Project Report for FlightFareComparer WebApp

Practice Module for Certificate in Designing Modern Software Systems

Team 8

Members:

|  |  |
| --- | --- |
| Name | Email |
| GOH JONG YANN | e0191317@u.nus.edu |
| LING JIE | e1333158@u.nus.edu |
| ONG JIA YUAN | e0415452@u.nus.edu |
| YEO MING HUI CINDY | e1336362@u.nus.edu |
| NG TIAN XUN | e1327862@u.nus.edu |

CONTENTS

1. Introduction 4

1.1 Background 4

1.2 Business Needs 4

1.3 Stakeholders 4

1.4 Project Scope 4

1.4.1 Functionality in Scope 4

1.4.2 Functionality out of Scope 4

1.4.3 Quality Attributes 4

2. Project Conduct 5

2.1 Project Plan 5

2.2 Project Status 5

2.3 Project Metrics 5

3. System Design 6

3.1 Software Architecture 6

3.2 Transition from Analysis to Design 6

3.2.1 <Transition Strategy N> 6

3.3 Use Case Model 6

3.3.1 Use Case Diagram 6

3.3.2 <Use Case Description N> 6

3.4 Analysis & Design Models 6

3.4.1 <Use Case N> 6

3.5 Design Problems and Patterns 6

3.5.1 <Design Problem N> 6

3.6 Database Schemas 7

4. DevSecOps and Development Lifecycle 8

4.1 Source Control Strategy 8

4.2 Continuous Integration 8

4.3 Continuous Delivery 8

5. <other things to be highlighted> 9

# Introduction

## Background

The FlightFareComparer WebApp is designed to provide users with a comprehensive platform to compare flight fares across multiple airlines, enabling them to find the best airlines for their travel plans.

In this project, the team will develop a web application that allows the user to search and compare prices for their ideal destination. After researching, user is able to save and bookmark their flights under their account for future reference.

## Business Needs

With a great variety of airlines available today, travellers will have a hard time comparing flights across many websites. FlightFareComparer is developed to help travellers shorten their research time by short listing and sorting the best deal for their ideal travel destination.

## Stakeholders

Travel agents, corporate clients and frequent travellers are the main stake holders for FlightFareComparer.

## Project Scope

Briefly describe the scope of the solution. State what is included in the scope and what is out of scope. If relevant, define the phases of the solution and what is delivered in each phase. Provide references to the documented scope. Also clearly define the scope for the architecture, design and implementations if you do not implement everything that you have architected and designed.

### Functionality in Scope

Briefly list down the requirements that are in the scope of the project.

### Functionality out of Scope

Briefly list down the requirements that are out of scope of the project.

### Quality Attributes

Briefly list down the non-functional requirements of the project.

# Project Conduct

## Project Plan

Describe the rough WBS tasks and the estimated efforts. In particular, clearly list the team members who is in charge of (a) the analysis & design of a use case (or user story) as well as (b) the identification and solving of a design problem using design pattern.

## Project Status

Describe the current status of the project. Highlight if there are outstanding issues to be addressed.

## Project Metrics

Provide the actual project milestones achieved as well as the rough total effort expended by each team member.

# System Design

## Software Architecture

Describe using suitable UML diagrams the overall software architecture of the system in terms of the software components (if the system is distributed) as well as the logical architectural layers and packages of the system.

Describe the target implementation platform in terms of the technology stacks, programming languages, application platforms, application frameworks, databases, etc.

## Transition from Analysis to Design

List the transition strategies that are required to guide the software designers to design a use case (or user story). The identified strategies should be sufficient to cover the design of the system end-to-end.

### User story

|  |  |  |
| --- | --- | --- |
|  | **User Story** | **Acceptance Criteria** |
| 1 | As a frontend and backend developer, I want to have different repository so that we can differentiate the development | Test that there is different repository for frontend and backend. |
| 2 | As a ci/cd developer, I want to host the source code on a web server so that my team member is able to checkin and checkout the code easily | Test that developer is able to clone the repository into their laptop |
| 3 | As a user, I would like to save my search in the web so that I can reduce time entering duplicate fields when searching | Test that developer is able to check in their code after their development" |

### <Transition Strategy N>

For each transition strategy, describe clearly the types of analysis objects impacted, the types of use cases (or user stories) impacted, the typical changes to the static structure (in class diagram) including the rationale, and the typical changes to the dynamic behaviour (in sequence diagram) including the rationale. The level of details should be sufficient to guide a newly joined software designer to design effectively.

## Use Case Model

### Use Case Diagram

Describe using suitable UML diagrams the overall use case model of the system. It must include the use cases that are in the scope as well as use cases that may be out of scope.

### <Use Case Description N>

For each use case in the scope, describe clearly its flow of events including normal flows and the relevant exceptional flows. The level of details should allow effective identification of analysis objects and interactions.

## Analysis & Design Models

### <Use Case N> by <Name of Member M>

For each use case in the scope, describe using suitable UML diagrams the analysis and the design of the use case. In general, the analysis of a use case can be captured in a class diagram and a sequence diagram; the same goes for the design too. Do ensure that the design complies with the applicable transition strategies. Each member of the team must have at least one use case to his/her name. For **each** use case, there should be:

* one class diagram for the analysis model
* one sequence diagram for each major flow for the analysis model
* one class diagram for the design model
* one sequence diagram for each major flow for the design model.

## Design Problems and Patterns

### <Design Problem N> by <Name of Member M>

For each design problem, articulate clearly the design problem (using brute force design if necessary), list the candidate design patterns, justify the selection of the design pattern, using class diagrams to show the static structure before and after applying the design pattern, using sequence diagrams to show the dynamic behaviours before and after applying the design pattern, and describe the applicable implementation decisions made for the design pattern. Note that the implementation of the system should be based on the design after applying the design pattern. Each member of the team must have at least one design problem to his/her name. For **each** design problem, there should be:

* Description of design problem with class and sequence diagrams
* Candidate design pattern(s)
* Motivation to choose a design pattern(s)
* Class diagram for the design solution
* Sequence diagram(s) for the design solution
* Implementation decisions

## Database Schemas

Describe using suitable diagrams the schema(s) of the database(s) required to support the system.

# DevSecOps and Development Lifecycle

## Source Control Strategy

Describe the strategy that you use to manage your source code and other artefacts in your configuration management. Include your project structures, repository strategy, branching strategy, authentication of developers, etc.

## Continuous Integration

Describe the pipeline on how you perform continuous integration in your project including the trigger for each integration jobs, the various jobs, tests, etc.

## Continuous Delivery

Describe the pipeline on how you automate deployment to different environment including how many environments are setup, who is responsible for approving promotion between environments and how deployment get verified.

# <other things to be highlighted>