Reasy Peasy

Analysis and Design Document

Student:Adrian Moldovan

**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

# My project is called “Reasy Peasey”.It is a reservation system for restaurants and any others pubs.The main idea is to make the reservation easier for clients and for Staff by doing all the work online.The user will be able to see in real time all tables and reserve a free one at the desired time.The pubs staff will have to accept the reservations.

Since every pub is different, there will be a secondary application for pubs where the

registration can be done.Pub registration will also have to be accepted by Application staff

(Me/Company staff) and if it doesn’t respect the requirments it will be declined with a

Message.

# Elaboration – Iteration 1.1

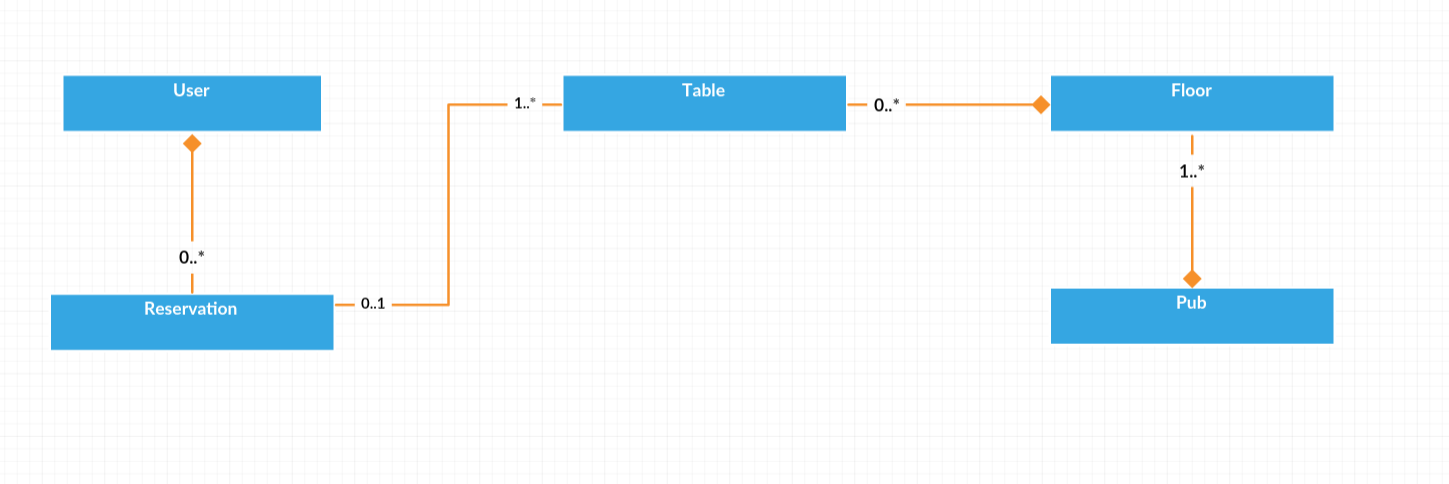
# Domain Model

An user can make reservations.

A reservation must include the desired pub, table, date and time.

Every pub consists of one ore more floors.

Every floor can have 0 or more tables.



# Architectural Design

## Conceptual Architecture

I chose MVC(Model-View-Controller).

  MVC is a software architectural pattern for implementing user interfaces on computers. It divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is presented to or accepted from the user.

**Components of MVC**

1) **Model:** It specifies the logical structure of data in a software application and the high-level class associated with it. It is the domain-specific representation of the data which describes the working of an application. When a model changes its state, domain notifies its associated views, so they can refresh.

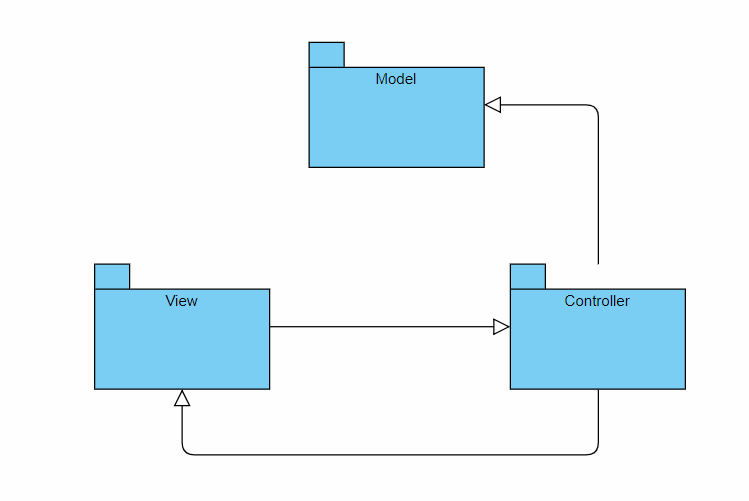
2) **View:** View component is used for all the UI logic of the application and these are the components that display the application’s user interface (UI). It renders the model into a form suitable for interaction. Multiple views can exist for a single model for different purposes.

