A PROJECT REPORT ON

Lab 2 Life

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY,

PUNE

IN THE PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING IN COMPUTER SUBMITTED BY,

1. **Ms. Madhura Shete(Exam Seat No.....)**
2. **Ms. Asmita Ghalme(Exam Seat No. .....)**
3. **Mr. Roshan Bhuruk (Exam Seat No. .....)**
4. **Mr. Jitendra Parmar (Exam Seat No. .....)**

UNDER THE GUIDANCE OF

Prof. C. P. Lachake

SINHGAD TECHNICAL EDUCATION SOCIETY

SKN SINHGAD INSTITUTE OF TECHNOLOGY & SCIENCE,

LONAVALA

GAT NO. 309, KUSGAON (BK.) OFF MUMBAI-PUNE EXPRESSWAY, LONAVALA, TAL - MAVAL, DIST - PUNE - 410401. ACADEMIC YEAR: 2025-2026

DEPARTMENT OF COMPUTER

SKN Sinhgad Institute of Technology and Science, Lonavala

Academic Year 2025-26

CERTIFICATE

This is to certify that the project report entitled

Name of Project

SUBMITTED BY,

5. **Ms. Madhura Shete(Exam Seat No.....)**
6. **Ms. Asmita Ghalme(Exam Seat No. .....)**
7. **Mr. Roshan Bhuruk (Exam Seat No. .....)**
8. **Mr. Jitendra Parmar (Exam Seat No. .....)**

Is a bonafide work carried out by them under the supervision of **Prof. C. P. Lachake** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the Degree of Bachelor of Engineering (Computer). The project work has not been earlier submitted to any other institute or university for the award of degree or diploma.

Prof. C. P. Lachake
Internal Guide

Prof. S. P. Gunjal
Head of Department

Dr. K. P. Patil
Principal
SKNSITS, Lonavala

Prof.....
External Examiner

Place: Lonavala

Date:

Acknowledgement

We express our sense of gratitude towards our project guide **Prof. C. P. Lachake** for his/her valuable guidance at every step of study of this project, also his/her contribution for the solution of every problem at each stage.

We are thankful to **Prof. S. P. Gunjal** Head, Department of Computer, all the staff members and project Coordinator **Prof. C. P. Lachake** who extended the preparatory steps of this project. We are very much thankful to respected Principal **Dr. K. P. Patil** for his support and providing all facilities for project.

Finally we want to thank to all our friends for their support & suggestions. Last but not the least we want to express thanks to our family for giving us support and confidence at each and every stage of this project.

Ms. Madhura Shete
Ms. Asmita Ghalme
Mr. Roshan Bhuruk
Mr. Jitendra Parmar

Abstract

The **Lab2Life** project presents an AI-driven healthcare solution that simplifies complex medical laboratory reports into clear, patient-friendly explanations. The system automatically reads lab reports such as blood, urine, or diagnostic test results using **Optical Character Recognition (OCR)** and processes the extracted text through **Natural Language Processing (NLP)** techniques. It then generates easy-to-understand summaries that include explanations of medical terms, normal and abnormal value interpretations, possible causes of deviations, and preventive or lifestyle recommendations.

In addition to simplifying medical data, the system provides **multilingual support**—allowing reports to be translated into regional languages such as Marathi, Hindi, and English—enhancing accessibility for patients from diverse linguistic backgrounds. A unique feature of Lab2Life is the **Doctor Verification Module**, enabling certified medical professionals to review, edit, and approve AI-generated summaries, ensuring accuracy and trustworthiness.

The system architecture integrates multiple technologies: **OCR (Tesseract/EasyOCR)** for text extraction, **Transformer-based NLP models (BioGPT/T5)** for summarization, and a **web application framework (React.js, FastAPI, MongoDB)** for seamless user interaction. Through this approach, **Lab2Life** bridges the gap between medical data and patient comprehension, empowering individuals to understand their health reports, reduce anxiety, and take informed steps toward better healthcare management.

