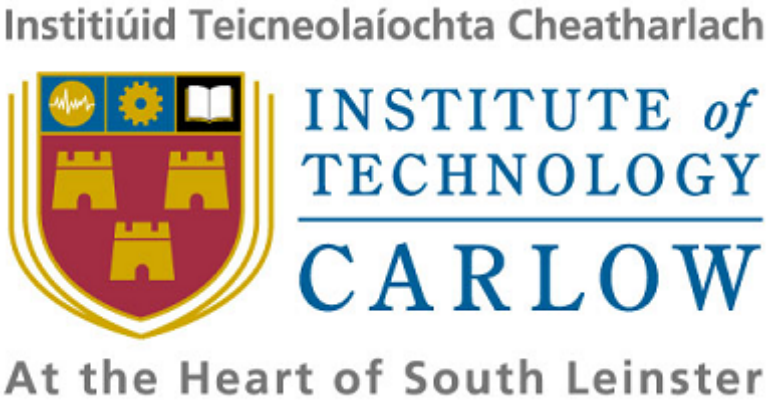


Functional Specification:

Code Editing in the Cloud

Author:

Alejandro Borrego Delgado (C00132731)

Final Project,

4th Bach (Honours) in Software Engineering

IT Carlow, 2009/2010

INDEX:

1. Vision:

1.1. Introduction

The aim of this project is to build a code editor in the cloud, with the ability of generating code in the cloud, and export as source code, or compiled program. As it works on the cloud, the only program that needs to be installed in local machines is a web browser.

1.2. Positioning

1.2.1. Problem Statement

The problem of	needing many tools to generate a software project, maintaining it and controlling it version
Affects	Software engineers and developers
The impact of which is	Reducing productivity of software engineers
A successful solution would be	Building a code editor in the cloud capable of generating code, controlling versions and export projects.

1.2.2. Product Position Statement

For	software engineers
who	works on teams and need to keep the last version updates.
other products like Bespin or CodeMirror	provides partial solutions to this problem.
our product	would have compiling capabilities improving mentioned products

1.3. Stakeholder Descriptions

1.3.1. Stakeholder Summary

Name	Description	Responsibilities
Project Manager	Is the person in charged of the project.	Editing and managing files. Editing and managing projects. Editing and managing users.
Developer	Is any of the developers involved in the project.	Editing and managing files.

1.4. Product Overview

1.4.1. Needs and features

Needs	Priority	Features
Managing Project and Files	Critical	The system will be able to create and edit Projects and files
Managing Users	Critical	The system will be able to create and manage Users
Exporting and Compilation Functionalities	High	The system should be able to Compile (generate .class or .jar files) and export this files or the source code
Log in system	Medium	Users of the system should need to log in to access to its functionalities

1.4.2. Other Product Requirements

Requirements	Priority
Version control system	Medium
Code Highlighting	Medium
Automatically save files while editing	Low
Automatically Log out inactive users	Low

2. System-Wide Requirements Specification

2.1. Introduction

Our system would be able to manage source code for software projects developed in JAVA. It has to provide create and editing projects and files functions as well as exporting project and files functionalities. It has also to manage users and their rights.

2.2. System-Wide Functional Requirements

The system must have the following functionalities:

- Create, read, update and delete (since now CRUD) software projects.
- Exporting source code of full projects.
- Creating and exporting .jar applications for full projects.
- CRUD source files inside a project.
- Exporting source code for single files.
- Creating and exporting .class files for single classes.
- Managing users (creating them and assigning rights).
- Allow users to log in the system.

2.3. Supplementary Specification

2.3.1. Functionality

The system should be able to automatically highlight code while typing.

The system should provide a version controlling system for files in a project.

Logging and Error Handling: The system would not provide an error logging system, is not needed.

2.3.2. Usability

Human Factors: The system should be stored on a server (on the cloud). Every machine connected to the network that the system is connected should only need a web browser to access to the application. The system should be seen properly in most common web browsers.

2.3.3. Reliability

Recoverability: If connection failed during editing a file. An automatically saved version of the file should be provided.

A locked file should be unlocked automatically after a time of no activity from the user who has locked the file.

2.3.4. Performance

The system should be able to run in most common browser in a not bad speed

2.3.5. Supportability

Adaptability: Code highlighting for other programming would not be provided in this version of the system, nevertheless the system should be built to allow this feature with no major changes.

Configurability: The system should provide the project manager with configuration functions for projects.

Implementation Constraints: The system should be implemented to be stored on the cloud. The chosen programming language and framework (Ruby on Rails) would allow that.

2.4. System Interfaces

2.4.1. User Interfaces

The software should be build in the cloud. Interfaces would be provided in HTML with JavaScript embedded on it (AJAX), so any browser that is able to interpret JavaScript language, should be able to display it.

2.5. Other System Constraints

The system would not be able to execute or debug a application. This decision have been taken to guarantee security.