3) **Controller:** Controllers act as an interface between Model and View components. It processes all the business logic and incoming requests, manipulate data using the Model component, and interact with the Views to render the final output. It receives input and initiates a response by making calls on model objects.

**Reasons for my choice:**

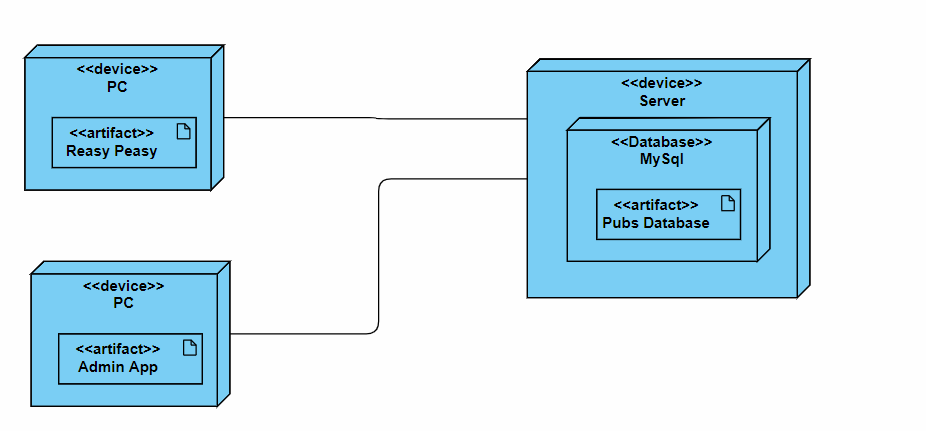
1. **Faster development process:** MVC supports rapid and parallel development. With MVC, one programmer can work on the view while other can work on the controller to create business logic of the web application
2. **Ability to provide multiple views:** In the MVC Model, you can create multiple views for a model. Code duplication is very limited in MVC because it separates data and business logic from the display.
3. **Support for asynchronous technique:** MVC also supports asynchronous technique, which helps developers to develop an application that loads very fast.
4. **Modification does not affect the entire model:** Modification does not affect the entire model because model part does not depend on the views part. Therefore, any changes in the Model will not affect the entire architecture.
5. **MVC model returns the data without formatting:** MVC pattern returns data without applying any formatting so the same components can be used and called for use with any interface.
6. **SEO friendly Development platform**: Using this platform, it is very easy to develop SEO-friendly URLs to generate more visits from a specific application.

## Package Design



## Component and Deployment Diagrams

**Deployment Diagram:**

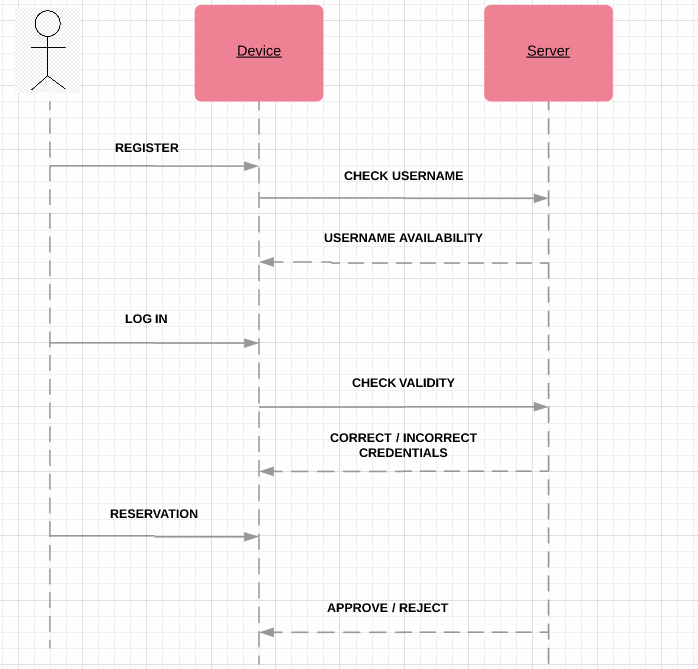
****

# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

**Sequence Diagram**

****

## 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