University of Pisa

Artificial Intelligence and Data Engineering

Distributed Systems and Middleware Technologies

FLconsole documentation

Authors: Çolak F. Messina F. Nocella F.

Academic Year 2023/2024

Contents

Introduction and Project Overview

Context and Project Objective

In this section, the context and objective of the project are described.

Application Highlights

In this section, the highlights of the application are presented.

Analysis

Requirements

In this section, the requirements of the project are presented.

Functional Requirements

The functional requirements of the project are described in this subsection.

Non-Functional Requirements

The non-functional requirements of the project are described in this subsection.

Constraints/Other Requirements

Any constraints or other requirements on the project are described in this subsection.

Actors

The actors who can interact with the web console system consist of the following:

- **User:** The user is the actor who can browse the system to view running and completed experiments and their results.
- **Admin:** The admin is the actor who can manage the system, including creating and deleting configurations and experiments, and viewing the results of experiments.

Use Case Modeling

Use Case Diagram

Scenarios

Analysis Class Diagram

The analysis class diagram of the project is presented in this subsection.

Sequence Diagrams

The sequence diagrams of the project are presented in this subsection.

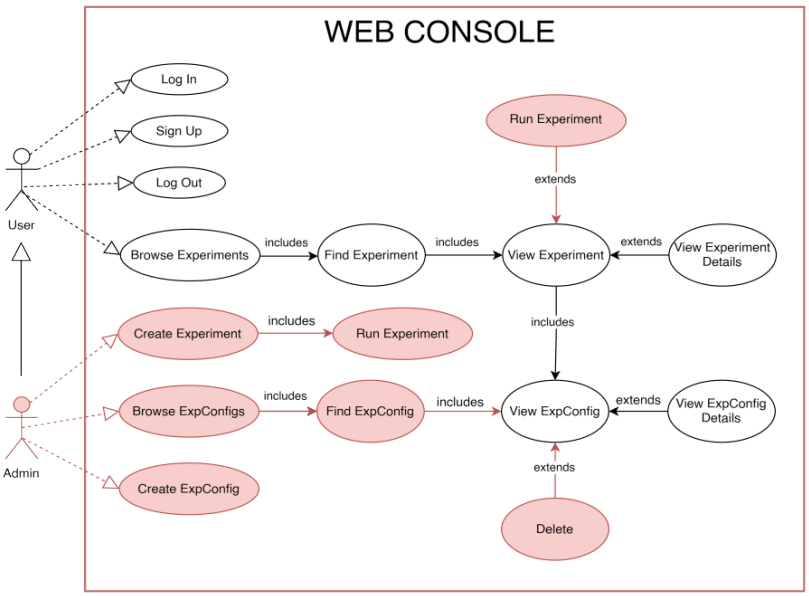

Figure 2.1: Use Case Diagram

Table 2.1: Use Case: Find Recipes

Use Case	**Find Recipes**
Primary Actor	User
Secondary Actor	-
Description	Allows the user to find a specific recipe
Pre-Conditions	User must be logged in
Main event steps	1. The user navigates to the "Search" feature 2. The user enters a string 3. The system searches the recipe database for matching results (title, keywords, description) 4. Upon finding matching posts, the system displays the search results
Post-Conditions	The user views a list of recipes matching the search criteria if there are any
Correlated Use cases	View Recipe, Browse Recipes
Alternative event steps	-

Design

Software Architecture

The software architecture of the project is described in this section.

Database Design

In this section, the database design of the project is presented.

MongoDB

Collections

ExpConfig document example:

```
{
    "id": "example_id",
    "name": "Example Experiment",
    "algorithm": "example_algorithm",
    "strategy": "example_strategy",
    "numClients": 10,
    "stopCondition": "example_condition",
    "threshold": 0.5,
    "parameters": {
        "param1": "value1",
        "param2": "value2",
        "param3": "value3"
    },
    "creationDate": "2024-03-14T00:00:00Z",
    "lastUpdate": "2024-03-14T12:00:00Z"
}
```

Experiment document example:

```
{
    "id": "example_id",
    "name": "Example Experiment",
    "expConfigSummary": {
        "id": "exp_config_id",
        "name": "Example Configuration",
        "algorithm": "example_algorithm"
    },
    "creationDate": "2024-03-14T00:00:00Z",
    "lastUpdate": "2024-03-14T12:00:00Z",
    "progressList": [
        {
        "creationDate": "2024-03-14T06:00:00Z",
        "parameters": {
            "param1": "value1",
            "param2": "value2"
        },
        "status": "In progress"
        },
        {
        "creationDate": "2024-03-14T09:00:00Z",
        "parameters": {
            "param1": "value1",
            "param2": "value2",
            "param3": "value3"
        },
        "status": "Completed"
        }
    ]
}
```

User document example:


```
{
"id": "example_user_id",
"email": "user@example.com",
"password": "example_password",
"creationDate": "2024-03-14T00:00:00Z",
"configurations": ["config_id1", "config_id2"],
"experiments": [
    {
    "id": "experiment_summary_id1",
    "name": "Experiment 1",
    "configName": "Configuration 1",
    "creationDate": "2024-03-14T06:00:00Z",
    },
    {
    "id": "experiment_summary_id2",
    "name": "Experiment 2",
    "configName": "Configuration 2",
    "creationDate": "2024-03-14T09:00:00Z",
    "lastUpdate": "2024-03-14T12:00:00Z"
    }
],
"role": "example_role"
}
```

Erlang Message Handler

The Erlang message handler design is described in this subsection.

Message structure

- Error message

```
{
    "type": "error",
    "cause": "error_in_collecting_data",
    "timestamp": "2024-03-13T12:34:56"
}
```

- Stop message

```
{
    "type": "stop",
    "cause": "experiment_finished",
    "timestamp": "2024-03-13T12:34:56"
}
```

- Data message

```
{
    "type": "data",
    "parameters": {
        "param1": "value1",
        "param2": "value2"
    },
    "timestamp": "2024-03-13T12:34:56",
    "status": "running"
}
```

Implementation

Development Environment

The development environment used for the project is described in this section.

Main Modules

The main modules of the project are described in this section.

Configuration

The configuration of the project is described in this section.

Data Access

The data access layer of the project is described in this section.

Data Transfer

The data transfer mechanisms used in the project are described in this section.

Service

The services provided by the project are described in this section.

User Interface

The user interface of the project is described in this section.

Adopted Patterns and Techniques

The patterns and techniques adopted in the project are described in this section.

Testing

Structural Testing

The structural testing performed on the project is described in this section.

JUnit Testing

The JUnit testing performed on the project is described in this section.

Functional Testing

The functional testing performed on the project is described in this section.

Test Cases

The test cases used for the project are presented in this section.

Conclusion

In this chapter, we summarize the key points of the document and discuss possible future directions for the project.