POLITECNICO
MILANO 1863

SOFTWARE ENGINEERING II

CKB – CodeKataBattle
Acceptance and Test Document

Version 1.0

Polito Attilio

Rigione Pisone Raimondo

Soricelli Francesco

February 11, 2024

Sommario

1 Tested project

1.1 Authors

- Federico Lolli
- Angelo Zangari

1.2 Repository URL

https://github.com/federico123579/LolliZangari

1.3 Documents considered

- RASD: Requirements Analysis Specification Document
- DD: Design Document
- ITD: Implementation and Testing Document

2 Installation

Backend:
In order to run and test the application, it has been followed the installation guide. It has been decided to run the application utilizing Docker Desktop, therefore it has been previously installed the recommended version through the link provided in the guide. After installing Docker, the zip folder containing all the code has been downloaded from the GitHub repository and, as indicated in the installation guide, it has been executed the following command in the code directory:
Just docker build
just docker up.
Once the installation process has been terminated, it has been started the testing of the application, running the client into a browser using Docker.

Frontend:
After verifying that the system was running, the following steps were performed: first of all, the command **just frontend-install** was used. This command was essential to prepare the frontend development environment. Through it, all dependencies necessary for the frontend to function, such as specific libraries, packages and modules, were installed. Next, to actually start the frontend, the command **just frontend-dev** was executed. This command started up a local server that hosted the frontend. Once started, the user interface was accessed via a web browser.

3 Project inspection

In order to test the functioning of the application, there have been defined some acceptance test cases, that are aimed to verify the satisfaction of the requirements that have been chosen to implement and that have been specified into the RASD document.

3.1 User: how a user can interact with the system

The following test cases are intended to verify that the interaction of the application's users with the system functions correctly. For each test case, the requirements to which it is mapped and the result obtained are specified.

Requirements	Result
• R01 – The system shall allow users to see the list of all ongoing tournaments • R20 – The system shall forbid users external to CB to see the live updated ranking • R22 – The system shall fetch candidates' solutions from GitHub	**R01 (test passed)** **PASS [0.840s] ckb_backend::integration tournaments::list** **R20 (test passed)** **Each request requires the authentication token in the headers** **R22 (not checked)**

3.2 Educator: how educators can interact with the system

The following test cases are intended to verify that the interaction of the application's educator with the system functions correctly. For each test case, the requirements to which it is mapped and the result obtained are specified.

Requirements	Result
• R04 – The system shall allow educators to log-in • R02 – The system shall allow educators to sign-up • R06 – The system shall allow educators to create tournaments • R08 – The system shall allow educators to invite other educators to create a CB in tournaments they own • R07 – The system shall allow educators to create a CB in tournaments for which they have permission	**R04 (test passed)** **PASS [1.303s]** **ckb_backend::integration login::with_credentials** **R02 (test passed)** **PASS [1.444s]** **ckb_backend::integration login::register** **R06 (test passed)** **PASS [0.866s]** **ckb_backend::integration tournaments::create**

• R09 – the system shall allow educators to grade only the CB they made/own • R10 – the system shall allow educators to manually grade CB submissions (given permission) • R11 – the system shall allow educators to upload materials (code, tests and documentation) for a CB (given permission) • R12 – the system shall allow educators to set a deadlines for CB registration (given permission) • R13 – the system shall allow educators to set a deadlines for final CB solution submissions (given permission) • R14 – the system shall allow educators to set a deadlines for tournament registration (given permission) • R15 – the system shall allow educators to set a maximum and minimum number of students per team (given permission) • R16 – the system shall allow educators to set which factors to consider for the ranking of the students in a CB (given permission)	**R08 (test passed)** **PASS [1.650s]** **ckb_backend::integration** **tournaments::add_collaborators** **R07 (test passed)** **PASS [1.578s]** **ckb_backend::integration** **battles::create** **R10 (not checked)** **R11(test passed)** **Test made with PostMan** **R12 (test passed)** **Test made with PostMan** **R13 (not checked)** **R14 (test passed)** **Test made using frontend** **R15 (test passed)** **Test made using frontend** **R16 (not checked)**

3.3 Student: how student can interact with the system

The following test cases are intended to verify that the interaction of the application's student with the system functions correctly. For each test case, the requirements to which it is mapped and the result obtained are specified.

