Low Level Design

Insurance Premium Prediction

Author: Rohit Murlidhar Shrimangle

Document Control

Change Record:

Date	Author	Comments
25-07-2024	Rohit Shrimangle	Introduction and Architecture Defined
25-07-2024	Rohit Shrimangle	Architecture Description
28-07-2024	Rohit Shrimangle	Unit test cases Defined

Review:

Date	Reviewer	Comments
28-07-2024	Rohit Shrimangle	Docs Content, Unit test cases

Content:

1. Introduction

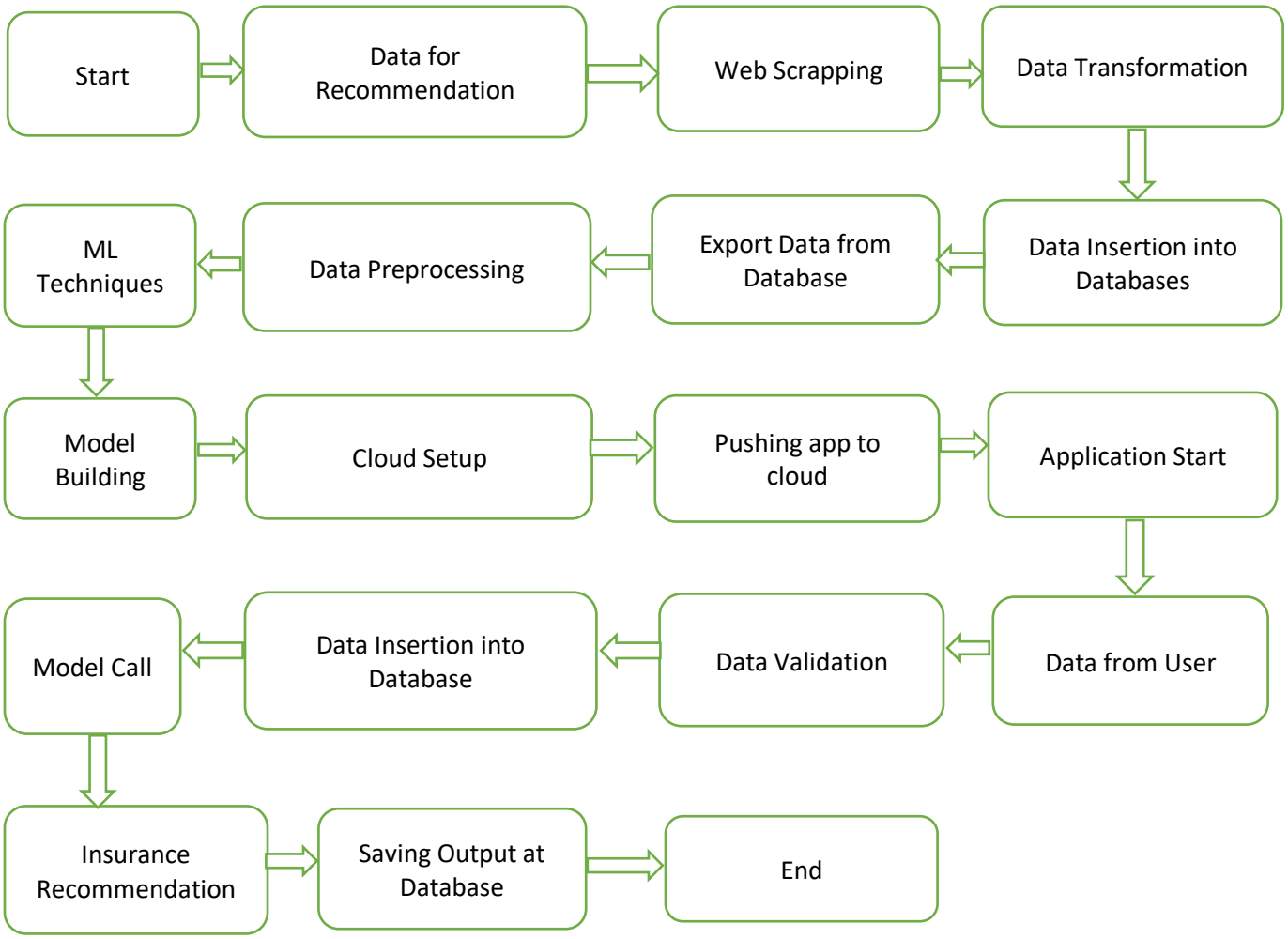
1.1. What is Low-Level design document?

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

The insurance.csv dataset contains 1338 observations (rows) and 7 features (columns). The dataset contains 4 numerical features (age, BMI, children and expenses) and 3 nominal features (sex, smoker and region) that were converted into factors with numerical value designated for each level.

3.2. Data Transformation

In the Transformation Process, we will convert our original dataset which is in JSON format to CSV Format. And will merge it with the Scrapped dataset.

3.3. 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

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. Data Pre-processing

Data Pre-processing steps we could use are Import the Libraries, Import the Loaded Data, Arrange the Data, Null value handling, outliers' removal, handling the duplicates, removal of unwanted features, Identification of numerical and categorical data, Imbalanced data set handling, Distribute Data into Training, Evaluation and Validation Sets, etc.

3.6. Model Building

we will find the best model. Algorithms will be passed with the best parameters derived. We will calculate the R2 scores, Mean Absolute Error (MAE), Mean Squared Error (MSE) for models and select the model with the best score. Similarly, All the models will be saved for use in Recommendation.

3.7. Data from User

Here we will collect physiological data from user such as user age, BMI, children, sex, smoker and region.

3.8. Data Validation

Here Data Validation will be done, given by the user

3.9. User Data Inserting into Database

Collecting the data from the user and storing it into the database. The database can be either MySQL or Mongo DB.

3.10. Model Call

Based on the model evaluation parameters, the respective model will be loaded and will be used to predict/Recommend the data.

3.11. Insurance Recommendation & Saving Output in Database

After calling model Expenses/Output will be recommended, this output will be saved in Database and it will be used to show the same Output if other users provide the same data.

3.12. Deployment

We will be deploying the model to AWS, but here I have used RENDER a free deployment platform.

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 URL is accessed	1. Application URL is accessible 2. 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	1. Application is accessible 2.User is able to access the URL	User should be able to see input fields on logging in
Verify whether user is able to edit all input fields	1. Application is accessible 2.User is able to access the URL	User should be able to edit all input fields
Verify whether user gets Submit button to submit the inputs	1. Application is accessible 2.User is able to access the URL	User should get Submit button to submit the inputs
Verify whether user is presented with recommended results on clicking submit	1. Application is accessible 2.User is able to access the URL	User should be presented with recommended results on clicking submit
Verify whether the recommended results are in accordance to the selections user made	1. Application is accessible 2.User is able to access the URL	The recommended results should be in accordance to the selections user made
Verify whether user has options to filter the recommended results as well	1. Application is accessible 2.User is able to access the URL	User should have options to filter the recommended results as well