Low Level Design (LLD)

ENERGY EFFICIENCY

Document Version: 0.1

Last Revised Date: 15-April-2023

Parvej alam Ansari

Document Version Control:

Version	Date	Author	Description
0.1	13-04-2023	Parvej alam	Introduction Architecure
0.1	15-04-2023	Parvej alam	Architecture Description Unit Test Cases

Contents

1. Introduction

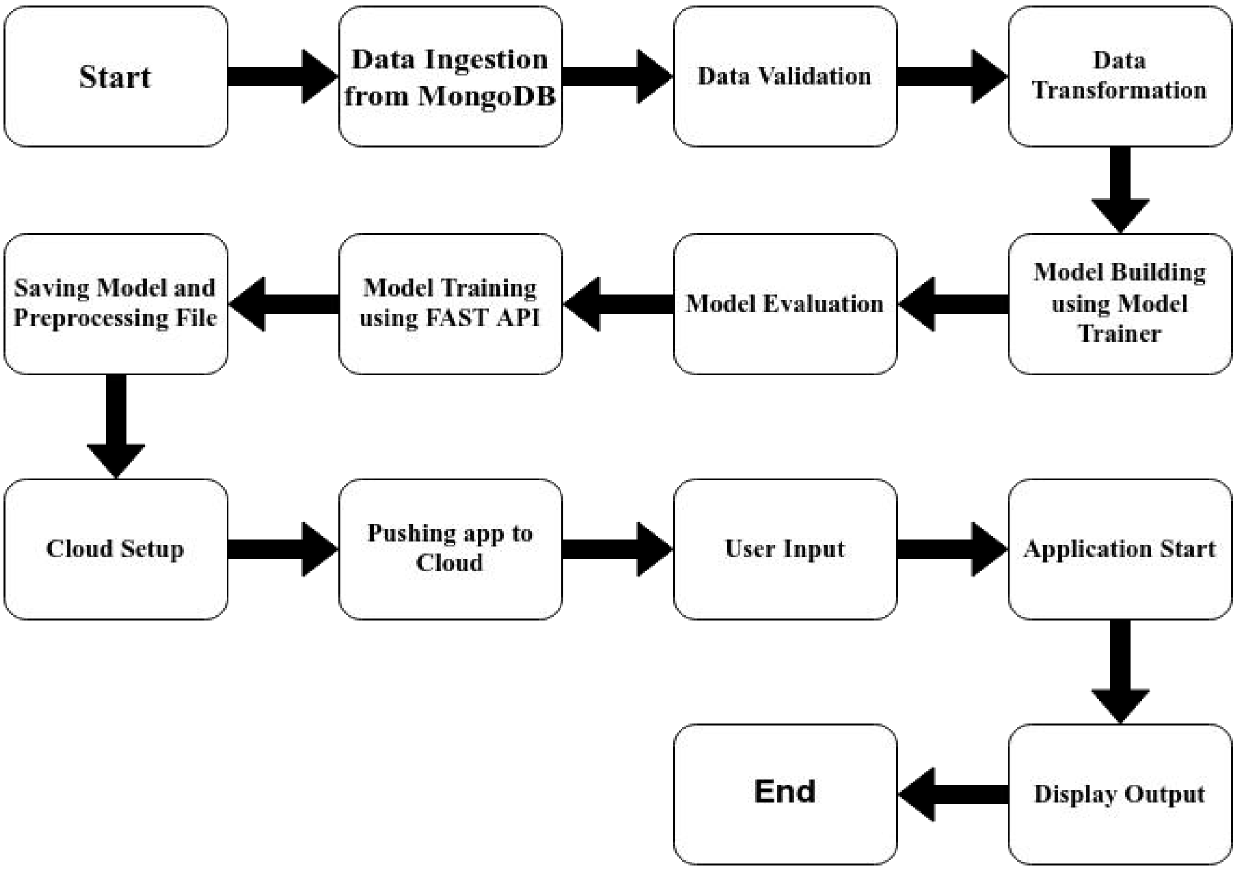
1.1 About Low-Level Design Document

The goal of LLD or a low-level design document (LLDD) is to give the internal logical design of the actual program code for Energy Efficiency 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.

2. Architecture



3. Architecture Description

3.1 Data Gathering

This data set we will be using is from the University of California, Irvine Machine Learning Repository.

The following is **UCI's information on the data set**:

Source:

The dataset was created by Angeliki Xifara (Civil/Structural Engineer) and was processed by Athanasios Tsanas (Oxford Centre for Industrial and Applied Mathematics, University of Oxford, UK).

