Fancy Restaurant

Analysis and Design Document

Student: Aralda Păcurar

**Group: 30238**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

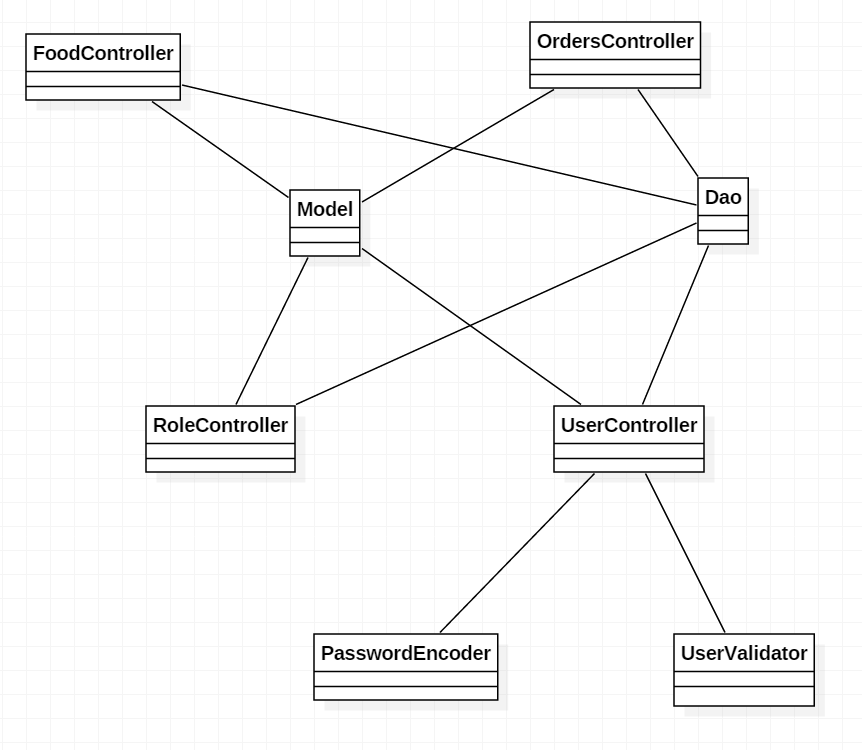
# Project Specification

# The project is meant to be an application developed for the Fancy Restaurant. It’s main goal is to enable online ordering from the restaurant so that clients can order the food that they want from a distance.

The application will have three types of users: clients, employees and admins, each one having a set of rights. Clients can order food and view the list of available food, employees can also add and remove food from the list and the admin can do everything an employee can do aswell as hiring and firing employees.

# Elaboration – Iteration 1.1

# Domain Model



# Architectural Design

## Conceptual Architecture

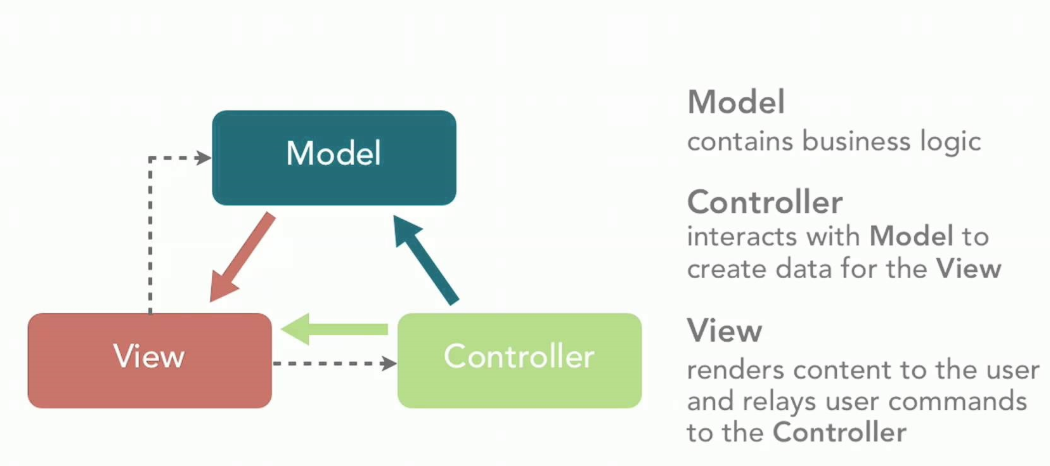
Model-View-Controller(MVC) is an architectural pattern commonly used for developing user interfaces that divide an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented and accepted from the user. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development.

The Model is the central component of the pattern. It is the application’s dynamic data structure, independent of the user interface.

The View is the representation of the information.

The Controller accepts input and converts it to commands for model or view.

## Package Design



## Component and Deployment Diagrams

*[Create the 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

# Data Model

# 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

Every functionality will be tested individually, using multiple scenarios and the results will be written in this document.

# Future improvements

In the future, the application can be improved by adding the possibility to cancel an order that was already placed.

# Bibliography

1. <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>
2. <https://firebirdsql.org/file/documentation/reference_manuals/fbdevgd-en/html/fbdevgd30-dot-net-mvc.html>