POLITECNICO DI MILANO

Software Engineering 2

CodeKataBattle - Improve your programming skills online

Design

Document

Authors:

Nicolò Giallongo - 10764261

Giovanni Orciuolo - 10994077

Giuseppe Vitello - 10766482

1 - Introduction

1.1 - Purpose

The purpose of the CKB application is to provide students and educators with a platform to improve their programming skills by taking part in friendly competitions (called “Tournaments”) where they can resolve various programming problems (called “Katas”) in their preferred language of choice (e.g. Java or Python).

The platform relies heavily upon GitHub in order to store solutions and to run tests and other activities on each push. As such, the users are required to use their GitHub account to access the application (or register one if they don’t have it).

1.2 - Scope

In this section of the document, we will discuss the main design choices that we have taken in relation to the domain of the product.
Given that CKB is an application that aims to allow its users to compete with other users in online code competition, it seems natural to implement it as a client-server application. More specifically, we agreed on using a three-tier architecture (presentation tier, business tier and data tier) to maintain a separation between the user interface, the logic of the system and the data, and in this way improve the maintainability, the stability and the security of the system. To further improve these aspects of the CKB application we decided to use the microservice approach to develop the system. In this way we also improve the scalability of the system, allowing it to adapt more easily and efficiently to different amounts of active users.

1.3 - Definitions, Acronyms, Abbreviations

In order to reduce ambiguity as much as possible, this document heavily uses acronyms and abbreviations to refer to entities and concepts in the CKB system. This section is aimed at defining each of these acronyms and abbreviations in a precise and concise way.

Acronym	**Definition**
IU	Interested Student: a Student U is Interested by a notification N: ● if U has selected the option to receive notification from all the users. ● if U has selected the option to receive notification only from their friend and the user who has sent N is a friend of U. ● N is a notification of a Tournament in which U is subscribed.

SST	Students Subscribed to a Tournament.
OTC	Original Tournament Creator.
TC	Tournament Coordinator, appointed by the OTC.
CDT	Current DateTime.
ETD	Enrollment Tournament Deadline. After this deadline, it is not possible to join the Tournament as a Student.
FTD	Final Tournament Deadline. After this deadline, the Tournament is ended and the final leaderboard is made available.
MNS	Maximum Number of Subscribers. It is the maximum number of Students that can subscribe to a specific Tournament.
EBD	Enrollment Battle Deadline. After this deadline, it is not possible to join the Battle as a Student.
FBD	Final Battle Deadline. After this deadline, the Battle is ended and the final leaderboard is made available. Moreover, the Tournament related to the Battle is updated with the new leaderboard.
OME	Optional Manual Evaluation. Personal score assigned by the Educator, who checks and evaluates the work done by students (the higher the better).
MAE	Mandatory Automated Evaluation, which includes: ● Functional aspects, measured in terms of number of test cases that pass out of all test cases (the higher the better); ● Timeliness, measured in terms of time passed between the registration deadline and the last commit (the lower the better); ● Quality level of the sources, extracted through Static Analysis Tools that consider multiple aspects such as security, reliability, and maintainability (the higher the better). Aspects are selected by the educator at battle creation time.
DAS	Directly Accepted Student: a Student U is directly accepted in a Tournament T if: ● T is public; ● T is friends only and U is a friend of the OCT of T.
EB	End Battle: ● If OME is not required, EB occurs when the FBD is reached.

	● If OME is required, EB is after the Educator performs it, or when the FTD is reached.
SAT	Static Analysis Tools.

1.4 - Revision History

Version	Date	Changelog
1.0	03/01/2024	First draft of the document. Completed section 1, sections 2 and 3 are work in progress.

