Low Level Design (LLD)

AI-Powered Data Analysis System

Revision Number: 1.8
Last date of revision: 22/08/2024

BY :- VIRAJ TUPE

Low-Level Design Document (LLD)

1. Overview

The Low-Level Design (LLD) document outlines the detailed implementation of various modules in the AI-powered data analysis system. This document focuses on the specific functions, their inputs and outputs, and error handling mechanisms. The system is designed to process data, apply machine learning algorithms, provide an API for analysis, and store data in a database. The following sections describe the LLD in detail.

---

2. Module 1: Data Processing

Function: load_data(file_path)

- **Input:**
  - file_path (string): The path to the CSV file that contains the data.
- **Output:**
  - DataFrame: A Pandas DataFrame containing the loaded data from the CSV file.
- **Error Handling:**
  - Log errors if the file cannot be found or loaded. The function should handle common issues such as file not found errors or incorrect file formats.
- **Description:**
  - This function is responsible for loading data from a specified CSV file. It reads the data into a Pandas DataFrame, which is then used for further processing. The function includes error handling to ensure that any issues during data loading are logged appropriately.

Function: preprocess_data(data, target_column)

- **Input:**
  - data (DataFrame): The DataFrame containing the loaded data.
  - target_column (string): The name of the target column in the DataFrame.
- **Output:**

- X_train, X_test, y_train, y_test (DataFrames): DataFrames containing the training and testing data split.

- **Operations:**
  - Handle missing values by either filling them with a default value or removing the corresponding rows.
  - Split the data into training and testing sets using a standard ratio (e.g., 80-20).

- **Description:**
  - This function preprocesses the data before it is fed into the machine learning model. It handles missing values, which are common in real-world datasets, and splits the data into training and testing sets. The target column, which contains the labels, is separated from the features during this process.

---

3. Module 2: Machine Learning

Function: build_model(X_train, y_train)

- **Input:**
  - X_train (DataFrame): The training data containing the features.
  - y_train (DataFrame): The training data containing the labels.

- **Output:**
  - model (object): A trained machine learning model.

- **Algorithm:**
  - Linear Regression is used as the base algorithm. This can be expanded to include other algorithms like Decision Trees, Random Forests, or Neural Networks depending on the use case.

- **Description:**
  - The build_model function trains a machine learning model using the provided training data. The function can be easily extended to use different machine learning algorithms by modifying the underlying implementation. The trained model is returned for evaluation and prediction.

Function: evaluate_model(model, X_test, y_test)

- **Input:**
    - model (object): The trained machine learning model.
    - X_test (DataFrame): The testing data containing the features.
    - y_test (DataFrame): The testing data containing the labels.
- **Output:**
    - metrics (dictionary): A dictionary containing model performance metrics like Mean Squared Error (MSE) and R-squared (R^2) values.
- **Operations:**
    - Generate predictions on the test data.
    - Calculate performance metrics like MSE and R^2 to evaluate the model's accuracy.
- **Description:**
    - This function evaluates the performance of the trained model using the testing data. The performance metrics provide insight into how well the model has learned from the training data and its ability to generalize to new data. The function can also include additional metrics such as precision, recall, and F1-score for classification problems.

---

4. Module 3: API

Function: /analyze

- **Method:** POST
- **Input:**
    - JSON (object): A JSON object containing the data to be analyzed.
- **Output:**
    - JSON (object): A JSON object containing the results of the analysis, including model predictions and any relevant metrics.
- **Error Handling:**
    - Return error messages for invalid inputs or any issues during data processing or model evaluation.

- **Description:**
    - The /analyze endpoint serves as the main entry point for the API. Users can send data in JSON format to this endpoint, which will then be processed by the system. The system will use the trained model to analyze the data and return the results in JSON format. Proper error handling ensures that users receive informative messages if something goes wrong.

---

5. Module 4: Database

Function: connect_to_mongo(db_name)

- **Input:**
    - db_name (string): The name of the MongoDB database to connect to.
- **Output:**
    - db (object): A MongoDB database connection object.
- **Error Handling:**
    - Log errors if the connection to the database fails, such as network issues or incorrect database credentials.
- **Description:**
    - This function establishes a connection to the specified MongoDB database. It is critical for storing and retrieving data that the system processes. The connection object returned by this function is used for all subsequent database operations.

