Low-Level Document (LLD)
Flight Price Prediction Project

# Data Collection and Preprocessing:

1.1. Data Collection:
Identify and select reliable sources for historical flight data, such as airline websites, travel agencies, or APIs.
Define the required attributes to be collected, including departure/arrival locations, dates, flight duration, airline, and ticket price.
Implement a data retrieval mechanism to fetch the data from the selected sources.

1.2. Data Cleaning and Transformation:
Perform data cleaning to handle missing values, outliers, and data inconsistencies.
Ensure consistency in attribute formats and data types.
Remove any duplicate records from the dataset.

1.3. Feature Engineering:
Analyze the collected data to identify additional relevant features that can enhance the prediction accuracy.
Derive new features from existing attributes, such as day of the week, month, or season.
Perform feature scaling or normalization, if required, to bring features to a similar scale.

Machine Learning Model Development:

2.1. Model Selection:
Evaluate different regression algorithms such as Random Forest, Gradient Boosting, or Neural Networks.
Select the most suitable algorithm based on factors like prediction accuracy, training time, and interpretability.

2.2. Data Splitting:
Split the preprocessed dataset into training and testing sets.
Allocate a certain percentage (e.g., 80:20) for training and evaluation respectively.

2.3. Model Training:

Implement the chosen regression algorithm using a suitable machine learning library (e.g., scikit-learn).
Train the model using the training dataset.
Fine-tune hyperparameters through techniques like grid search or random search.

2.4. Model Evaluation:
Evaluate the trained model's performance using appropriate regression metrics such as mean squared error (MSE), mean absolute error (MAE), or R-squared score.
Compare the model's performance against baseline models or industry standards.

User Interface Design and Implementation:
3.1. Interface Design:
Design an intuitive and user-friendly interface for users to input their travel preferences.
Determine the required input fields, including departure/arrival locations, travel dates, and any additional relevant information.

3.2. Interface Development:
Implement the user interface using appropriate technologies such as HTML, CSS, and JavaScript.
Integrate the interface with the machine learning model to display predicted ticket prices based on user input.

Model Updates and Maintenance:
4.1. Data Collection for Updates:
Establish a regular data collection process to retrieve updated historical flight data.
Ensure the collected data is representative of the latest market conditions.

4.2. Model Retraining:
Periodically retrain the machine learning model using the updated dataset to improve prediction accuracy.
Implement a schedule for model updates, considering factors like data availability and prediction performance.

4.3. Automatic Model Updates:
Set up a mechanism to automatically update the trained model and its predictions without disrupting user experience.
Implement version control and rollback mechanisms to handle potential issues arising from model updates.

# Deployment and Testing:
5.1. Deployment Environment:
Select an appropriate environment for deploying the Flight Price Prediction system, such as a web server or a cloud platform.

5.2. System Integration and Testing:
Integrate the trained machine learning model, user interface, and data retrieval components into a cohesive system.
Conduct thorough testing to ensure all components function correctly and provide accurate predictions.

Documentation:
6.1. User Guide:
Create a comprehensive user guide to help users understand how to use the Flight Price Prediction system effectively.
Include instructions on inputting travel preferences, interpreting predictions, and utilizing additional features.

6.2. Technical Documentation:
Document the technical details of the system, including the system architecture, algorithms used, and implementation guidelines.
Provide instructions for future maintenance, troubleshooting, and potential enhancements.