1.5 - Reference Documents

- The specification document Assignment RDD AY 2023-2024.pdf

1.6 - Document Structure

This document is divided in 5 main parts:

1. **Introduction**: introduces the main architectural and design choices made and briefly explains the motivations behind them.
2. **Architectural Design**: shows in detail the architectural decisions adopted and how they are developed and combined to create the global architecture of the system.
3. **User Interface Design**: shows the design adopted for the user interface, presenting a preview of the interface using the corresponding mockups.
4. **Requirements Traceability**: shows the mapping between the requirements defined in RASD and the design elements previously discussed.
5. **Implementation, Integration and Test Plan**: shows the order in which we plan to implement subsystems and components as well as the plan of the integration and test.
6. **Effort Spent**: contains information about the number of hours each group member has worked on this document.
7. **References**: contains information about the resources used to redact this document.

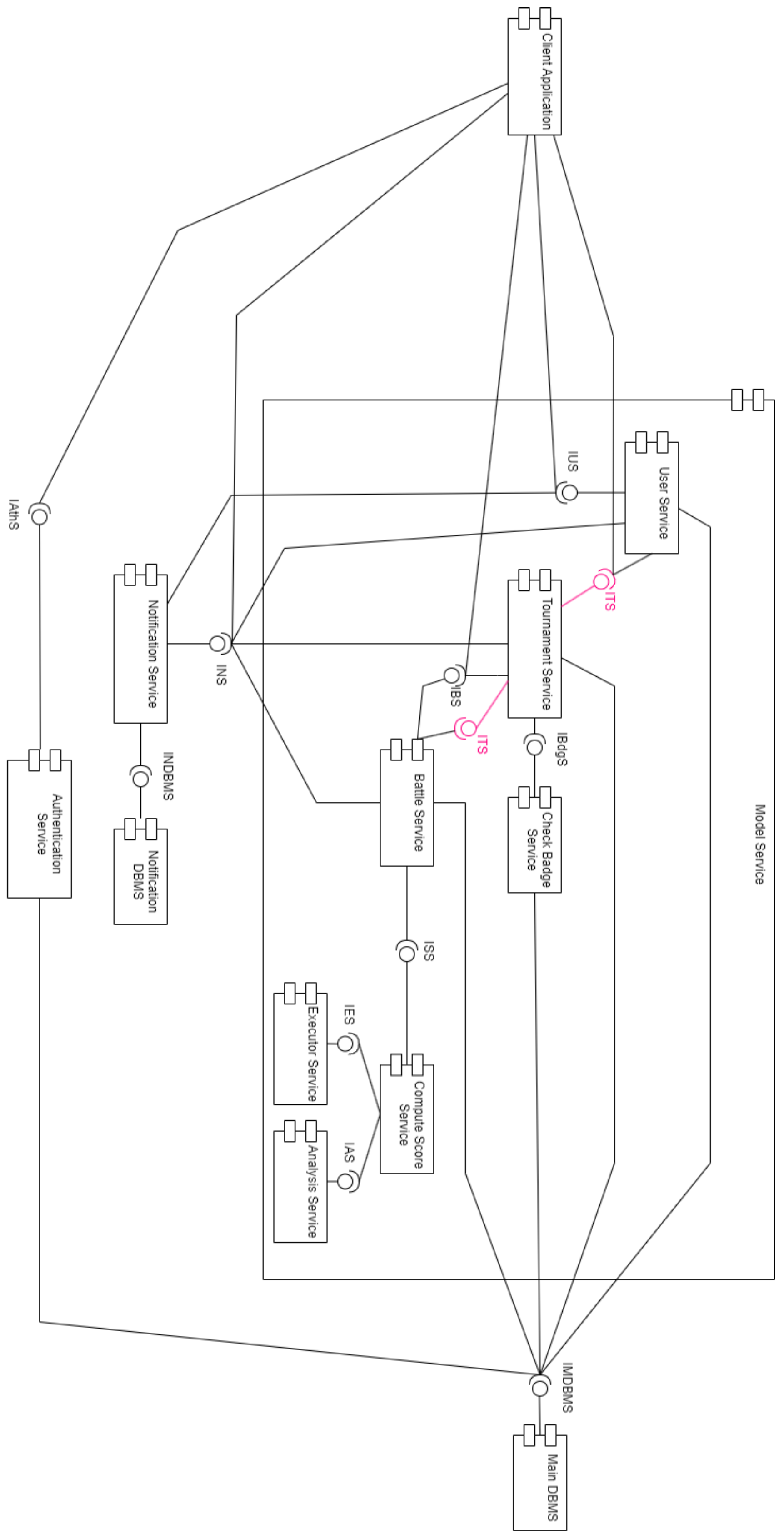
2 - Architectural Design

2.1 - Overview

- Client-Server architecture
- Microservices

2.2 - Component View

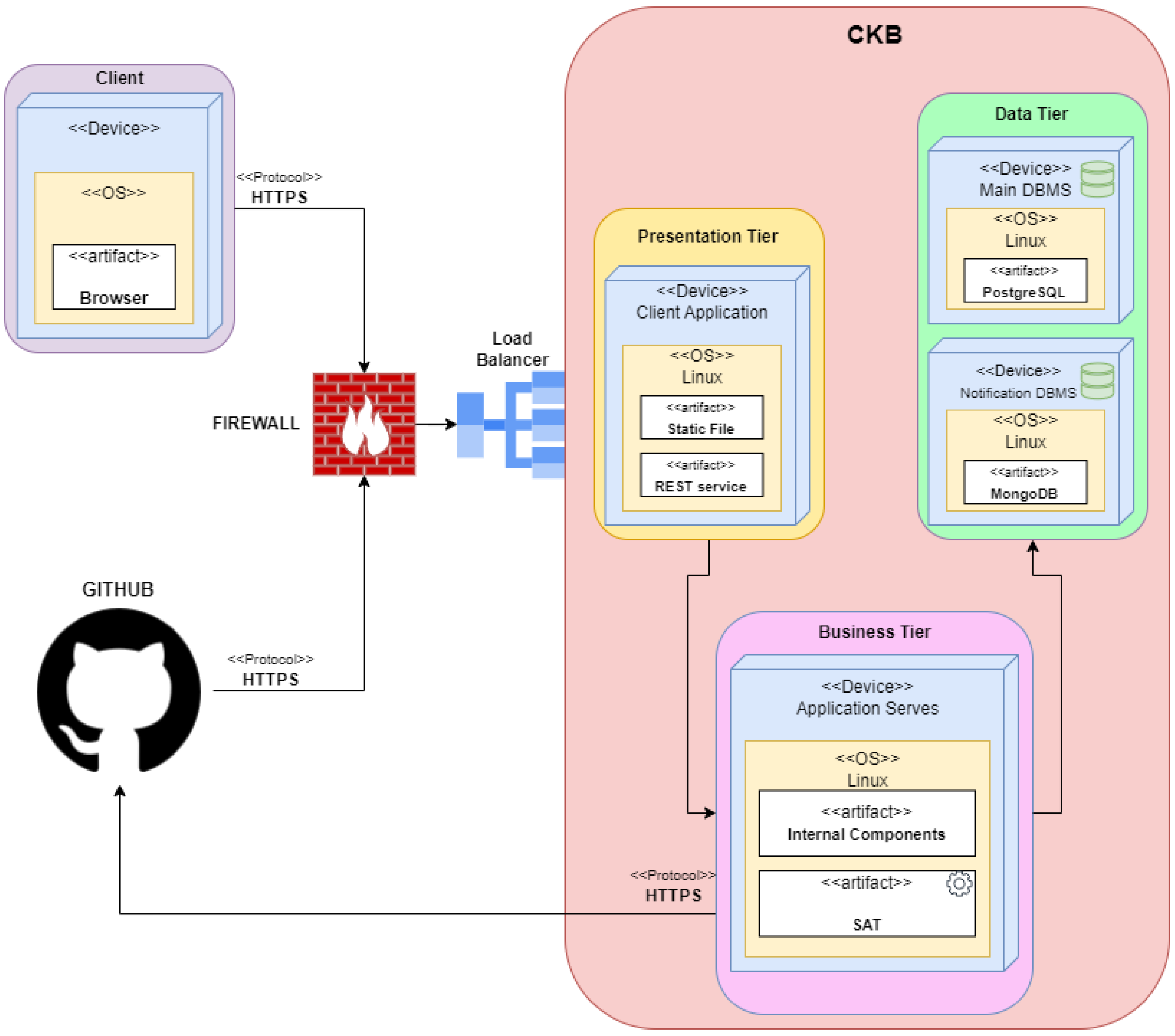
2.2.1 - Component Diagram



2.2.2 - Components Description

- **Client Application.** Represents the web application used by the final user to interface with the CKB system, so it reads the request and sends the html pages.
- **Authentication Service.** Handles authentication with GitHub following the OAuth 2.0 protocol and subsequent token verification processes.
