**Low Level Design (LLD)**

**PRESCRIPTION LABEL READING**

**Revision Number - 1.1**

**Last Date of Revision – 01/01/2024**

**Mary Ara N.**

# Document Control

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** |  | **Version** | **Description** | **Author** |
| 12/22/2023 | 1.0 |  | First draft | Mary Ara N. |
| 01/01/2024 | 1.1 |  | LLD added | Mary Ara N. |

**Contents**

Document Version Control…………………………………………………………………………1

1. Introduction……………………………………………………………………………………….3
   1. What is Low Level Design Document?............................................................................3
   2. Scope…………………………………………………………………………………………….3
2. Problem Statement.…………………………………………………………………………….3
3. Architecture……………………………………………………………………………………...4
4. Architecture Description……………………………………………………………………….5
   1. Data Description………………………………………………………………………………..5
   2. Data Transformation…………………………………………………………………………...5
   3. Data conversion………………………………………………………………………………...5
   4. Text extraction………………………………………………………………………………….5
   5. Text to speech conversion…………………………………………………………………….5
   6. Deployment……………………………………………………………………………………..5

4.7 Testing…………………………………………………………………………………………..5

# 1. Introduction

## 1.1 What is Low Level Design Document?

The goal of the Low-level design document (LLDD) is to give the internal logic design of the actual program code for the Heart Disease Diagnostic Analysis dashboard. LLDD describes the class diagrams with the methods and relations between classes and programs specs. It describes the modules so that the programmer can directly code the program from the document.

**1.2 What is Scope?**

Low-level design (LLD) is a component-level design process that follows a stepby-step refinement process. The 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. Problem Statement

Build a solution that should recognize and identify the text in the prescriptions and should read out the name of medicines and dosage limits to the visually impaired patients.

In order to do this, you should apply OCR techniques to extract the text data from the prescriptions and convert them into speech.

# 3. Architecture

Text to speech conversion

End

Saving output voice in local repo

Testing

Application start

Data selection

Data conversion: image to text

Local storage

Local deployment with FastAPI

text extraction

Data preprocessing

Data import

Start

# Architecture Description

**4.1 Data Description:**

The data used in this project is a prescription label selected from one of the local pharmacies. It is in a .jpg image file format. This data is stored in the local repository.

**4.2 Data transformation:**

Once imported, preprocessing is done using computer vision in order to produce a clearer image.

**4.3 Data Conversion:**

Python’s Tesseract OCR engine is used to convert the pre-processed image to text.

**4.4 Text extraction:**

The relevant prescription text is extracted using Regular Expression.

**4.5 Text to speech conversion:**

Google Cloud’s text-to-speech API is used to generate the audio file which is then stored in the local repository.

**4.6 Local deployment:**

A FastAPI app is written to enable local deployment.

**4.7 Testing:**

Testing is done using Postman.