Low level design (LLD)

Thyroid Disease Detection

Table of Contents

**1 Introduction2**

1.1 What is lor-level document?2

1.2 Scope2

**2 Architecture3**

**3 Architecture Design4**

3.1 Data Description4

3.2 Data Collection4

3.3 Data Processing4

3.4 Model Building4

3.5 Data from user4

3.6 Prediction4

3.7 Database operation4

3.8 Deployment5

**4 Unit test cases5**

**1 Introduction**

* 1. What is Low-Level design document?

The goal of LLD or a low-level design document is to give the internal logical design of the actual program code for Thyroid Disease Detection. 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. Scope

Low-level design 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.

1. **Architecture**

The complete architecture of the project is shown below.

**Start**

**Data from UCI repository**

**Data pre-processing**

**Model building**

**Data from user**

**Application start**

**Pushing app to cloud**

**Cloud setup**

**Prediction**

**Save the user input and output at database**

**Display result**

**End**

1. **Architecture Design**
   1. Data Description

Thyroid disease records supplied by the Garavan Institute and J. Ross Quinlan, New South Wales Institute, Syndney, Australia.1987. The different Classes are increased binding protein, decreased binding protein and negative.

* 1. Data Collection

The dataset is collected from the UCI repository page.

* 1. Data Processing

In this process, raw data is transformed into valid form for the machine learning models. This includes, data cleaning, imputing missing values, removing outliers, drop unnecessary columns, imbalanced data set handling etc.

3.4 Model Building

Multiple models were trained and evaluated based on accuracy score and recall. The best performing models were selected as final model. In our case, it is Decision tree.

3.5 Data from user

Here we will collect patient’s age, sex, TSH, T3, FTI, TT4 and T4U levels.

3.6 Prediction

Once predict button is pressed, the model will predict and then the display the result

3.7 Database operation

The data from the user and corresponding result are inserted into the database. The database used is Cassandra.

3.8 Deployment

We will be deploying the model to AWS. This is a workflow diagram for the thyroid detection.

1. **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 on logging in | 1. Application is  accessible  2. User is signed up to the application | 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 signed up 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 | 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 signed up to the application | 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 signed up to the application | The recommended results should be in accordance to the selections user made |