- **User Service.** Handles registered users management, friends' list and privacy policy..
- **Tournament Service.** Handles everything related to tournaments,from creation to end of the Tournament, creation of battle.
- **Battle Service.** Handles everything related to tournament battles and their katas.
- **Executor Service.** Handles execution of code in an isolated sandbox. It is used to run kata submissions and check their output against the provided tests.
- **Analysis Service.** Handles execution of static analysis tools (SATs) in an isolated sandbox.
- **Notification Service.** Abstracts the process of delivering and storing notifications.
- **Main DBMS:** Holds and manages data about Tournaments, Battles, Users, Badges and the relationships between them. The DBMS of choice is PostgreSQL.
- **Notification DBMS:** Holds and manages data about notifications. The DBMS of choice will be a NoSQL solution (Redis).

2.3 - Deployment View



Presentation Tier:
Business Tier:
Data Tier:
Client:
GitHub:
Firewall:
LoadBalancer:

2.4 - Component Interfaces

2.5 - Runtime View

2.6 - Selected Architectural Styles and Patterns

2.7 - Other Design Decisions

ERD: https://editor.ponyorm.com/user/giovanni_or2/CodeKataBattle

3 - User Interface Design

This section is dedicated to presenting a general overview of the user interface of the CodeKataBattle system, as well as showing some mockups mainly focused on UX rather than UI, since that aspect will be handled by focus groups.

3.1 - Overview

```
stateDiagram-v2
    home: Home Page
    login: Login Page
    tournament_list: Tournaments List Page
    tournament_create: Tournament Create Page
    tournament_details: Tournament Details Page
