Cooking Recipe Rating Web based Application

Analysis and Design Document

Student:Andreea Sabina Lazaroiu

**Group:30432**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <05/04/18> | <1.0> | First Version | Andreea Sabina Lazaroiu |
| <16/05/18> | <2.0> | Second Version | Andreea Sabina Lazaroiu |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 5

2.3 Component and Deployment Diagrams 6

III. Elaboration – Iteration 1.2 7

1. Design Model 7

1.1 Dynamic Behavior 7

1.2 Class Design 7

2. Data Model 10

3. Unit Testing 10

IV. Elaboration – Iteration 2 10

1. Architectural Design Refinement 10

2. Design Model Refinement 10

V. Construction and Transition 10

1. System Testing 10

2. Future improvements 10

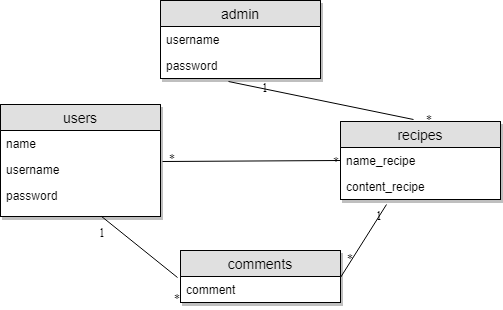
VI. Bibliography 10

# Project Specification

The application consists of a website of cooking recipes. Users can create an account to this website and log in. They can also post, view recipes, comment on a recipe and view comments of a recipe. This website will be managed by the administrator which will also have to login. The administrator can view users, view recipes, delete incorrect recipes and delete users.

# Elaboration – Iteration 1.1

# Domain Model

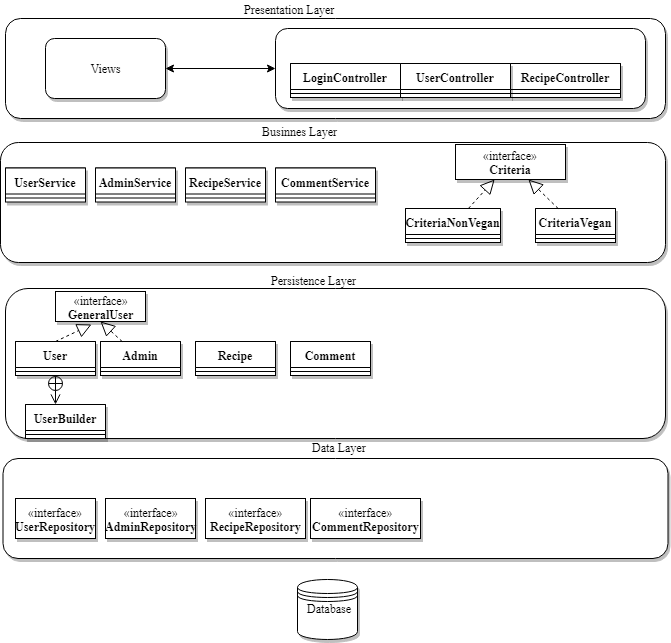


# Architectural Design

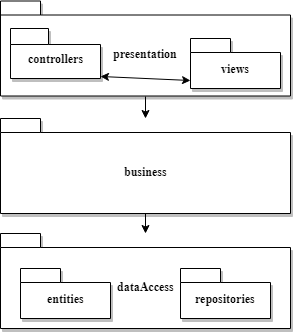
## Conceptual Architecture

This system will be implemented based on the Layered Architecture Pattern. This pattern is recommended for the applications that can be decomposed into different groups of subtasks which will be the case for this application. Also with this pattern parts of the system should be exchangeable. This pattern permits future late changes to not ripple through the system. This pattern provides high testability and high ease of development, low coupling and high cohesion.

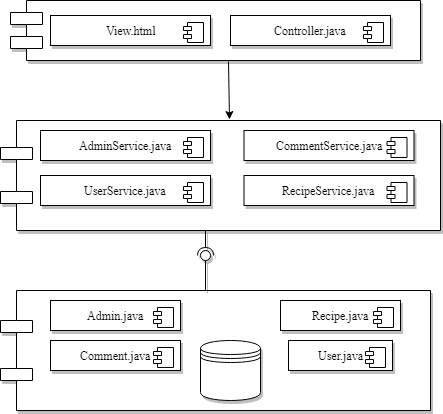
The system will also have an MVC Pattern applied to it in the presentation layer.