Keywords:
AI in Healthcare, Medical Report Summarization, Natural Language Processing (NLP), Optical Character Recognition (OCR), Biomedical Text Analysis, Patient-Friendly Reports, Doctor Verification, Multilingual Translation, Healthcare Automation, Lab2Life System

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

In today's digital healthcare environment, patients frequently receive laboratory and diagnostic reports filled with complex medical terminologies, abbreviations, and numeric results that are often difficult to interpret without medical expertise. This creates a communication gap between healthcare professionals and patients, leaving many individuals dependent solely on doctors for understanding their health conditions. Such dependency can delay awareness, increase anxiety, and reduce the patient's ability to take proactive steps toward maintaining good health.

To address this issue, the **Lab2Life** system introduces an **AI-powered platform** that automatically analyzes medical reports and presents them in a simple, human-understandable format. The system reads digital or scanned lab reports using **Optical Character Recognition (OCR)** and processes the extracted information through **Natural Language Processing (NLP)** and **Machine Learning (ML)** models. It then generates summaries that explain each parameter, highlight abnormal results, provide possible causes, and recommend lifestyle modifications or precautions.

A distinctive feature of **Lab2Life** is the inclusion of a **Doctor Verification Module**, which allows certified medical professionals to validate and approve AI-generated explanations before sharing them with patients. This ensures accuracy, trustworthiness, and ethical use of artificial intelligence in healthcare communication.

By offering **multilingual support**, **data privacy**, and **web-based accessibility**, Lab2Life empowers patients to better understand their lab results in their preferred language, promoting health literacy and self-awareness. The project thus bridges the knowledge gap between raw medical data and patient understanding, contributing to more informed and engaged healthcare participation.

1.2 Motivation

In today's technology-driven healthcare environment, patients often receive medical reports filled with complex numerical data and medical jargon that are difficult to interpret without expert assistance. While doctors play a vital role in explaining results, the heavy dependency on them for basic report interpretation often leads to **delayed awareness, anxiety, and limited health literacy** among patients.

Many individuals receive blood or diagnostic test reports indicating abnormal results (e.g., high cholesterol, low hemoglobin) but fail to understand their implications due to the absence of simplified explanations. Existing digital healthcare solutions primarily focus on **storing or sharing medical data**, not on **interpreting and translating** it into a form patients can easily comprehend.

The motivation behind developing **Lab2Life** is to bridge this crucial gap between **medical data and human understanding**. By integrating **Artificial Intelligence (AI)** and **Natural Language Processing (NLP)**, the system can automatically read, summarize, and explain medical reports in a patient-friendly manner. Additionally, with **multilingual translation** and **doctor verification**, the project ensures that accurate, accessible, and verified medical information reaches every patient—irrespective of their technical or linguistic background.

This initiative is inspired by the vision of **making healthcare more inclusive and transparent** by empowering individuals with knowledge about their own health. Through **Lab2Life**, patients can gain confidence, reduce dependency, and actively participate in their healthcare journey with a clearer understanding of their medical data.

1.3 Objectives

The main objective of the **Lab2Life** system is to create an **AI-powered healthcare platform** that interprets, summarizes, and explains medical reports in simple, understandable language

while maintaining accuracy through doctor verification. The project aims to make healthcare data accessible and comprehensible for everyone, regardless of their medical knowledge.

The detailed objectives are as follows:

To develop an intelligent report interpretation system using Artificial Intelligence (AI) and Natural Language Processing (NLP).
Design and train an AI model capable of reading lab and diagnostic reports, detecting key health parameters, and generating human-understandable summaries.

To implement Optical Character Recognition (OCR) for automated text extraction from medical reports.
Enable the system to process uploaded PDF or image-based reports and accurately extract information such as test names, values, and reference ranges.

To generate patient-friendly explanations with contextual health insights.
Include the meaning of each parameter, normal and abnormal ranges, possible causes, preventive measures, and doctor recommendations.

To provide multilingual support for improved accessibility.
Translate summaries into regional languages such as **Marathi, Hindi, and English**, ensuring inclusivity for non-English-speaking users.

To integrate a Doctor Verification Module.
Allow certified medical professionals to review and approve AI-generated reports before presenting them to patients to ensure trust and authenticity.

To design a secure, user-friendly web platform.
Build a responsive web interface using **React.js, FastAPI, and MongoDB**, allowing users to upload, view, and manage their reports safely.

To ensure data privacy and security.
Incorporate encryption, authentication, and secure database storage to protect sensitive medical information.

