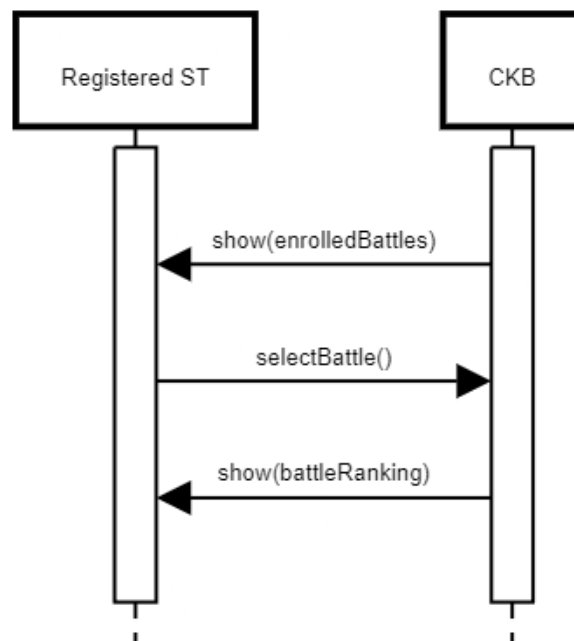
Figure 3.11: UC10 sequence diagram.

UC11: ED manually evaluates the code

Actor	Registered ED
Entry conditions	The ED is already registered
	The ED is the creator or a collaborator of the competition
	The ED is in the battle page
Event Flow	1. ED clicks on a T name
	2. CKB shows the information about the T
	3. ED clicks on the "Evaluate" button
	4. CKB shows the information about the last code submission
	5. ED navigates through the code
	6. ED assigns a score to the code
	7. CKB saves the score
Exit condition	The score is saved
Exceptions	4.1 The T didn't submit any code
	7.1 CKB isn't able to save the score
	In the first case the ED visualize a message that there are no code submissions
	In the other case the ED is notified with an error message

Table 3.30: ED manually evaluates the code.
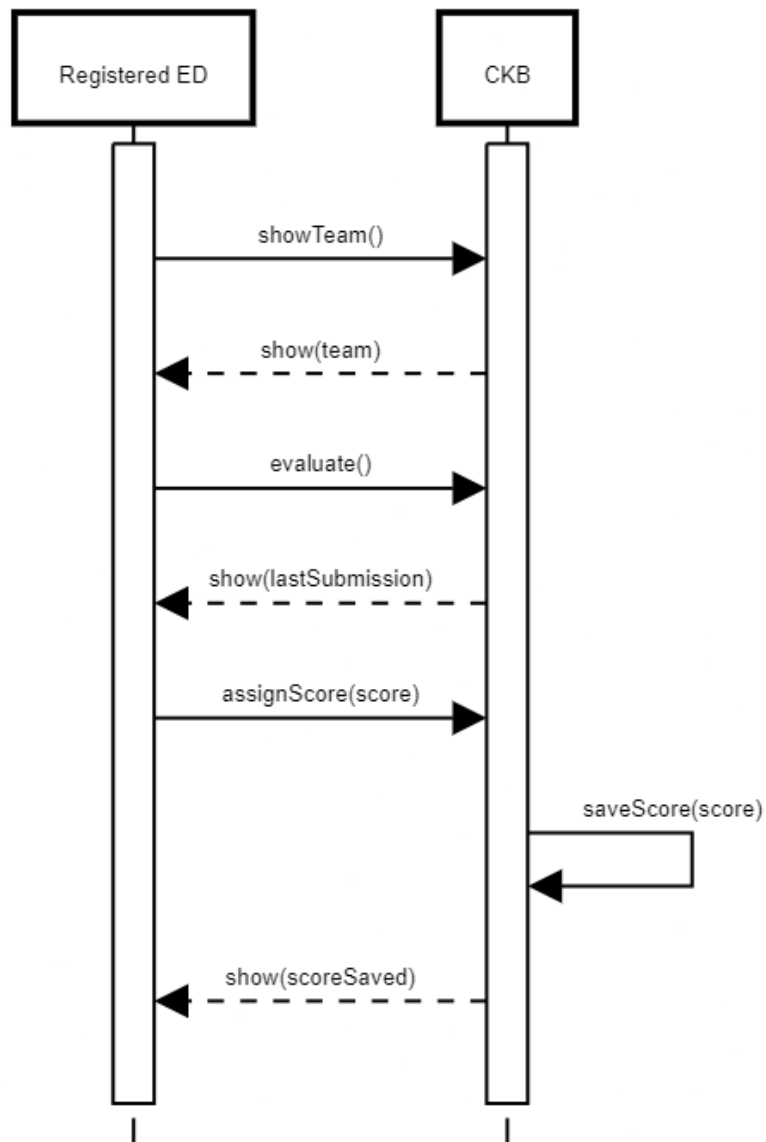
Figure 3.12: UC11 sequence diagram.

3.2.   Performance Requirements

3.3.   Design Constraints

4 | Formal Analysis Using Alloy

Organize this section according to the rules defined in the project description.

5 | Effort Spent

Members of group	Effort spent (hours)	
Filippo Balzarini	Introduction	1h
	Overall description	1h
	Specific requirements	5h
	Formal analysis	1h
	Reasoning	4h
Christian Biffi	Introduction	3h
	Overall description	3h
	Specific requirements	6h
	Formal analysis	0h
	Reasoning	4h
Michele Cavicchioli	Introduction	1h
	Overall description	3h
	Specific requirements	4h
	Formal analysis	0h
	Reasoning	3h

Table 5.1: Effort spent by each member of the group

Bibliography

List of Figures

List of Tables

List of Symbols

Variable	Description	SI unit
$\boldsymbol{u}$	solid displacement	m
$\boldsymbol{u}_f$	fluid displacement	m