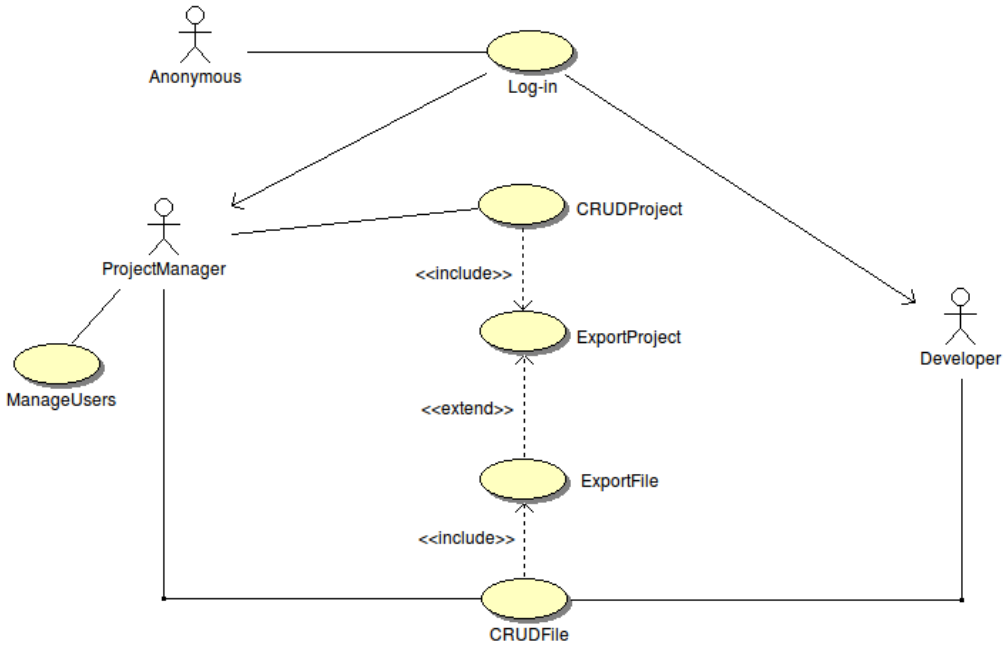
The system would not allow multiple editing on the same file. Is not a very necessary feature, current ways or merging documents are not perfect and mistakes on already finished code could show. As an extra, its implementation would make a much more harder design and implementation.

2.6. Required Documentation

Following documents would be submitted as project documentation:

- Research Manual
- Project Plan
- Functional Specification
- Design Manual
- User Manual
- Project Report

3. Uses Cases:


Anonymous
Log-in
CRUDProject
ProjectManager
<<include>>
ExportProject
Developer
ManageUsers
<<extend>>
ExportFile
<<include>>
CRUDFile

3.1. CRUD Project

3.1.1. Brief Description

The use case begins when Project Manager selects an operation between Creates, Read, Update and Delete Project. Depending on the operation selected, a new project will be created, or an existing project would be edited, showed or removed from the system. During this use case the project could be exported (as a .zip file including the source code or a .jar file).

3.1.2. Actor Brief Descriptions

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

3.1.3. Basic Flow of Events (Update a Project)

1. The use case begins when Project Manager request editing a project.
2. The system show the projects available for the Project Manager.
3. Project Manager selects the project to edit and summit.
4. The system show the editing view.
5. The Project Manager changes the info about the project.
6. The Project Manager close the view.
7. The system go back to the main view.
8. The use case ends.

3.1.4. Alternative Flows

If in step 1 of the basic flow the Project Manager selects create operation, then

1. The system requests a name for the project.
2. The system creates the project.
3. The use case resumes at step 4.

If in step 1 of the basic flow the Project Manager selects remove operation, then

1. The system show the projects available for the Project Manager.
2. Project Manager selects the project to be removed and summit.
3. The system ask for confirmation.
4. The Project Manager accepts / cancels.
5. The system removes every file related to the project,
6. The use case resumes at step 7

If in step 3 of the basic flow or in step 2 of alternative 2 the Project Manager introduces a wrong project, then

1. The system shows an error message.
2. The use case resumes at step 7.

If in step 5 of the basic flow the Project Manager try to make a forbidden operation, then

1. The system shows an error message.
2. The use case resumes at step 5.

3.2. CRUD File

3.2.1. Brief Description

The use case begins when Project Manager or Developer selects an operation between Creates, Read, Update and Delete File. Depending on the operation selected, a new file will be created, or an existing file would be edited, showed or removed from the system. During this use case the files could be exported (as a source file or a .class file).

3.2.2. Actor Brief Descriptions

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

Developer: Is any of the developers involved in the project. Has rights of editing and managing files, but not the project.

3.2.3. Basic Flow of Events (Update a File)

1. The use case begins when Project Manager or Developer request editing a file.
2. The system show the files editable by the Project Manager/Developer.
3. Project Manager/Developer selects the file to edit and summit.
4. The system locks the selected file.
5. The system show the editing view.
6. The Project Manager/Developer start editing
7. The Project Manager/Developer finishes his edition
8. The system ask if changes may be saved.
9. The Project Manager/Developer accepts / cancels.
10. The system go back to the main view.
11. The use case ends.