To evaluate system accuracy and performance.
Assess the quality of OCR extraction, summarization accuracy, translation reliability, and user satisfaction through testing and feedback.

Chapter 2

Literature Survey

Sr. No	Title of paper/study	Description with seed idea	Technology used	Merits/demerits
1	BioGPT: Generative Pre-trained Transformer for Biomedical Text Generation	Explores using domain-specific GPT models to understand and summarize biomedical research and reports. Seed idea: applying large language models to medical literature for knowledge generation.	Transformer-based NLP (GPT architecture), Biomedical corpus pre-training	**Merit:** Generates medically coherent summaries and context-rich responses. **Demerit:** Requires high-end GPU resources and continuous fine-tuning for accuracy.

2	Med-PaLM: Large Language Models Encode Clinical Knowledge	Focuses on aligning LLMs like PaLM with clinical reasoning for medical question answering and decision support. Seed idea: integrating factual accuracy and reasoning into healthcare chatbots.	Large Language Models (LLMs), Reinforcement Learning with Human Feedback (RLHF	**Merit:** Understands complex clinical questions and provides contextual explanations. **Demerit:** May produce hallucinated responses without supervision.
3	ExplainMed: Patient-Friendly Explanation of Medical Reports Using NLP	Converts complex diagnostic language into patient-friendly summaries. Seed idea: enhancing patient comprehension using AI text simplification.	NLP pipelines, transformer-based summarization	Merit: Improves patient health literacy. Demerit: Limited support for regional languages and cultural nuances.
4	OCR-Based Medical Report Analyzer for Diagnostic Assistance	Automates the digitization of printed or handwritten lab reports using OCR. Seed idea: enabling machine understanding of physical medical records.	Optical Character Recognition (Tesseract, EasyOCR), Data extraction and classification	**Merit:** Enables automation and digitization of manual reports. **Demerit:** Accuracy drops with noisy or low-quality scans.

5	Deep Learning for Diabetic Retinopathy Detection	Uses CNNs to detect diabetic retinopathy from retinal images. Seed idea: applying AI vision models for disease prediction.	Convolutional Neural Networks (CNNs), Image preprocessing	**Merit:** Achieves doctor-level accuracy in specific imaging tasks. **Demerit:** Focused only on image-based diagnostics, not text-based medical data.
6	MIMIC-IV Clinical Dataset: A Foundation for AI in Healthcare	Presents an open medical dataset for AI model training in diagnosis and report understanding. Seed idea: creating reproducible AI models using large-scale hospital data.	Data aggregation, standardization, NLP and ML pipelines	**Merit:** Enables large-scale, transparent AI research. **Demerit:** Data privacy and access restrictions.
7	A Multilingual Transformer for Healthcare Communication	Proposes multilingual NLP systems for healthcare text translation to improve accessibility. Seed idea: integrating multilingual AI for medical text comprehension.	MarianMT, mBERT, neural machine translation (NMT)	**Merit:** Enhances accessibility for non-English speakers. **Demerit:** Possible loss of medical accuracy in translation.

8	Lab2Life: An AI-Based System for Simplified Medical Report Interpretation and Patient Empowerment	Proposed system that reads, summarizes, translates, and verifies medical reports through doctor collaboration. Seed idea: end-to-end patient-centered AI healthcare platform.	OCR (Tesseract), NLP (BioGPT/T5), Translation APIs, FastAPI, MongoDB, Doctor Verification	**Merit:** Comprehensive, accurate, and multilingual. **Demerit:** Dependent on domain dataset and doctor verification for reliability.

Table 2.0.1: **Literature Survey**

Chapter 3

Problem Statement

In the current healthcare landscape, most patients receive diagnostic and laboratory reports containing complex medical terminology, abbreviations, and numerical data that are difficult to understand without medical expertise. This lack of understanding often results in confusion, anxiety, and delayed responses to potential health issues.

While some mobile health applications provide report storage or basic result tracking, they fail to interpret the data meaningfully or provide patient-friendly explanations. Furthermore, existing systems do not offer multilingual support or professional verification, leaving patients uncertain about the accuracy of AI-generated outputs.

Patients in non-urban areas or those with limited medical knowledge face even greater challenges due to language barriers and the absence of easy-to-understand explanations for abnormal test results.