Dataset link: https://archive.ics.uci.edu/ml/datasets/Energy+efficiency (excel format)

Dataset link: https://www.kaggle.com/datasets/elikplim/eergy-efficiency-dataset (csv format)

3.2 Dataset Description

Data Set Information:

We perform energy analysis using 12 different building shapes simulated in Ecotect. The buildings differ with respect to the glazing area, the glazing area distribution, and the orientation, amongst other parameters. We simulate various settings as functions of the afore-mentioned characteristics to obtain 768 building shapes. The dataset comprises 768 samples and 8 features, aiming to predict two real valued responses. It can also be used as a multi-class classification problem if the response is rounded to the nearest integer.

Attribute Information:

The dataset contains eight attributes (or features, denoted by X1...X8) and two responses (or outcomes, denoted by y1 and y2). The aim is to use the eight features to predict each of the two responses.

Specifically:

- Relative Compactness
- Surface Area - m^2
- Wall Area - m^2
- Roof Area - m^2
- Overall Height - m
- Orientation - 2: North, 3: East, 4: South, 5: West
- Glazing Area - 0%, 10%, 25%, 40% (of floor area)
- Glazing Area Distribution (Variance) - 1: Uniform, 2: North, 3: East, 4: South, 5: West
- Heating Load - kW
- Cooling Load - kW

	Relative Compactness	Surface Area	Wall Area	Roof Area	Overall Height	Orientation	Glazing Area	Glazing Area Distribution	Heating Load	Cooling Load
0	0.98	514.5	294.0	110.25	7.0	2	0.0	0	15.55	21.33
1	0.98	514.5	294.0	110.25	7.0	3	0.0	0	15.55	21.33
2	0.98	514.5	294.0	110.25	7.0	4	0.0	0	15.55	21.33
3	0.98	514.5	294.0	110.25	7.0	5	0.0	0	15.55	21.33
4	0.90	563.5	318.5	122.50	7.0	2	0.0	0	20.84	28.28

3.3. Data Ingestion from MongoDB

Data Insertion into Database

a. Database creation and connection - create a database with name passed. If the database is already created, open the connection to the database.
b. Table creation in the database.
c. Insertion of files in the table

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.

3.4. Data Validation

After data ingestion stage, the data should be validated in terms of number of columns, column names etc.

3.5. Data Transformation

As soon as data is successfully validated from data validation stage, The validated data is then sent to the data transformation stage where data is wrangled, cleaned by using some data imputation techniques. Furthermore if outliers are present then apply some outliers handling techniques. Next, if dataset is not having proper scaling then will apply proper scaling techniques to trasnform tha data. Now data is ready to next stage i.e. Model Trainer.

3.6. Model Building using Model Trainer

In this stage, Number of regression type supervised machine learning models should be apply and will try to find out the best suitable model as per our availbale dataset by using some metric evaluation techniques. After getting the best suitable regression model, its traning and testing accuracy will be determined again. Apart from that to improve the training and testing accuracy finalised model is hypertuned by finding out the best parameters using grid-search cv techinque.

3.7. Model Evaluation

In this stage, selected and hypertuned model is evaluated using some criteria like training and testing accuracy with the help of proper metric evaluation techniques, overfitting and underfitting etc. once this seleted model is properly evaluated, it will be passed to the pipeline stage.

3.8. Model Training using FAST API

After successful run of individual component, a pipeline will be created where each and every component discussed above will merge and that pipeline will be triggered using FAST API where the finalised model shall be trained with availabe training dataset.

3.9. Saving Model and Preprocessing File

In this stage, the trained model file plus the preprocessing file will be stored in pickeled format.

3.10. Cloud Setup and Model Pushing

After the model/product/app is ready, the app is deplyed to AWS by using some AWS services like Elastic Beanstalk (EB) and Docker hub. This cloud deplymnet will help user to access the application through any internet devices.

3.11. User Input and Application output

By using the app URL, user will get access to the application/model, where user will feed inputs to the model and that model will render the best outcome to its webpage in terms of heating load and cooling load.

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	Application URL should be accessible to the user
Verify whether the Application loads completely for the user when the URLis accessed	1. Application URLis accessible 2. Application is deployed	The Application should load completely for the user when theURL is accessed
Verify whether the User is able to sign up in the application	1. Application is accessible	The User should be able to sign up in the application
Verify whether user is able to see inputfields on logging in	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be able to see inputfields on logging in
Verify whether user is able to edit allinput fields	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be able to edit all inputfields
Verify whether user gets Submit button to submit the inputs	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should get Submit button tosubmit the inputs
Verify whether user is presented withrecommended results on clicking submit	1. Application is accessible 2. User is signed up to the application 3. User is logged in to the application	User should be presented with recommended results on clicking submit