 ARCHITECTURE

FLIGHT FARE ESTIMATOR

|  |  |
| --- | --- |
| **Written By** | **Anmol Vaid** |
| **Document Version** | **1.0** |
| **Last Revised Data** | **11-Dec-2021** |

Document Version Control:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| 1.0 | 11-Dec-2021 | Anmol Vaid |  |
|  |  |  |  |

Contents

[1. Introduction 2](#_Toc90218322)

[1.1 What is Low-Level design document? 2](#_Toc90218323)

[1.2. Scope 2](#_Toc90218324)

[2. Architecture 2](#_Toc90218325)

[3. Dataset 3](#_Toc90218326)

[3.1 Dataset Overview 3](#_Toc90218327)

[4. Logging 4](#_Toc90218328)

[5. Database 4](#_Toc90218329)

[6. Deployment 4](#_Toc90218330)

[7. Proposed Solution 4](#_Toc90218331)

[8. User I/O workflow 4](#_Toc90218332)

[9. Test Cases 5](#_Toc90218333)

# Introduction

## 1.1 What is Low-Level design document?

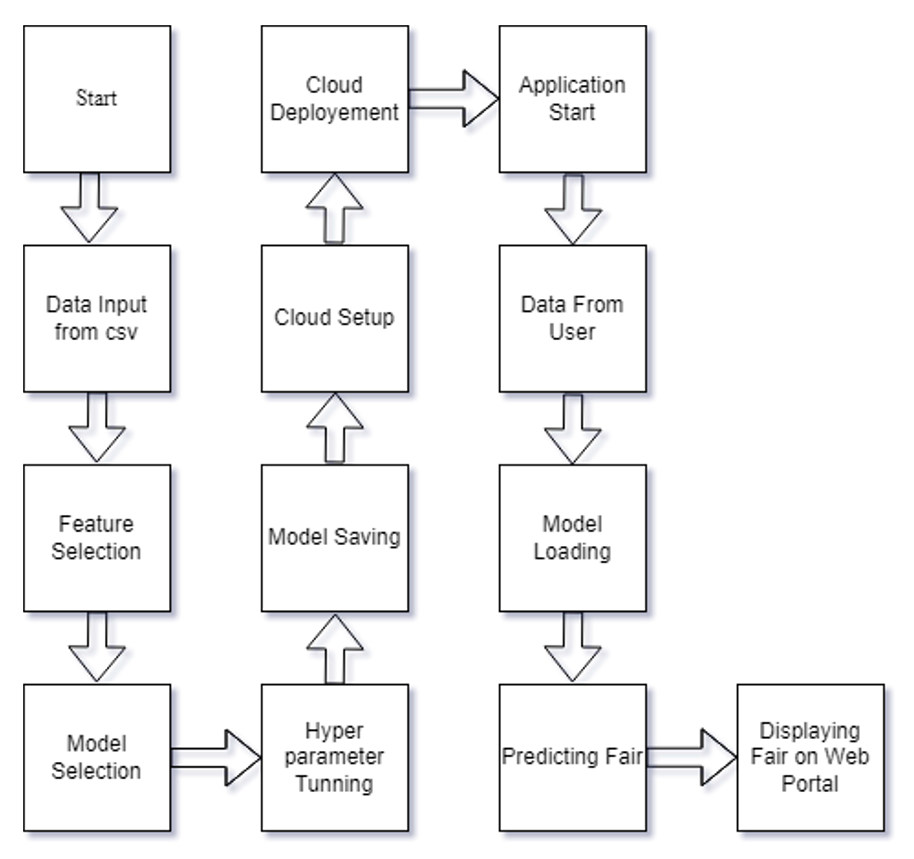
The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for flight fare estimation System. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so that the programmer can directly code the program from the document.

## 1.2. Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work

# Architecture

Below is architecture of this project.



