Public Transport Tickets

Analysis and Design Document

Student:Luca-Dan Adrian

**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 main objective of this project is to create a simple way for a user to buy tickets for a bus and to find out about the schedule of any bus. The users would be able to pay for a ticket through a paying method and then they will receive a code for the bus they are on, in order to be checked if necessary.

# 

# Elaboration – Iteration 1.1

# Domain Model

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

# Architectural Design

## Conceptual Architecture

In order to create a web application, I will use model-view-controller architectural pattern. It is used to create user interfaces and it is has three major components. This design pattern allows us to reuse code and to develop the application in a parallel way.

The three major parts are:

Model: Is the central component of the pattern. It is the application’s dynamic data structure which

Independent of the user interface. It manages data directly.

View: It is the representation of the information

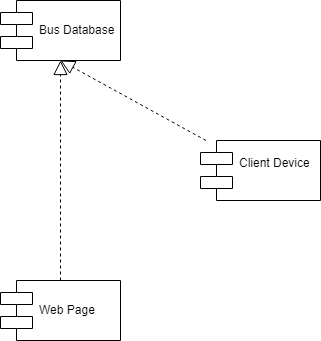
Controller: Accepts input and converts it to commands for the model or view. The controller responds to the user input and performs interactions on the data model objects. The controller receives the input and it passes the input to the model (optionally validates data)

## Package Design

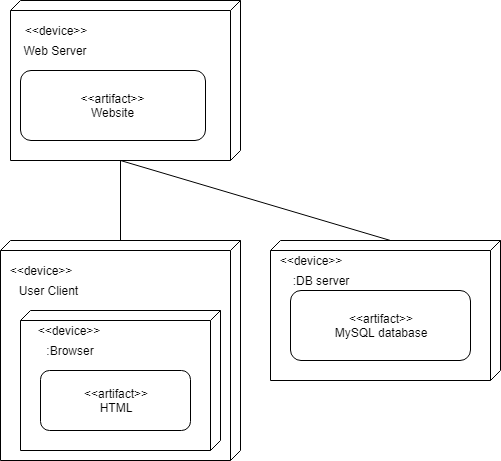
*[Create a package diagram]*

## Component and Deployment Diagrams

Component diagram: not yet finished



Deployment diagram:



# 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

<https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

<https://en.wikipedia.org/wiki/Component_diagram>

<https://www.lucidchart.com/pages/uml-component-diagram>

<https://en.wikipedia.org/wiki/Deployment_diagram>