3.2.4. Alternative Flows

If in step 1 of the basic flow the Project Manager/Developer selects create operation, then

1. The system requests a name for the file.
2. The system creates the file.
3. The use case resumes at step 4.

If in step 1 of the basic flow the Project Manager/Developer selects remove operation, then

1. The system show the files available for the Project Manager/Developer.
2. Project Manager/Developer selects the file to remove and summit.
3. The system ask for confirmation.
4. The Project Manager/Developer accepts / cancels.
5. The use case resumes at step 10

If in step 3 of the basic flow or in step 2 of alternative 2 the Project Manager / developer introduces a wrong file, then

1. The system shows an error message.
2. The use case resumes at step 10.

If in step 5 of the basic flow the Project Manager/Developer try to make a forbidden operation, then

1. The system shows an error message.
2. The use case resumes at step 5.

3.3. Export Project

3.3.1. Brief Description

The use case begins when inside CRUD Project use case, the Project Manager press the export button. The system generates a .jar or .zip file that would be save to Project Manager machine.

3.3.2. Actor Brief Descriptions

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

3.3.3. Basic Flow of Events

1. The use case begins when inside CRUD Project use case, the Project Manager press the export button
2. The user selects, the kind of file to export (.jar, .zip)
3. The system ask for confirmation
4. The User Accepts / Cancels
5. The system executes use case Export File.
6. The system Generate a file.
7. The system display the generated file.
8. The use case ends.

3.3.4. Alternative Flows

If in step 5 or 6 of the basic flow there is an error or failure, then

1. The system generate an error file.
2. The system display the error file.
3. The use case resumes at step 8.

3.4. Export File

3.4.1. Brief Description

The use case begins when inside CRUD File use case, the Project Manager or Developer press the export button. The system generates a .class or .java file that would be save to Project Manager or Developer machine.

3.4.2. Actor Brief Descriptions

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

Developer: Is any of the developers involved in the project. Has rights of editing and managing files, but not the project.

3.4.3. Basic Flow of Events

1. The use case begins when inside CRUD File use case, the Project Manager or Developer press the export button
2. The Project Manager / Developer selects, the kind of file to export (.java, .class)
3. The system ask for confirmation
4. The Project Manager / Developer Accepts / Cancels
5. The system Generate a file
6. The system display the generated file
7. The use case ends.

3.4.4. Alternative Flows

If in step 5 of the basic flow there is an error or failure, then

1. The system generate an error file
2. The system display the error file
3. The use case resumes at step 7

3.5. Manage users

3.5.1. Brief Description

The use case begins when Project Manager selects an operation to do with the users between creating deleting and assign rights. Depending on the operation selected, a new Developer/Project Manager will be created, or an Developer/Project Manager would be edited or removed from the system.

3.5.2. Actor Brief Descriptions

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

3.5.3. Basic Flow of Events (Assigning rights to a user)

1. The use case begins when Project Manager request editing an user.
2. The system show the users available for the Project Manager.
3. Project Manager selects the user to edit and summit.
4. The system show the editing view.
5. The Project Manager assign rights to the user.
6. The Project Manager close the view.
7. The system go back to the main view.
8. The use case ends.

3.5.4. Alternative Flows

If in step 1 of the basic flow the Project Manager selects create operation, then

1. The system requests an ID for the user.
2. The system creates the user.
3. The use case resumes at step 4.

If in step 1 of the basic flow the Project Manager selects remove operation, then

1. The system show the users available for the Project Manager.
2. Project Manager selects the user to be removed and summit.
3. The system ask for confirmation.
4. The Project Manager accepts / cancels.
5. The use case resumes at step 7

If in step 3 of the basic flow or in step 2 of alternative 2 the Project Manager introduces a wrong user, then

1. The system shows an error message.
2. The use case resumes at step 7.

If in step 5 of the basic flow the Project Manager try to make a forbidden operation, then

1. The system shows an error message.
2. The use case resumes at step 5.

3.6. Log-in

3.6.1. Brief Description

The use case begins when an anonymous user selects log-in operation on the system. If the log-in is successful the anonymous user becomes Developer or Project Manager, and the main view would be showed.

3.6.2. Actor Brief Descriptions

Anonymous: Is every person that try to access to the system.

Project Manager: Is the person in charged of the project. Not only has rights for editing and managing files, but also editing and managing projects and users.

Developer: Is any of the developers involved in the project. Has rights of editing and managing files, but not the project.

3.6.3. Basic Flow of Events

1. The use case begins when an anonymous user selects log-in operation on the system.
2. The system check for the giving details about the user in the db.
3. Anonymous become a Developer or a Project Manager.
4. The system go back to the main view.
5. The use case ends.

3.6.4. Alternative Flows

If in step 2 of the basic flow the system detects a wrong ID or password, then

3. The system shows an error message.
4. The use case resumes at step 5.