LOW LEVEL DESIGN (LLD)

i

# **Architecture**

Thyroid Disease Detection System

Written by – Vinayak Verma

**Document Version Control**

**Change Record:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comments** |
| **1.0** | **01-07-2024** | Vinayak verma | **Architecture** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Reviews:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Reviewer** | **Comments** |
|  |  |  |  |

**Approval Status:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Review Date** | **Reviewed By** | **Approved By** | **Comments** |
|  |  |  |  |  |

LOW LEVEL DESIGN (LLD)

## **Contents**

1.Architecture

2. Architecture Description

2.1Data Description

2.2Export Data from DB to CSV for training

2.3Data Preprocessing

2.4Data Classification

2.5Model saving

* 1. Cloud setup
  2. Push App to cloud
  3. Data from client side for prediction

2.8 Prediction

### 2. Architecture

2

**Data**

**Preprocessing**

**Data classi**

**-**

**fication**

**Export data from**

**database**

**to**

**csv**

**for**

**training**

**Start**

**Data**

**from**

**client to be**

**predicted**

**Application**

**Start**

**Prediction**

**Data**

**Preprocessing**

**Data classi**

**-**

**fication**

**Model saving**

**Cloud setup**

**Hyper**

**-**

**parameter**

**tuning**

**End**

**Pushing app**

**to**

**cloud**

**Export data from**

**database**

**to**

**csv**

**to prediction**

**Architecture Description: Thyroid Disease Prediction**

3.1 Data Description

We will utilize the Thyroid Disease Data Set from the UCI Machine Learning Repository. This dataset comprises 7,200 instances across different data batches.

3.2 Export Data from Database to CSV for Training

We will consolidate all batches of data from the database into a single CSV file for model training purposes.

3.3 Data Preprocessing

In this phase, we will explore the dataset and perform exploratory data analysis (EDA) as needed. Based on our findings, we will undertake data preprocessing tasks such as handling null values and removing unnecessary columns. We will create separate modules for these preprocessing steps to ensure they can be utilized during both training and prediction phases.

3.4 Data Classification

We will employ the Random Forest technique for data classification. This method uses an ensemble of decision trees to classify data into predefined categories. The algorithm builds multiple decision trees on different subsets of the data and combines their predictions for the final classification. This approach enhances accuracy, reduces overfitting, and effectively handles complex datasets. The trained random forest model will be saved for future predictions.

3.5 Hyperparameter Tuning

Post-classification, we will perform hyperparameter tuning to optimize the model's predictive performance.

3.6 Model Saving

After classification, we will save the trained models for future prediction use.

3.7 Cloud Setup

We will set up a cloud environment for model deployment. This involves creating a Streamlit app and user interface to integrate our trained models. The Streamlit app will act as the backend, while the user interface will interact with the models.

3.8 Push App to Cloud

Once the cloud setup is complete and the app has been locally tested, we will deploy the entire application to the cloud to make it publicly accessible.

3.9 Data from Client Side for Prediction Purpose

With the application deployed in the cloud, we can start receiving prediction data from clients. The received data will go through the same data cleansing process as the training data, utilizing the previously developed modules. The data will be pre-processed, classified, and fed into the appropriate saved models for predictions.

3.10 Prediction Displayed

After completing all the above steps, the final result, i.e., the thyroid disease prediction, will be displayed on the application.