Applied software engineering

Hamza Bhatti (21223241)

Contents

[Introduction 2](#_Toc468283802)

[Requirements From Case Study 2](#_Toc468283803)

[UML Modelling 2](#_Toc468283804)

[Use Case Description 2](#_Toc468283805)

[Class Diagram 2](#_Toc468283806)

[Sequence Diagram 2](#_Toc468283807)

[Architecture Diagram 3](#_Toc468283808)

[Implementation 4](#_Toc468283809)

[Software Testing 5](#_Toc468283810)

[Evaluation 6](#_Toc468283811)

[UML 6](#_Toc468283812)

[Implementation 6](#_Toc468283813)

[Software Testing 6](#_Toc468283814)

[Appendix 7](#_Toc468283815)

# Introduction

For this assignment a case study was provided. Appropriate UML, implementation in Java, along with testing was produced based on one chosen use case. The report will also cover critical analysis of all the previously mentioned elements of the assignment.

# Requirements From Case Study

What is the chosen use case and what is required from the case study?

The use case that was chosen to be implemented was “Update site popularity” which also included “Prioritise site for marketing”. A set of requirements were found within the case study which had to be followed to ensure that the implementation would be successful.

The requirements understood where:

1. The check will occur on the 30th December every year.
2. 6 regions are currently present, all of which have sites.
3. Site have ratings which are Bronze, Silver and Gold.
4. Site visitors will determine the new rating. If visitors are below 10,000 a Bronze is given, if between 10,000 and 30,000 a Silver is given and when above 30,000 a Gold is given.
5. While the rating is given, the site will be checked if it needs a marketing campaign.

For use case to be successful, these requirements had to be adhered to in the UML, Java implementation and had to be tested.

# UML Modelling

To aid in implementing the requirements of the use case, a series of UML diagrams where created. The diagrams provide a description of what should occur, the components that should be present, their interactions and sequence of events that take place to meet the requirements.

## Use Case Description

\* Appendix \_ Use case diagram

Talk about the functionality of the use case diagram – What’s going to happen in the software

The use cases that are dealt with in the diagram itself are the “Update site popularity” which include “Prioritise site for marketing”. The actor for these use cases is the System itself as there is not outside interaction by a physical user.

The use case description produced described how the system that would implemented will function. It covered the goal, pre-conditions, post-condition and how the use cases would be triggered along with the steps needed to reach the goal.

The requirements that had been identified in the previous section of the report are embedded in the whole use case diagram. One requirement was that the date needed to be 30th of December for the popularity to be changed. In the use case description, it is described that the date will be checked against a target. If the target is reached, the use case could carry on, otherwise nothing would happen.

Another requirement was that the popularity ratings would be determined by the amount of visitors of a site. The description states that the rating will change to either Bronze, Silver or Gold depending on if the visitor count meets a certain threshold.

Finally, the check to see if the marketing is needed for a site is identified in the description. This occurs before the “Update site popularity” use case has been completed. The process itself checks if the site needs marketing based on if the site gets half of the Bronze rating threshold.

## Class Diagram

\* Appendix \_ Class diagram

What are the elements in the class diagram – classes (attributes, operations), associations, pattern and abstraction?

To ensure that the use case and requirements could be met in the implementation, an appropriate class diagram had to be produced. The class diagram provided contains the different classes, their attributes, operations and their associations with one another. A design pattern is also provided here, which will be further described later in this report.

The classes created were based on the requirements. The SaxonSystem class was made as the business logic would be running through it. This being the two use cases. The Region class was created to hold number of sites. The Site class was created with operations that would help retrieve the amount of visitors and set a new rating. Again, these operations would be accessed through the system and not directly used anywhere else in the program.

All classes presented have interfaces which act as contracts. This makes the diagram itself abstract. This is the case because classes that implement the interfaces only show what attributes are needed for the implementation. The contract themselves have no bearing on the implementation, but show the method headings that need to be used to achieve the use cases. Having the contracts themselves would help with reusability and maintainability, but this will be discussed later in the report.

The abstraction of the diagram helps remove clutter, such as extra classes that do not need to be there. It also only shows the attributes and operations that are needed to fulfil the goal of the use case. This removes any confusion and leaves that system open and less rigid for any type of implementation for future development.

## Sequence Diagram

What functions where used? – Just list them I guess.

Behaviour of the system – Talk about what happens at each step.

## Architecture Diagram

Talk about the sections of the architecture and how it fits with the other UML.

# Implementation

Talk about how the system meets the use case requirements

Features of the code – interfaces, abstraction how classes interact, the design pattern (why is it there?)

Quick instruction on how to run the program – what is included with the package

# Software Testing

What is the purpose of testing?

Talk about the tests you did - How did they help meet the requirements?

Explain what happens during the tests to get the result.

Give instructions on how to run

# Evaluation

## UML

Why was the UML abstract in certain places – in the use case diagram and class diagrams?

How can the UML change to be better?

## Implementation

How did the code follow the uml – is the code abstract too?

How is the code reusable etc? – Software principles along with maintainability etc

How could the code change to be better?

## Software Testing

How did the tests help?

Why were the tests written the way they were?

How could the tests change to be better?

# Appendix

PUT DIAGRAM AND CODE HERE