Function: insert_data(db, collection_name, data)

- **Input:**
    - db (object): The MongoDB database connection object.
    - collection_name (string): The name of the collection where data will be inserted.
    - data (dictionary or DataFrame): The data to be inserted into the collection.
- **Output:**
    - None
- **Operations:**

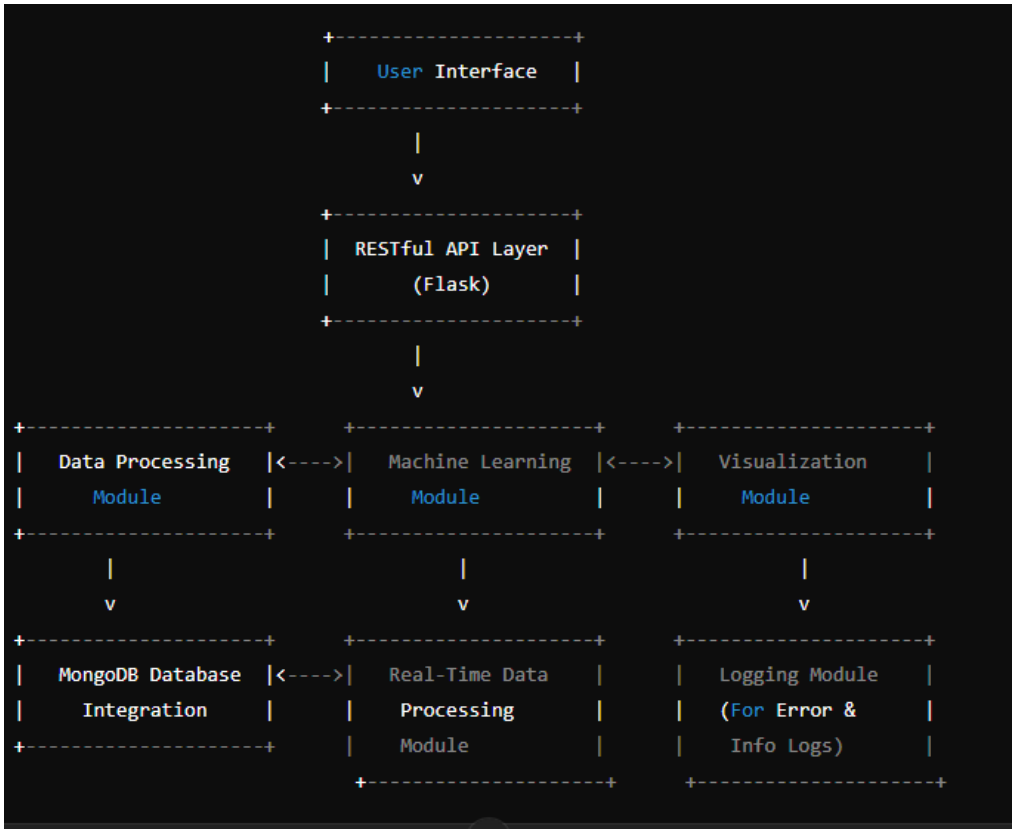
- Insert a document into the specified collection in the MongoDB database.

- **Description:**
  - The insert_data function is responsible for inserting data into a MongoDB collection. This function supports inserting individual documents or bulk inserts, depending on the input data format. The function does not return any value, but successful insertion will ensure that the data is stored for later retrieval or analysis.

---

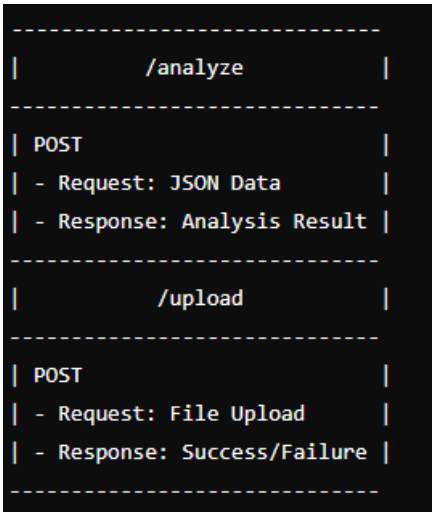
6. System Architecture Diagram

The system architecture for the AI-powered data analysis platform includes several key modules that interact to provide a seamless user experience. Below is an overview of the architecture:



7. Wireframe Documentation

1. API Endpoint Layout:



2. User Interface Layout (if developed as a web app):