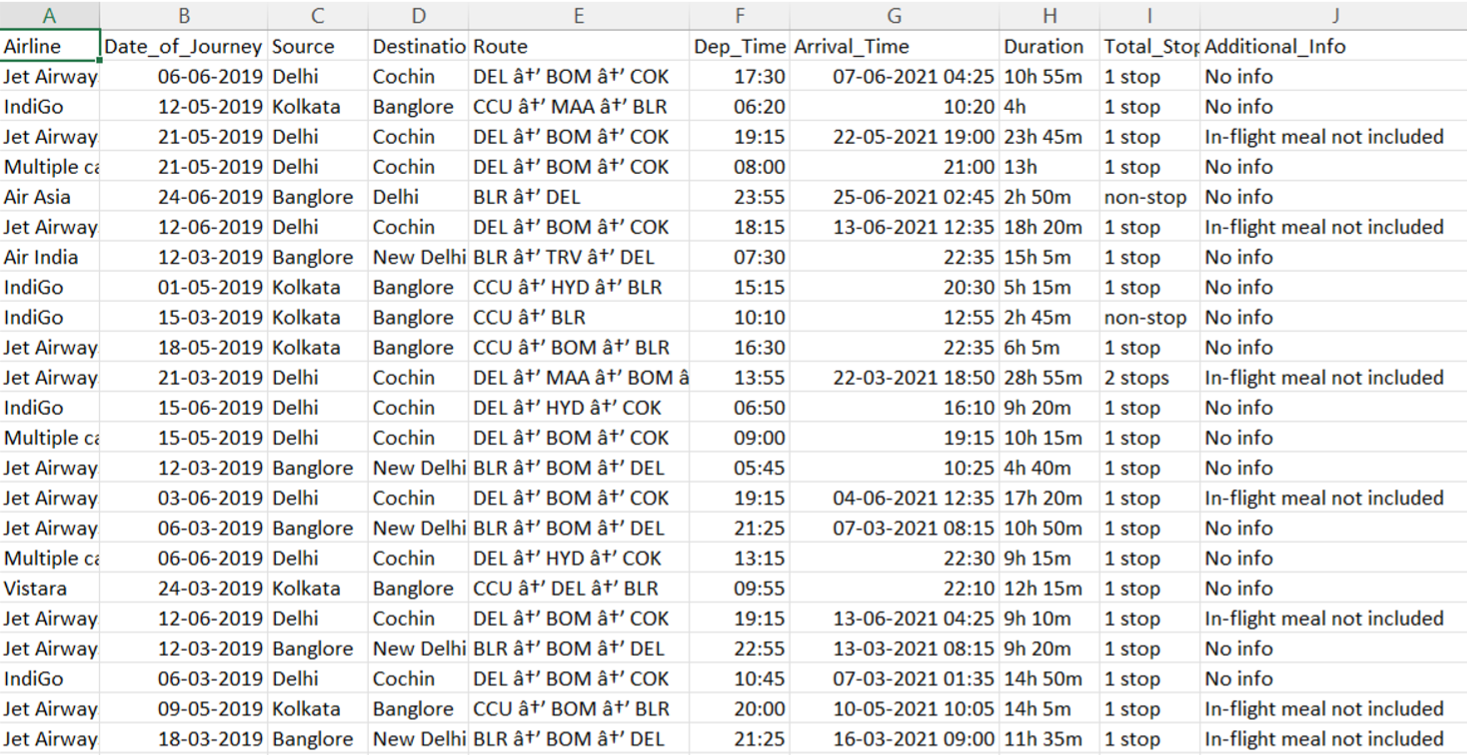
# 3. Dataset

## 3.1 Dataset Overview

The training dataset consists of 12 columns and every column datatype is string except the last one that is “Price” which is a integer datatype.



Testing Data consists of only 10 columns because there will not be two columns first is Id and second is Price. The data type inside the test day for every column is a string.



# 4. Logging

Logging is very important to keep track of the activities performed by our application. I have used logging module to do so. All the logs either it is train or test, both will be present inside filelog folder. Logging helps us in debugging process also so it is mandatory to do.

# 5. Database

No database is use for in this case. This was Kagggle dataset available in csv format.

# 6. Deployment

Deployment is done in Heroku.

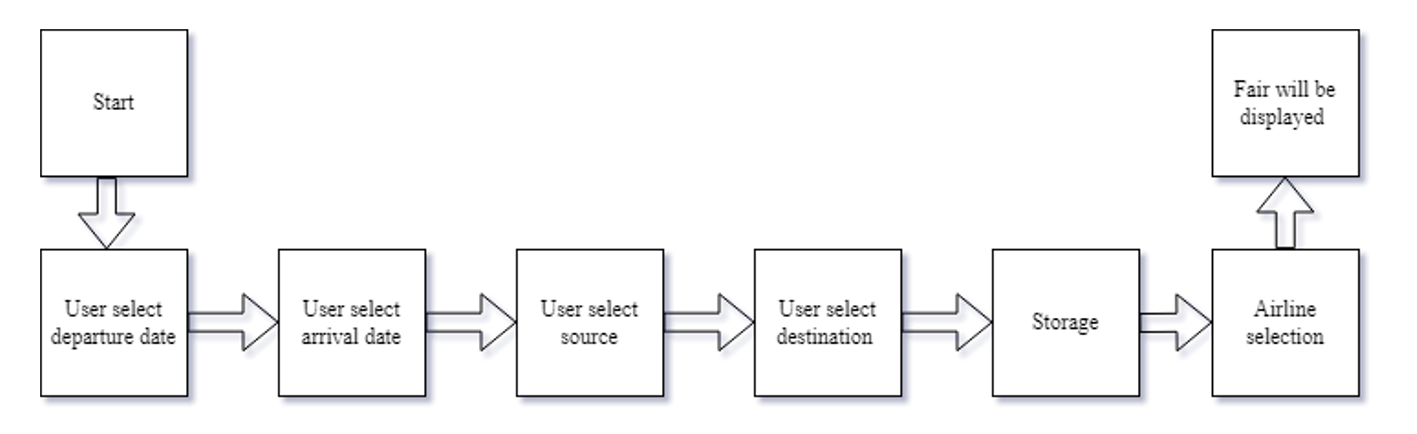
Link:  https://flight-test-011.herokuapp.com/predict

# 7. Proposed Solution

For this we are going to build a simple ML model which will be able to predict the flight fare based on the data given. Doing some EDA on the dataset we got to know that xgboost, Random Forest will be the best for this case.

# 8. User I/O workflow

    Below is user flow.



# 9. Test Cases

Test cases are given below

|  |  |  |
| --- | --- | --- |
| **Test Case Description** | **Pre-Requisite** | **Expected Result** |
| Verify whether the Application URL is accessible to the user |    Application URL should be defined | Yes, it is defined. |
| Verify whether the Application loads completely for the user when the URL is accessed |    Application URL is accessible | Yes, it is accessible and deployed |
|    Application is deployed |