Therefore, there is a need for an AI-powered system that can automatically extract, interpret, and explain medical reports in simple language, supported by doctor verification and multilingual translation, ensuring both accuracy and accessibility.

The proposed **Lab2Life** system addresses these challenges by providing:

Automated medical report interpretation using OCR and NLP.
Simplified, patient-friendly summaries highlighting key findings.
Multilingual explanations in English, Hindi, and Marathi.
Doctor verification to ensure accuracy and trustworthiness.
Through this approach, **Lab2Life** bridges the gap between raw medical data and patient comprehension, empowering individuals to make informed healthcare decisions.

Chapter 4

Project Requirement Specification

4.1 Hardware Requirements

Component	Minimum Requirement	Recommended Requirement

Processor (CPU)	Dual Core (Intel i3 / AMD Ryzen 3)	Intel i5 / Ryzen 5 or higher
RAM	4 GB	8 GB or more
Storage	500 MB free space	1 GB or more
Display	1280 × 720 pixels	Full HD (1920 × 1080 pixels)
Internet Connection	Required for API access and cloud storage	Stable high-speed internet
Mobile Device (for testing)	Android smartphone (Android 8.0 or above)	Android 10+ with 4 GB RAM
Camera	5 MP or higher (for scanning reports)	12 MP or higher for better OCR accuracy

4.2 Software Requirements

Component	**Specification / Description**
Operating System (Development Machine)	Windows 10 / 11, macOS, or Ubuntu
Programming Languages	Python (AI/NLP backend), JavaScript (React.js frontend)
Frameworks & Libraries	FastAPI (backend), React.js (frontend), Tesseract OCR, HuggingFace Transformers, BioGPT / T5 models
Database	MongoDB (for storing user data, reports, and summaries)
APIs & Tools	Google Translate API / MarianMT for multilingual translation
AI/ML Environment	Python 3.10+, TensorFlow / PyTorch
Development Tools	Visual Studio Code, Jupyter Notebook
Version Control	Git / GitHub
Browser Compatibility	Google Chrome, Mozilla Firefox, Microsoft Edge

Deployment Platform	Render / Heroku / Google Cloud Platform
Testing Tools	Postman (API testing), PyTest (model validation)

