POLITECNICO
MILANO 1863

Requirements Analysis and Specification Document
CodeKataBattle

SOFTWARE ENGINEERING 2
PROFESSOR DI NITTO ELISABETTA

CONTI ALESSANDRO   CODICE PERSONA: 10710583   MATRICOLA: 252665
DI PAOLA ANTONIO   CODICE PERSONA: 10717589   MATRICOLA: 956038

Table of contents

1. INTRODUCTION

1.1. Purpose

The purpose of our project, named CodeKataBattle (abbreviated as CKB), is to provide a digital platform that enables students to develop and refine their skills in software development through a collaborative experience. The CKB platform allows students to practice in a programming language of their choice. The exercises offered on this platform are created by educators and require students to complete the code or parts of it, adding the missing components they deem appropriate. The code is then executed and subjected to specific tests created by educators to verify its correctness.

Exercises are organized into tournaments by educators, within which 'battles' are held to evaluate the performance of the students. This process culminates in the creation of performance rankings among all students involved in the battles and tournaments. In this way, CKB encourages friendly competition and the growth of students' skills in the field of software development.

1.1.1. Goal

[G1]   Educator creates a tournament and within it, he creates one or more battle.

[G2]   Student registers for a tournament and will receive all notifications related to it.

[G3]   Student joins a battle within a tournament they are register for and invite some friends to partecipate with them, thus creates a team for the battle.

[G4]   Student receives an evaluation for the code he pushed to their GitHub repository.

[G5]   Educator reviews all the evaluated code, and if they deem it necessary, he can modify the evaluation they have received.

[G6]   Educator closes a tournament and the platform publishes the ranking.

[G7]   Both students and educators check the update ranking of the battle and the tournament.

1.2. Scope

1.2.1. World phenomena

Phenomena events that take place in the real world and that the machine cannot observe.

[WP1].   Educator wants to create a tournament

[WP2].   Educator wants to create a battle

[WP3].   Student wants to join a tournament

[WP4].   Student contacts peers participating in the tournament to form a team

[WP5].   Students fork the GitHub repository

[WP6].   Students and Educator wants to subscribe into the platform

1.2.2. Shared phenomena

1.2.2.1.   Controlled by the machine and observable by the world

[SP1]. The platform assigns scores to groups to create a competition rank

[SP2]. Students are notified when a new tournament are creates

[SP3]. All student that are subscribed on a tournament are notified of all upcoming battles creation

[SP4]. The platform creates a GitHub repository containing the code Kata and then sends the link to all students who are members of subscribed teams

[SP5]. When the battle ends the system evaluates the student automatically

1.2.2.2. Controlled by the world and observable by the machine

[SP6]. Students join into a tournament

[SP7]. Educator creates a tournament

[SP8]. Educator creates a battle

[SP9]. Educator gives permission to other ones to create battle in the tournament

[SP10]. Group of student join into a battle

[SP11]. Groups deliver their solution to the platform

[SP12]. Students set up the automated workflow in GitHub so that informs the CKB as soon as students push a new commit.

[SP13]. Each push before the deadline triggers the CKB platform, which pulls the latest sources, analyzes them, and runs the tests on the corresponding executable to calculate and update the battle score of the team

[SP14]. At the end of the battle, optionally the educator evaluates the student by overwriting the result that the platform returned

1.3. Definitions, Acronyms, Abbreviations

1.3.1. Definitions

- Commit: Commits are the core building block units of a Git project timeline. Commits can be thought of as snapshots or milestones along the timeline of a Git project. Commits are created with the `git commit` command to capture the state of a project at that point in time.

- Fork: A fork is a new repository that shares code and visibility settings with the original "upstream" repository.

- Code kata: Code kata contains the description of a battle and the software project on which the student will have to work, including also test cases and build automation scripts

- Test Case: A test case is a singular set of actions or instructions that the Educator wants to perform to verify a specific aspect of the project pushed by the students. If the test fails, the result might be a software defect that students might haven't found.

- Upload: Upload in which the educator sends to the platform database the code kata for a specific battle.

- Repository: A Git repository is the .git/folder inside a project. This repository tracks all changes made to files in your project, building a history over time. Meaning, if you delete the .git/folder, then you delete your project's history.

- Branch: In Git, a branch is a new/separate version of the main repository. Branches allows users to work on different parts of a project without impacting the main branch. That main branch is the one seen as the default one.

- Push: The `git push` command is used to upload local repository content to a remote repository. Pushing is how you transfer commits from your local repository to a remote repo.

- Pull: The git pull command is used to fetch and download content from a remote repository and immediately update the local repository to match that content. Merging remote upstream changes into your local repository is a common task in Git-based collaboration work flows.