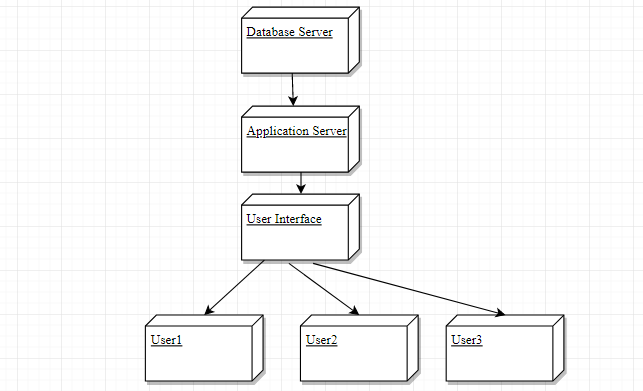


## Package Design



## Component and Deployment Diagrams





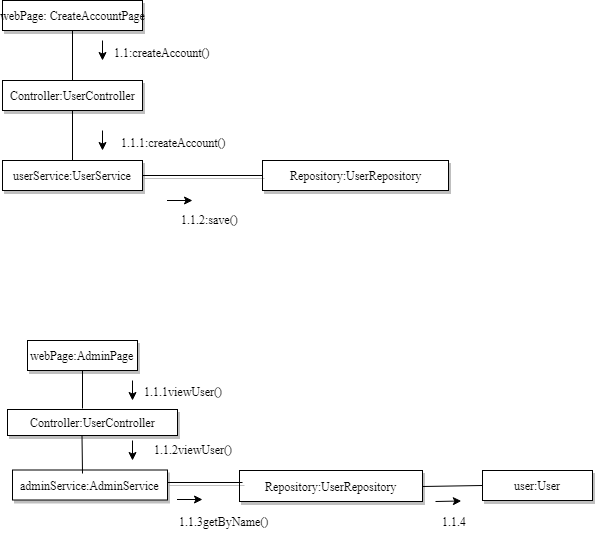
# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

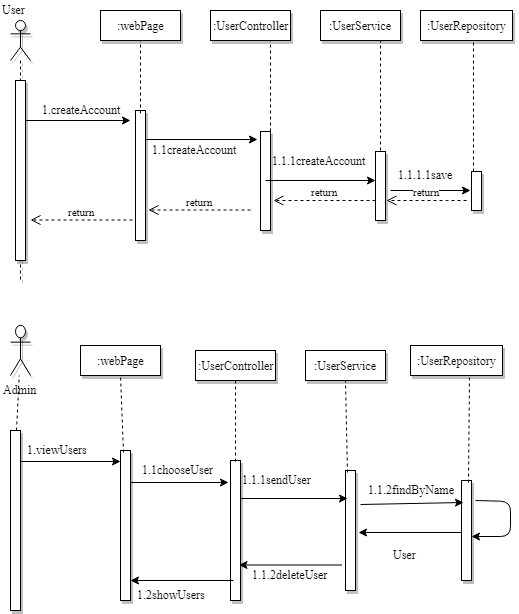
Communication Diagrams

* User: register
* Admin: view user



Sequence Diagrams:

* User: create Account
* Admin: delete user



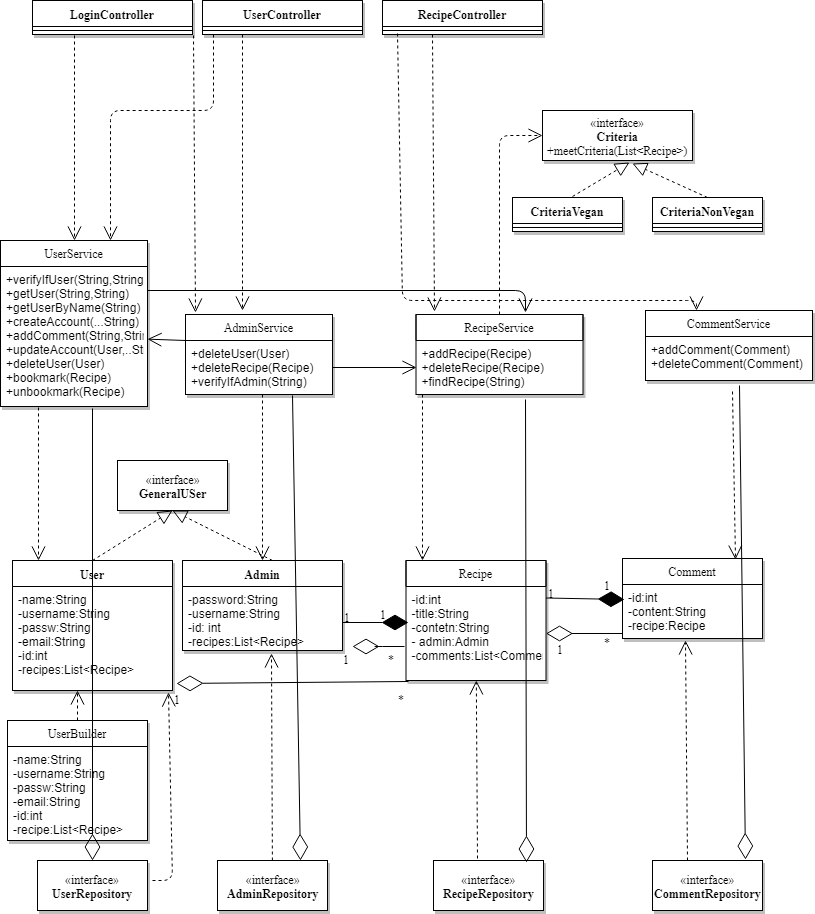
## Class Design

Design Patterns Used: Factory Pattern, Builder Pattern and Filter Pattern.

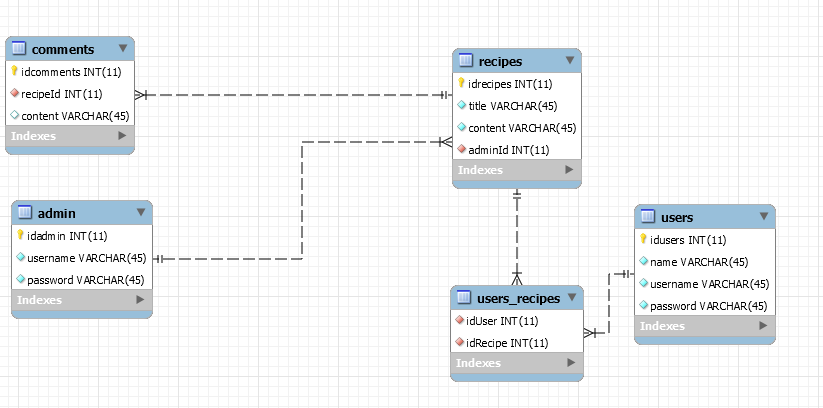
For this project the Factory Pattern will be used as follows: the idea is to define an interface to construct an object and let each subclass decide what class to instantiate.

The Builder Pattern is used to create object of type User in our case because it has many attributes.

The Filter Pattern is implemented in the business layer when we want to filter the vegan recipes from the non-vegan one.



# Data Model

**

# Unit Testing

There will be performed JUnit Test methods for the most relevant scenarios.

# Elaboration – Iteration 2

# Architectural Design Refinement

The package design has been lightly modified by putting all the Controller Classes from the presentation package into a sub package named controllers.

The main conceptual architecture has not been altered and remained the same as before. It has been slightly refined visually.

# Design Model Refinement

The Class Diagram has been modified and is presented before this chapter. It has been tried to implement Low Coupling and High Cohesion as good as possible.

# Construction and Transition

# System Testing

Test methods have been performed for the following test case scenarios using JUnit Test methods:

* View Recipes
* Add Recipe
* Delete Recipe

# Future improvements

The application will present along textual recipes also visual recipes, meaning images and videos.

The Users will be capable to send messages among each other on this platform.

# Bibliography