

# Low Level Design (LLD)

## Air Quality Index

Written By	Diana Laveena DSouza
Document Version	0.1
Last Revised Date	23-Jan-2023

## Document Control

### Change Record:

Version	Date	Author	Author
0.1	23/01/2023	Diana Laveena DSouza	Introduction & Architecture Defined

### Reviews:

Version	Date	Reviewer	Comments
0.1	23/01/2023		Document Content, Version Control and Unit Test Cases to be added

### Approval Status:

Version	Review Date	Reviewed By	Approved By	Comments

# Contents

1	Introduction .....	4
1.1	What is Low-Level design document? .....	4
1.2	Scope .....	4
2	Architecture .....	5
3	Architecture Description .....	6
3.1	Data Description .....	6
3.2	Data Collection .....	6
3.3	Data Insertion into Database .....	6
3.4	Data Preprocessing .....	6
3.5	EDA Analysis .....	6
3.6	Data Preprocessing .....	7
3.7	Model Building and Evaluation .....	7
3.8	Data from User .....	7
3.9	Data Validation .....	7
3.10	Prediction .....	7
3.11	Deployment .....	7
4	Unit Test Cases .....	8

# 1 Introduction

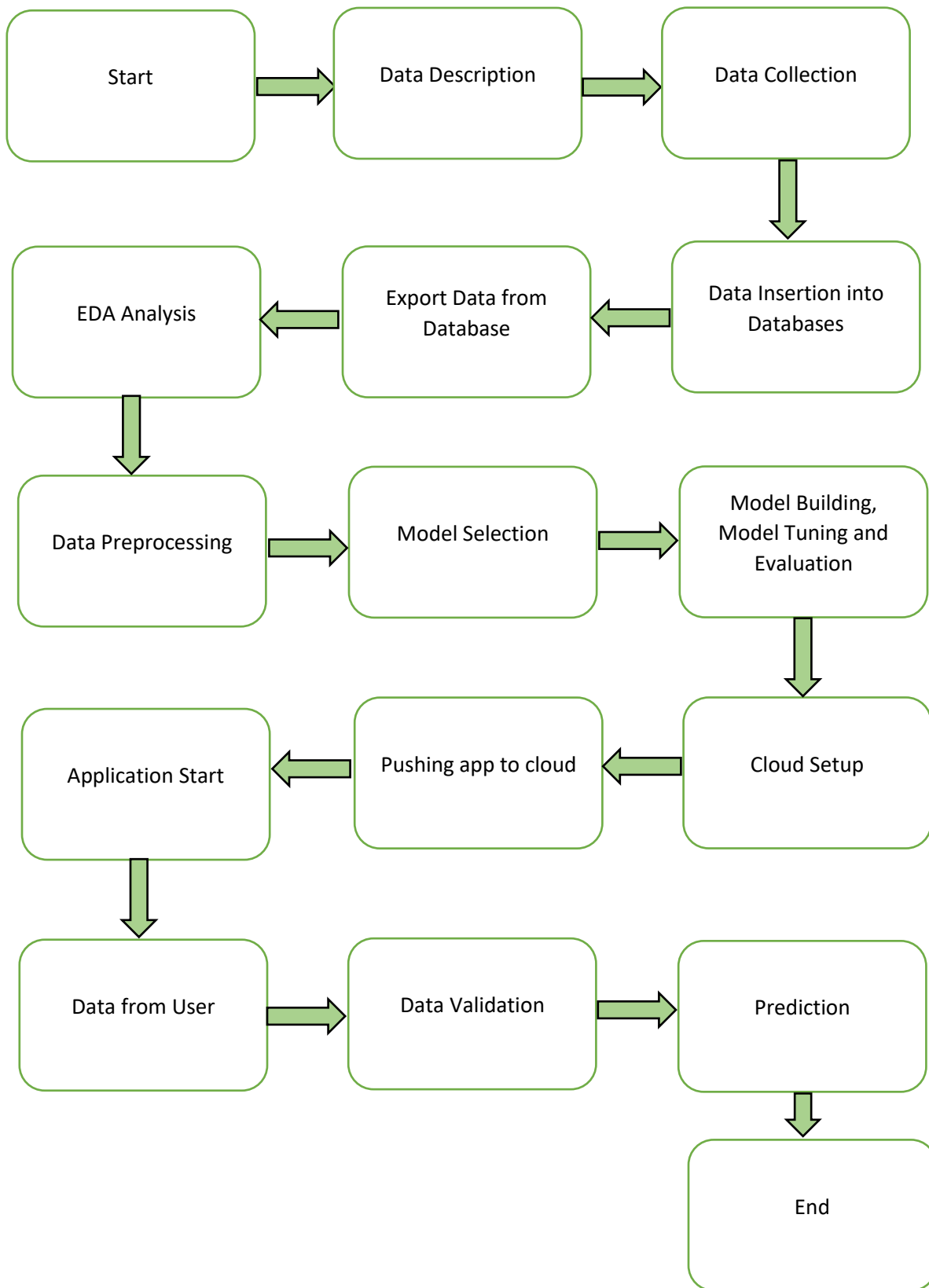
## 1.1 Why is Low-Level Design Document?