1.3.2. Acronyms

- <u>CKB</u>: CodeKataBattle
- <u>CK</u>: CodeKata
- <u>GH</u>: GitHub
- <u>GHA</u>: GitHub Action

1.3.3. Abbreviations

- [Gn]: the n-th goal of the system
- [WPn]: the n-th world phenomena
- [SPn]: the n-th shared phenomena
- [UCn]: the n-th use case
- [Sn]: the n-th scenario
- [Rn]: the n-th functional requirement
- Repo: Git repository

1.4. Revision history

1.5. Reference documents

This document is strictly based on:

- The specification of the RASD and DD assignment of the Software Engineering 2 course, held by professor Matteo Rossi, Elisabetta Di Nitto and Matteo Camilli at the Politecnico di Milano, A.Y 2022/2023.
- Slides of Software Engineering 2 course on WeBeep.
- Official link of CodeKata: http://codekata.com/.

1.6. Documents Structure

2. OVERALL DESCRIPTION

2.1. Product perspective

2.1.1. Scenarios

[S1].   Educator creates a new tournament

Greg and Paul are two computer science teachers who want to assess their students' programming skills. While Greg was browsing the internet, he discovered that CKB platform could be suitable software for this purpose. As a result, he decided to register on the platform and asked Paul to do the same. To complete the registration, he had to fill out a form with his personal information and received a confirmation via email once the registration was completed.

Subsequently, Greg logged into the platform by entering the email address and password used during registration on the login page. After Greg logged into the application, he created a new tournament, granting Paul permission to create challenges within the newly created tournament.

Finally, Greg asked all his students to register on the platform as students and join in the tournament he had created.

[S2].   Educator creates a battle

John has created a tournament and now wants to add a new battle to it. To do this, he has prepared a detailed description of the activities to be performed, has set up a code with some parts to be completed, established evaluation criteria, chosen the group size, and set the registration deadline. Now, John can access his profile to actually create the challenge on the platform. From the main page, he selected the tournament he just created and filled out the form with the previously prepared information. At this point, the platform will notify all the students registered for the tournament, and those interested have time until the deadline to form a group and register.

[S3].   Student joins a tournament

Ted wants to improve his programming skills, and a friend suggests trying the CKB platform. Ted trusts his friend and decides to register as a student on the platform. Now that he has created his profile, he can access the platform and start searching for a tournament to join.

From his homepage, Ted can view and search for all the available and active tournaments. In case he doesn't find a tournament that meets his needs or there are no active tournaments, the platform will inform him by sending an email when a new tournament is created.

When Ted finds a tournament suitable for his needs, all he has to do is register before the established deadline, and he can start improving his programming skills.

[S4].   Student joins a battle

Ann, along with her friends Liz and Eddie, received a notification from the CKB platform about the creation of a new battle in the tournament organized by Mr. Smith. They decided to join as a team, and Ann initiated the process. To do so, Ann logged into the platform and, from Mr. Smith's tournament page, found the newly created competition, registered, and invited her two friends to join her to form a group.

Once the registration deadline had passed, they received an email from the platform containing a link to their GitHub repository created specifically for this battle. Eddie, being the first to view and open the email, accessed the GitHub repository and, following the instructions, forked the repository and modified the GitHub Actions so that every new commit made by her or her friends would be visible to the CKB evaluation platform.

[S5]. Student push a commit in the main branch

Mary is participating in a battle, and after making a change to the project in a secondary branch and ensuring that everything works correctly, she decides to merge her branch with the main branch. After confirming that the branch merge was successful, Mary makes a commit that activates the GitHub Actions, allowing the CKB platform to evaluate the changes made.

The proposed solution is assessed positively, and this enables Mary to achieve a score of 95/100. If she wants to further improve her score, she can make additional code changes by making additional commits before the battle's deadline.

[S6]. Last battle ends and educator closes the tournament

Tom is an educator who has created a tournament on the CKB platform. He has already assigned six battles, and the seventh has just concluded.

All the students who participated in the seventh battle were automatically notified of its completion and can view a draft of their scores and the related ranking. However, Tom noticed that Alice used a highly maintainable solution by implementing some design patterns. He wishes to reward her with an extra 5 bonus points and proceeds to modify the scores. Once the interval for score modifications has ended, a new email is sent to all participants with their final scores and rankings.

At this point, the results of this battle are added to the overall results of the students in the tournament, thus affecting the general ranking.

Tom decides that seven battles are more than sufficient for this tournament and closes it, allowing the platform to automatically send an email to all registered students to inform them of the tournament's closure along with the final rankings and scores.

2.1.2. Domain class diagram

2.1.3. Statecharts

2.2. Product functions

2.3. User characteristics

2.4. Assumption, Dependencies and Contraints

2.4.1. Domain Assumption

3. SPECIFIC REQUIREMENTS

3.1. External interface

3.1.1. User interface

3.1.2. Hardware interface

3.1.3. Software interface

3.1.4. Communication interface

3.2. Functional requirements

3.2.1. Use cases diagram

3.2.2. Activity diagram

3.3. Performance requirements

3.4. Design constraints

3.4.1. Standards compliace

3.4.2. Hardware limitations

3.4.3. Any other constraint

3.5. Software system attributes

3.5.1. Reliability

3.5.2. Availability

3.5.3. Security

3.5.4. Maintainability

3.5.5. Portability

4. FORMAL ANALYSIS USING ALLOY

4.1. Signatures

4.2. Facts

5. EFFORT SPENT

6. REFERENCES