Chapter 5

System Proposed Architecture

5.1 Architecture Diagram

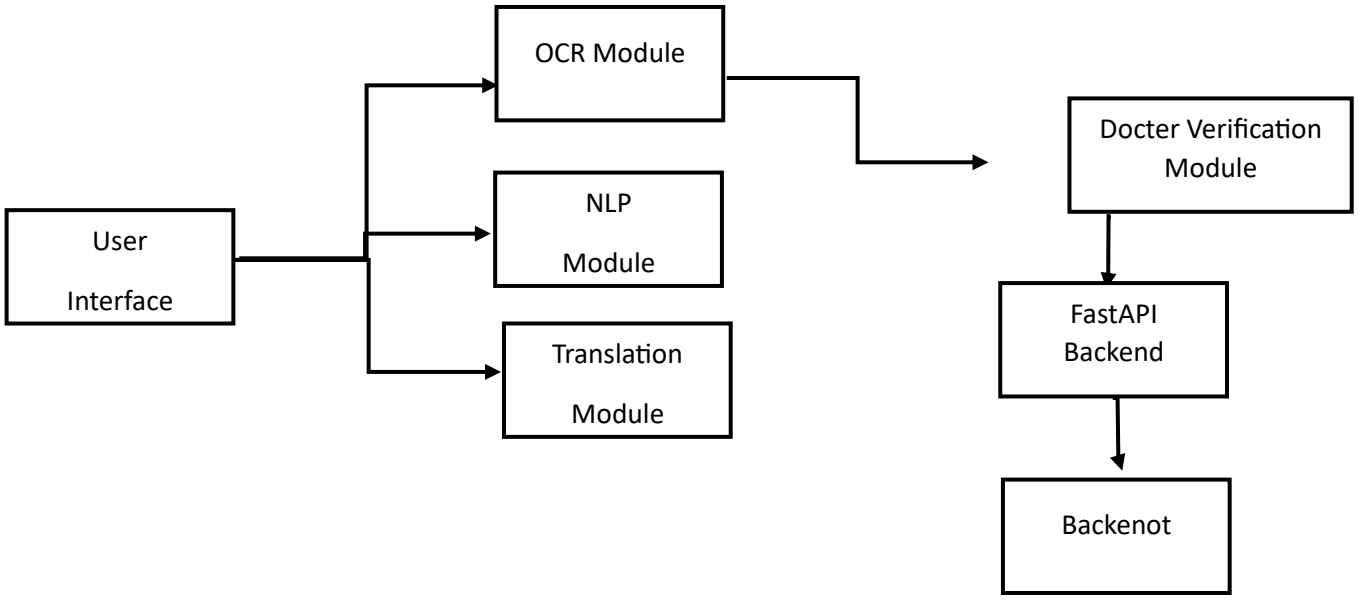


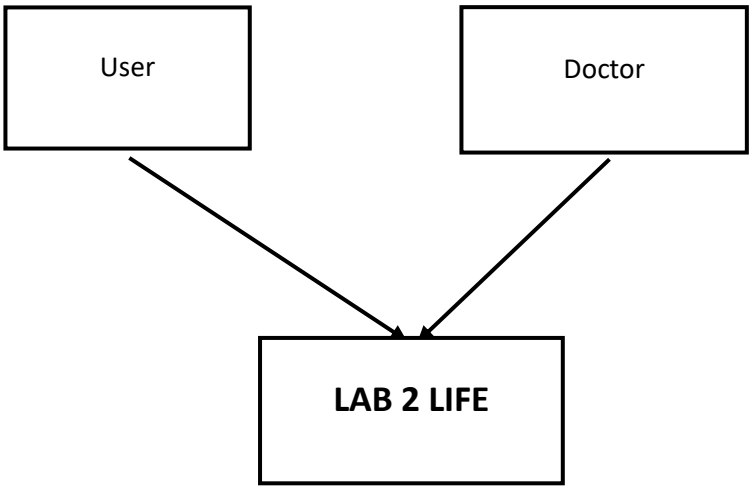
Figure 1 System Proposed Architecture