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 Prediction. LLD describes the class diagrams with the methods and relations between classes and program specs. It describes the modules so 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.

## 2 Architecture



## 3 Architecture Description

### 3.1 Data Description

**Context:** Collection of information about Air\_Quality\_22\_23 and their AQI helps greatly in Regression tasks.

**Content:** List of records contains State, City, Station, From Date, To Date, PM2.5, PM10, NO2, NH3, CO, SO2, Ozone, AQI, AQI\_Bucket; Dataset is split into train & validation for building the machine learning model.

### 3.2 Data Collection

Data Collected from the Central Pollution Control Board, Ministry of Environment.

### 3.3 Data Insertion into Databases

- Database creation and connection – Create a Cassandra database with a namespace key.
- Table creation in the database.
- Insertion of files in the table.

### 3.4 Export Data from Database

Data Export from Database – The data in a stored database is exported as a CSV file to be used for Data Pre-processing and Model Training.

### 3.5 EDA Analysis

EDA Analysis includes Descriptive Statistics, Univariate, Bivariate Analysis, Hypothesis Testing, Checking missing values, outliers, and inconsistency in the columns, duplicate records, checking target class counts.

## **3.6 Data Preprocessing**

Data Preprocessing includes removing unwanted features, duplicate records if present, handling outliers, missing values, inconsistency in columns, feature creation, feature transformation, Label Encoding/One Hot Encoding. Splitting preprocessed data into train and validation sets.

## **3.7 Model Building and Evaluation**

We will find the best model to predict Air Quality Index. We will calculate the RMSE, RMSLE and R2 scores for models and select the model with the best score. CI/CD using DVC and CML tools.

## **3.8 Data from User**

Here we will collect the user's air pollutants information.

## **3.9 Data Validation**

Here Data Validation will be done, given by the user.

## **3.10 Prediction**

The model will predict the Air Quality Index.

## **3.11 Deployment**

We will be deploying the model to GCP.

This is a workflow diagram for the Air Quality Prediction.

## 4 Unit Test Cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Application URL is accessible to the user	1. Application URL should be defined	The application URL should be accessible to the user.
Verify whether the Application loads entirely for the user when the URL is accessed	1. Application URL is accessible 2. Application is deployed	The Application should load entirely for the user when the URL is accessed.
Verify whether the user can input the text in all input fields	1. Application is accessible	The user should be able to input the text in all input fields.
Verify whether the user gets Submit button to submit the inputs.		The user should get Submit button to submit the inputs.
Verify whether the user is presented with results on clicking submit.		The user should be presented with results on clicking submit