Low Level Design (LLD)

# THYROID DISEASE DEDECTION APPLICATION

|  |  |
| --- | --- |
| Written By | N sri sai charan reddy |
| Document Version | 1.0 |
| Last Revised Date | 09-10-2023 |

**Document Control**

**Change Record:**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Comments |
| 1.0 | 09-Oct-2023 | N sri sai charan reddy | Introduction & Architecture Defined |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Reviews:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version | Date | Reviewer | Comments | | |
| 1.0 | 12-Oct-2023 | N sri sai charan reddy | Document Content, Version Control and Unit Test Cases to be Added | | |
| **Approval Status:** | | |  | | |
| Version | Review Date | Reviewer By |  | Approved By | Comments |
| 1.0 | 12-Oct-2023 | N sri sai charan reddy |  | N sri sai charan reddy |  |

## Contents

1. Introduction
   1. Why this Low-Level Design Document
   2. Scope
2. Architecture
3. Architecture Description
   1. Data Description
   2. Export Data from DB to CSV for training
   3. Data Preprocessing
   4. Data Clustering
   5. Get best model of each cluster
   6. Hyper parameter Tuning
   7. Model saving
   8. Cloud setup
   9. Push App to cloud
   10. Data from client side for prediction
   11. Export prediction to CSV
4. Unit Test Case

**Document Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Date Issued | Version | Description | Author |
| 09/10/2023 | 1 | Initial HLD – V1.0 | N sri sai charan reddy |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

### 1. Introduction

**1.1 What is Low-Level Design Document?**

The primary objective of a Low-Level Design Document (LLD) is to provide an internal, detailed blueprint of the program code for the Thyroid Disease Detection System. The LLD elucidates the internal logic by depicting class diagrams, including methods and their relationships, as well as program specifications. Additionally, it outlines the various modules within the system, enabling programmers to directly translate the document into program code..

#### 1.2 Scope

Low-level design (LLD) is a meticulous design process that operates at the component level and follows a systematic step-by-step refinement approach. This method is employed for designing various components, including data structures, essential software architecture, source code, and, ultimately, high-performance algorithms. It is important to note that while the basic data organization may be initially outlined during the requirement analysis phase, LLD refines and further defines this data organization as part of the data design process.

### 2. Architecture

Start

Pushing App to

Cloud

Get Best Model

for Each Cluster

Hyper parameter

Tu

ning

Model Saving

Cloud Setu

p

Data Clustering

Data

Preprocessing

Export Data From

Database To csv

for

Training

End

Export

Prediction to csv

Data

Preprocessing

Data

Clustering

Model call for

Specific Cluster

Prediction

Export Data From

csv To Prediction

Data From Client

to be Predicted

Application Start

### 3. Architecture Description

#### 3.1 Data Description

We will be using Thyroid Disease Data Set present in UCI Machine Learning Repository. This Data set is satisfying our data requirement. Total 7200 instances present in different batches of data.

#### 3.2 Export Data from database to CSV for Training

Here we will be exporting all batches of data from database into one csv file for training.

#### 3.3 Data Preprocessing

We will be exploring our data set here and do EDA if required and perform data preprocessing depending on the data set. We first explore our data set in Jupyter Notebook and decide what pre-processing and Validation we have to do such as imputation of null values, dropping some column, etc and then we have to write separate modules according to our analysis, so that we can implement that for training as well as prediction data.

#### 3.4 Data Clustering

K-Means algorithm will be used to create clusters in the pre-processed data. The optimum number of clusters is selected by plotting the elbow plot. The idea behind clustering is to implement different algorithms to train data in different clusters. The K-means model is trained over pre-processed data and the model is saved for further use in prediction

#### 3.5 Get best model of each

Here we will train various model on each cluster which we will obtain in Data Clustering, and then will try to get best model of each cluster.

#### 3.6 Hyper parameter Tuning

After selecting best model for each cluster, we will do hyper parameter tuning for each selected model, and try to increase performance of the models.

#### 3.7 Model Saving

After performing hyper parameter tuning for models, we will save our models so that we can use them for prediction purpose.

#### 3.8 Cloud Setup

Here We will do cloud setup for model deployment. Here we also create our flask app and user interface and integrate our model with flask app and UI

#### 3.9 Push app to cloud

After doing cloud setup and checking app locally, we will push our app to cloud to start the application.

#### 3.10 Data from client side for prediction purpose

Now our application on cloud is ready for doing prediction. The prediction data which we receive from client side will be exported from DB and further will do same data cleansing process as we have done for training data using modules we will write for training data. Client data will also go along the same process of Exporting data from DB, Data pre-processing, Data clustering and according to each cluster number we will use our saved model for prediction on that cluster.

#### 3.11 Export Prediction to CSV

Finally when we get all the prediction for client data, then our final task is to export prediction to csv file and hand over it to client.

#### 4 Unit Test Cases

|  |  |  |
| --- | --- | --- |
| Test Case Description | Pre-Requisite | Expected Result |
| Verify Whether the Application URL is  accessible to user | 1.Application URL should be defined | Application URL should be  Accessible to 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 be load completely for the user when the  URL 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 successfully login to the application | 1.Application is  accessible    2.User is signed up to the application | User should be able to successfully login to the application |
| Verify whether user is able to see input fields 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 input fields on logging in |
|  |  |  |

Page

|  |  |  |
| --- | --- | --- |
| Verify whether user is able to edit all input 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 input fields |
| 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 to submit the inputs |
| Verify whether user get prediction/output back after submitting 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 their output after submitting the inputs |
| Verify whether the output which user get is accordance to inputs user made. | 1. Application is accessible  2.User is signed up to the application  3.User is logged in to the application | The output should be in accordance with the inputs user made. |
| Verify whether user have option to download their result or not. | 1. Application is accessible  2.User is signed up to the application  3.User is logged in to the application | User should have option to download their output result. |

Page