Chapter 6

High Level Design of Project

6.1 DFD

6.1.1 Level-0 DFD



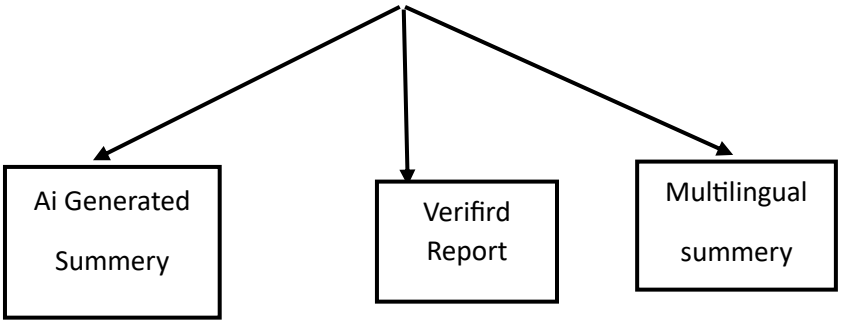


Figure 2 Level-0 DFD

6.1.2 Level-1 DFD

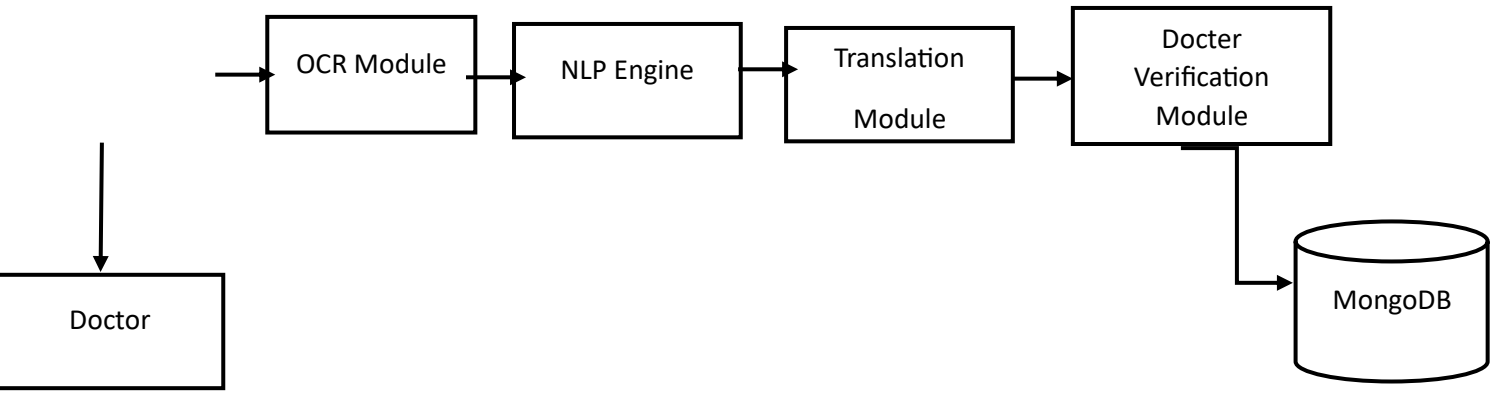


Figure 3 Level-1 DFD

6.2 UML

6.2.1 Use-Case Diagram

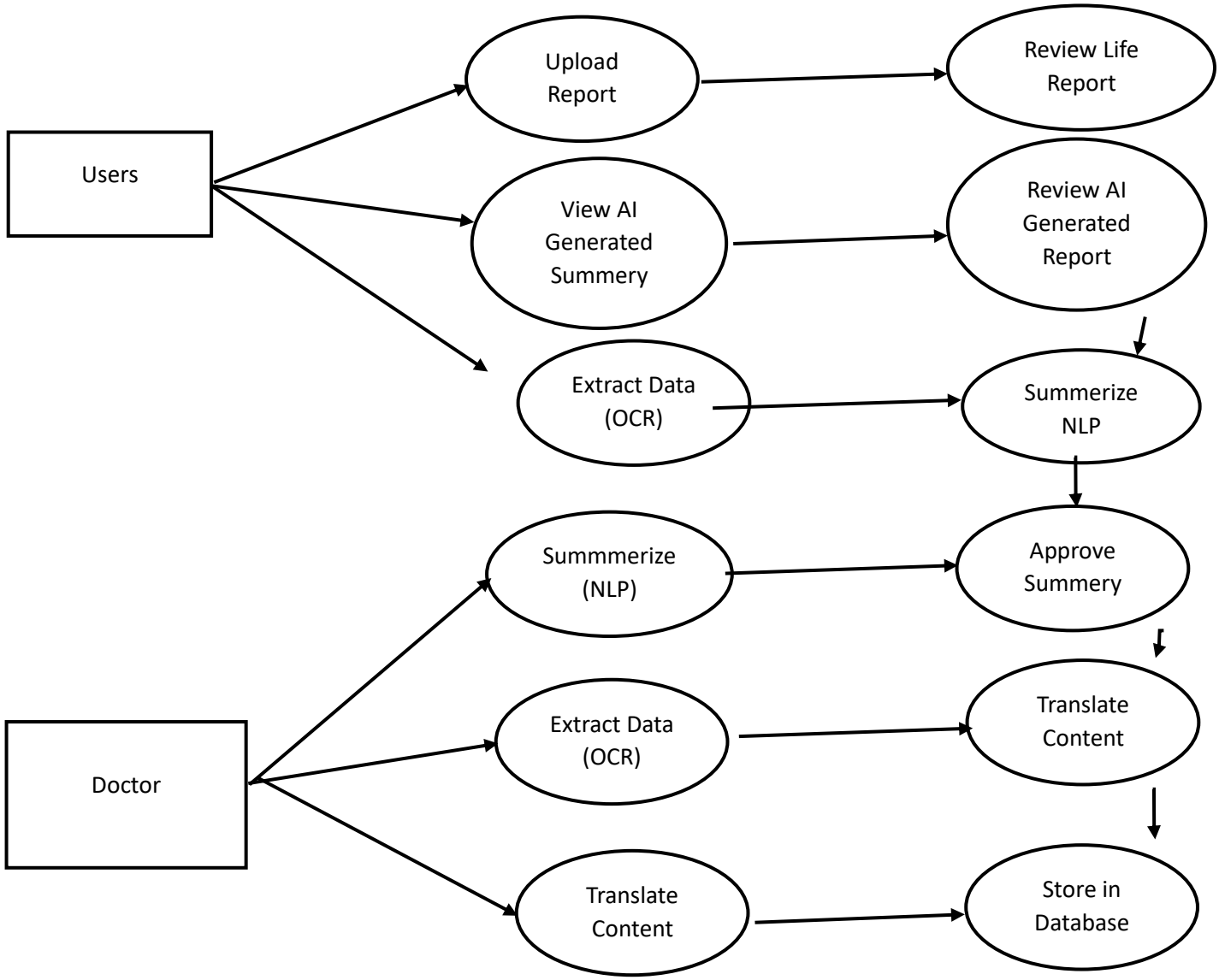
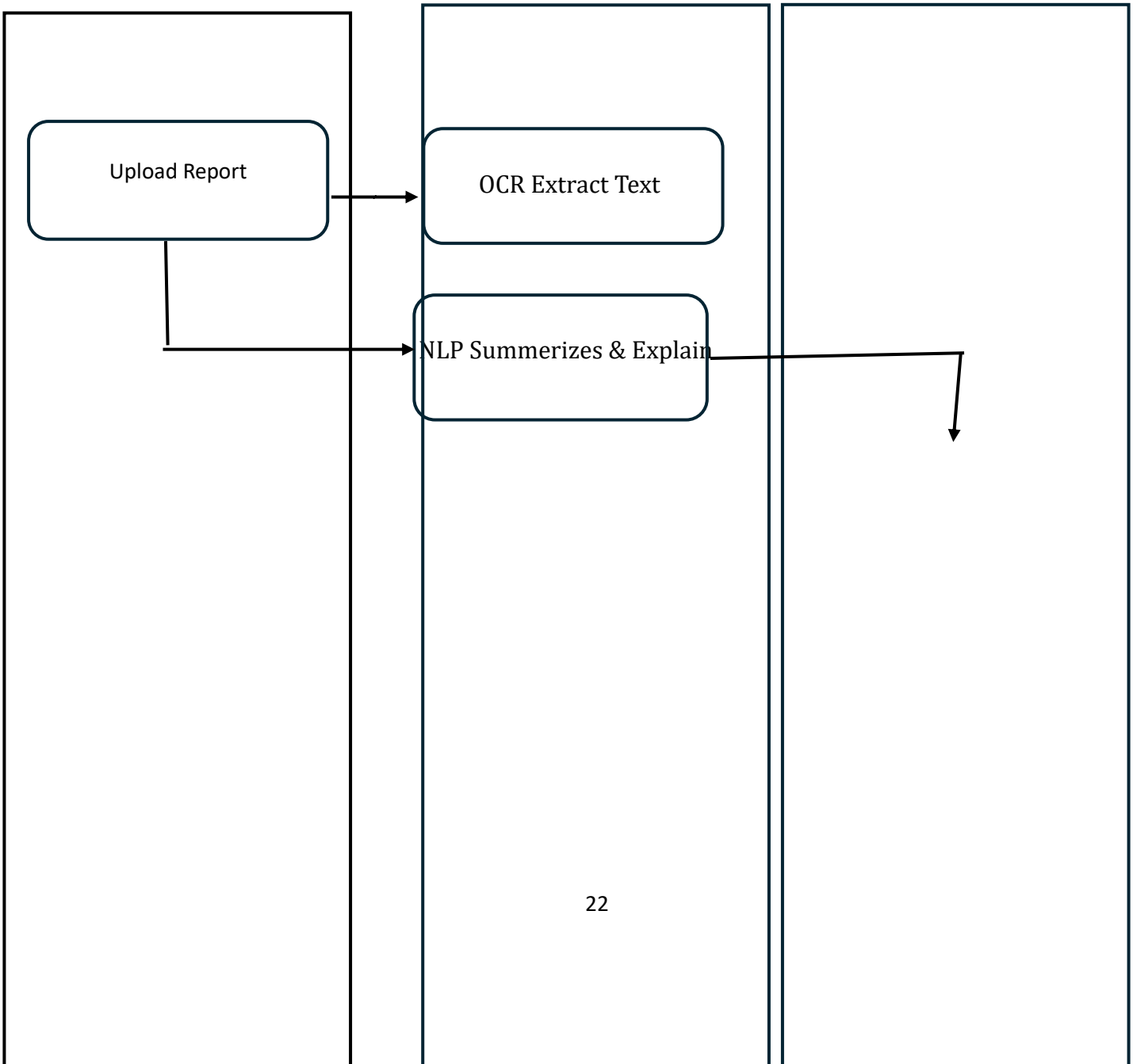


