Low Level Design Document

Store Sales Prediction

Arshad Shaikh

Version 1.0

01/04/2023

Version Control

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Description | Responsible Party | Date |
| 1.0 | Initial version | Arshad Shaikh | 01-04-2023 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Introduction

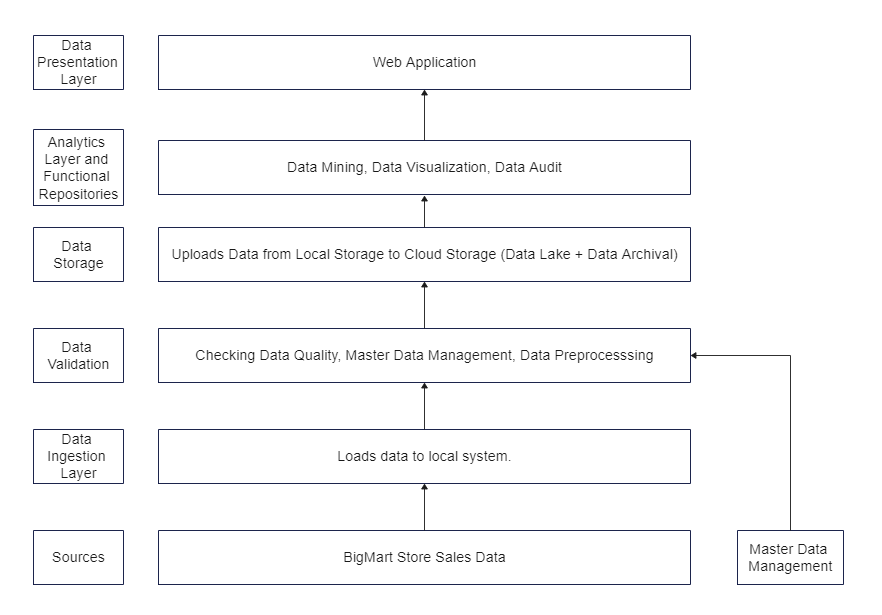
## What is Low Level Design Document?

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

## 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



# Architecture Description

## Data Description

Nowadays, shopping malls and Big Marts keep track of individual item sales data in order to forecast future client demand and adjust inventory management. In a data warehouse, these data stores hold a significant amount of consumer information and particular item details.

This dataset contains two files –

Train.csv – This contains 12 columns and 8523 rows.

Test.csv – This contains 11 columns and 5681 rows.

## Data Validation

This checks whether the data is as per the information provided by the master data management.

## Data Insertion To Database

* Database Creation and connection - Create a database with name passed. If the database is already created, open the connection to the database.
* Table creation in the database.
* 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.

## Data Preprocessing

Data Pre-processing steps we could use are Null value handling, stop words removal, punctuation removal, Tokenization, Lemmatization, TFIDF, Imbalanced data set handling, Handling columns with standard deviation zero or below a threshold, etc.

## Model Building

We will use Random Forest Regressor as the base model and use GridSearchCv for Hyperparameter Tuning. The model with the highest R2 score is selected.

## Deployment

The application is hosted on AWS.

# Unit Test Cases

|  |  |  |
| --- | --- | --- |
| Test Case Description | Prerequisite | Expected Result |
| Verify whether the Application URL is accessible to the user | Application URL should be defined | Application URL should be  accessible to the user |
| Verify whether the Application loads completely for the user when the URL is accessed. | Application URL is accessible  Application is deployed | The Application should load  completely for the user when the URL is accessed |
| Verify whether user is able to see input fields on homepage. | Application is accessible. | User should be able to see the input fields on opening the application. |
| Verify whether user is able to edit all input fields. | Application is accessible | User should be able to edit the input fields. |
| Verify whether user gets the submit button to submit the inputs. | Application is accessible | User should be presented with the prediction on clicking the submit button. |
| Verify whether user is presented with results on submission. | Application is accessible | User is able to see the predictions as per the input given. |
| Verify whether user is able to see the upload test file section. | Application is accessible | User should be able to see the upload file button. |
| Verify whether user is able to upload files | Application is accessible | User should be able to upload test files. |
| Verify whether user gets the download prediction button. | Application is accessible | A file should get downloaded on clicking the Download Prediction button. |
| Verify whether prediction file gets downloaded. | Application is accessible | A file gets downloaded. |
| Verify whether user is able to see the train data page. | Application is accessible | User should be able to see the train data page |
| Verify whether user is able to  see input fields | Application is accessible | User should be able to see the input fields on navigating to train data page. |
| Verify whether user is able to upload training file. | Application is accessible | User is able to upload training file and train new model. |
| Verify whether new model gets trained. | Application is accessible.  Valid Training data is provided. | New Model is trained |
| Verify whether user gets an email after model gets trained. | Application is accessible  Valid Training Data is provided.  Valid email is provided | An email is sent to the user. |