Requirements	Result
• R05 – The system shall allow students to log-in • R03 – The system shall allow students to sign-up • R17 – The system shall allow students to invite other students to form teams • R18 – The system shall allow students (in teams) to have their solution graded • R19 – The system shall allow students to be ranked in real time during the battle • R21 – The system shall allow students to see the complete ranking of a tournament	**R05 (test passed)** **PASS [1.303s]** **ckb_backend::integration** **login::with_credentials** **R03 (test passed)** **PASS [1.444s]** **ckb_backend::integration** **login::register** **R17 (test passed)** **PASS [2.335s]** **ckb_backend::integration** **teams::invitations_and_acceptance** **R18 (not checked)** **R19 (not checked)** **R21 (test passed)** **Test made using frontend**

```
Finished test [unoptimized + debuginfo] target(s) in 30.22s
Starting 26 tests across 9 binaries (2 skipped)
    PASS [   1.150s] ckb_backend::integration login::with_expired_token
    PASS [   1.167s] ckb_backend::integration login::with_token
    PASS [   1.188s] ckb_backend::integration battles::get_factors
    PASS [   1.303s] ckb_backend::integration login::with_credentials
    PASS [   1.444s] ckb_backend::integration login::register
    PASS [   1.575s] ckb_backend::integration battles::get_info
    PASS [   1.578s] ckb_backend::integration battles::create
    PASS [   0.477s] ckb_backend::integration login::with_unauthorized_token
    PASS [   0.584s] ckb_backend::integration login::with_unverified_token
    PASS [   1.411s] ckb_backend::integration tournaments::battle_list
    PASS [   1.499s] ckb_backend::integration tournaments::add_participants
    PASS [   1.656s] ckb_backend::integration teams::team_token_and_team_login
    PASS [   1.812s] ckb_backend::integration teams::register_and_info
    PASS [   1.650s] ckb_backend::integration tournaments::add_collaborators
    PASS [   1.628s] ckb_backend::integration tournaments::close
    PASS [   2.291s] ckb_backend::integration notifications::endpoint
    PASS [   2.335s] ckb_backend::integration teams::invitations_and_acceptance
    PASS [   0.866s] ckb_backend::integration tournaments::create
    PASS [   0.937s] ckb_backend::integration tournaments::list
    PASS [   0.955s] ckb_backend::integration tournaments::get_participants
    PASS [   1.074s] ckb_backend::integration tournaments::get_info
    PASS [   0.936s] ckb_backend::integration user::get_educators
    PASS [   1.124s] ckb_backend::integration user::info_by_id
    PASS [   1.144s] ckb_backend::integration user::info_by_username
    PASS [   1.098s] ckb_backend::integration user::update_user
    PASS [   1.078s] ckb_backend::integration user::whoami
----------
  Summary [   4.986s] 26 tests run: 26 passed, 2 skipped
```

Figure 1 TEST

These are the runned tests to check the validity of the request listed in DD. Since the fact that the frontend is not completely implemented, we were not able to execute all the tests; for some tests not included in the provided ones, we used PostMan creating the requests following the indications given in the documentation. Checking the status code of the response, we were able to establish if the tests where passed or not.

4 Conclusion

In conclusion, the application was consistently implemented. Most of the requirements are met precisely, some were not tested due to the non-functional frontend. However, in general, the application has the required functionality, but in some sections it is not very intuitive to understand how some of them are performed.

However, considering that the application is a prototype, it is mostly consistent and fairly well implemented.

5 Effort Spent

Student	Hours
Student 1	*10*
Student 2	*10*
Student 3	*10*