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 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# 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 ratings of a recipe. This website will be managed by the administrator which will also have to login. The administrator can add keywords, view users, view incorrect recipes, delete incorrect recipes and block users.

# Elaboration – Iteration 1.1

# Domain Model

*[Define the domain model and create the conceptual class diagrams]*

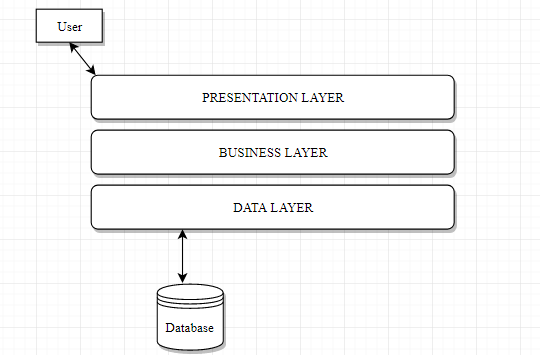
Classes: AdminDao, UserDao, RecipeDao,RegisterDao ,Admin,Recipe, User,Register, AdminService, UserService, RecipeService, RegisterService.

Interface: DaoInterface

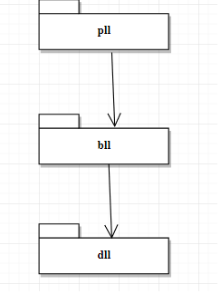
# Architectural Design

## Conceptual Architecture

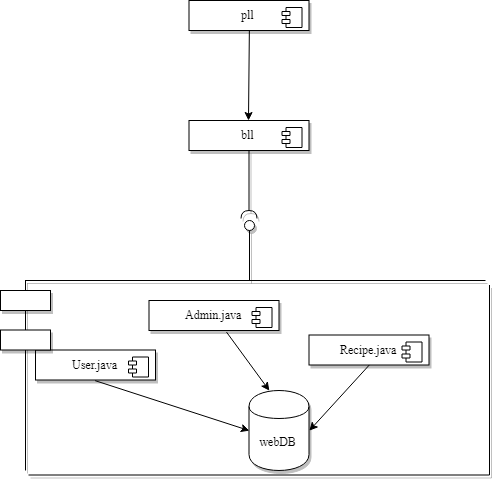
This system will be implemented based on the Layered Architecture Pattern. This pattern is recommended for the applications that can be decomposed into different groups of subtasks which will be the case for this application. Also with this pattern parts of the system should be exchangeable. This pattern 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

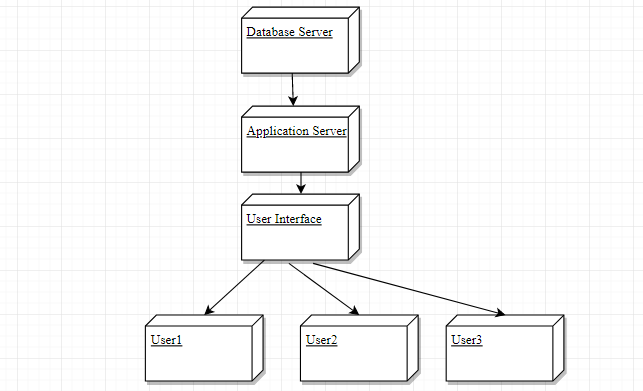


## Package Design



## Component and Deployment Diagrams





# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

*[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]*

## Class Design

*[Create the UML class diagram; apply GoF patterns and motivate your choice]*

# Data Model

*[Create the data model for the system.]*

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

# Construction and Transition

# System Testing

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

*[Present future improvements for the system]*

# Bibliography