    tournament_edit: Tournament Edit Page
    battle_details: Battle Details Page
    battle_edit: Battle Edit Page
    battle_create: Battle Create Page
    kata_submission: Kata Submissions Page
    user_profile: User Profile Page
    edit_profile: Edit User Profile Page
    notifications: Notifications Page
    friends_list: Friends List Page
    home --> login: if not logged in
    home --> tournament_list
    home --> user_profile: if logged in
    home --> notifications: if logged in
    home --> tournament_create
    tournament_list --> tournament_details
    tournament_details --> battle_details
        tournament_details  -->  tournament_edit:  if  logged  user
   is\ncreator or coordinator
```

```
    tournament_details --> battle_create: if logged user is\ncreator
  or coordinator
   battle_details --> battle_edit: if logged user is creator
   battle_details --> kata_submission
   user_profile --> edit_profile
   user_profile --> friends_list
```

3.2 - Mockups

https://www.figma.com/file/T3paTQUDTwdw49rkhTvXBL/Frontend-Mockups?type=design&node-id=0%3A1&mode=design&t=t3KtYBauaHz5l9eE-1

4 - Requirements Traceability

5 - Implementation, Integration and Test Plan

6 - Effort Spent

Group Member	Effort Spent (hours)
Nicolò Giallongo	Introduction: 3h Architectural Design: 6h User Interface Design: Requirements Traceability: Implementation, Integration and Test Plan:
Giovanni Orciuolo	Introduction: 1h Architectural Design: 5h User Interface Design: 4h Requirements Traceability: Implementation, Integration and Test Plan:
Giuseppe Vitello	Introduction: 1h Architectural Design: 6h User Interface Design: Requirements Traceability: Implementation, Integration and Test Plan:

7 - References