Figure 4 Use-Case Diagram

6.2.2 Activity Diagram



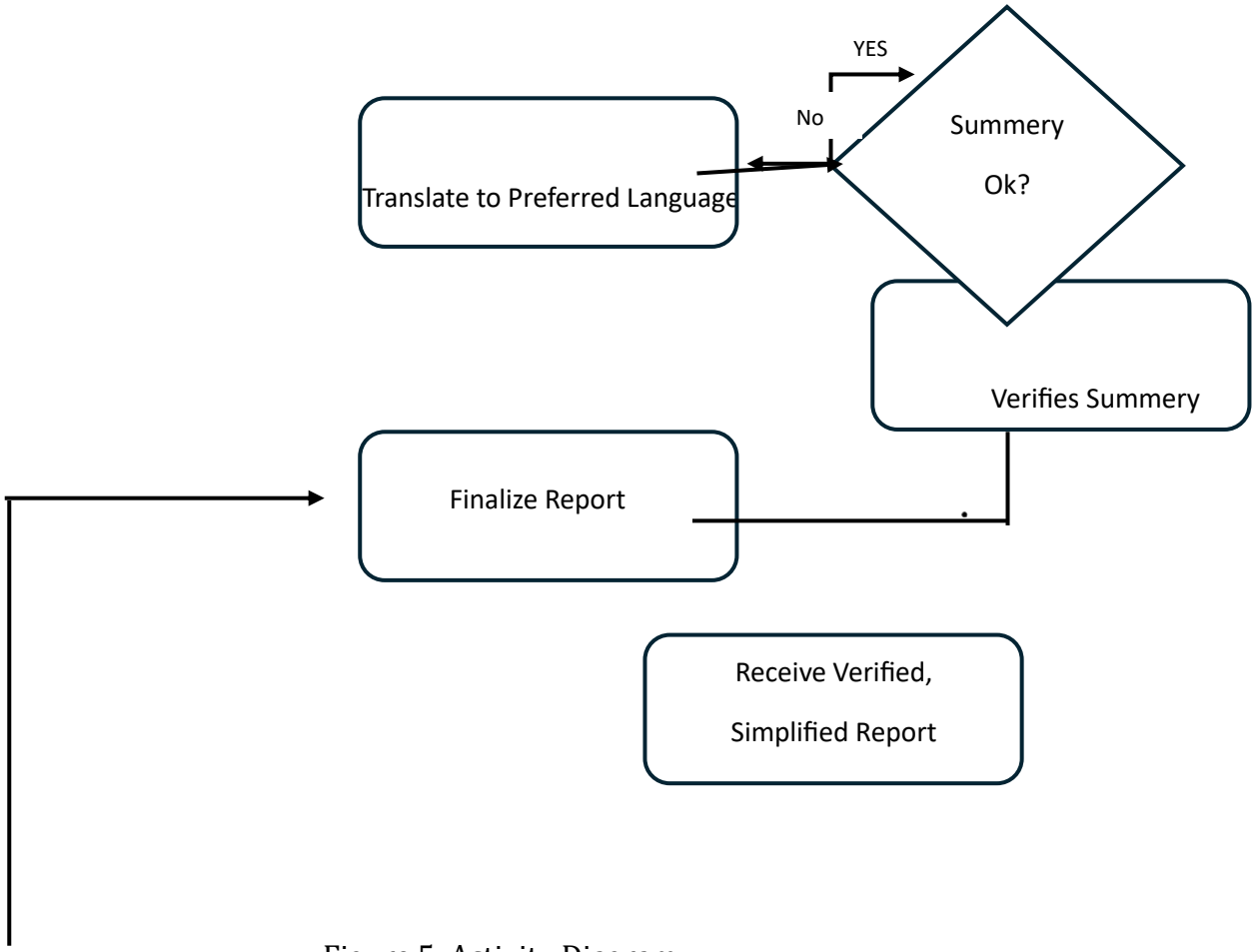
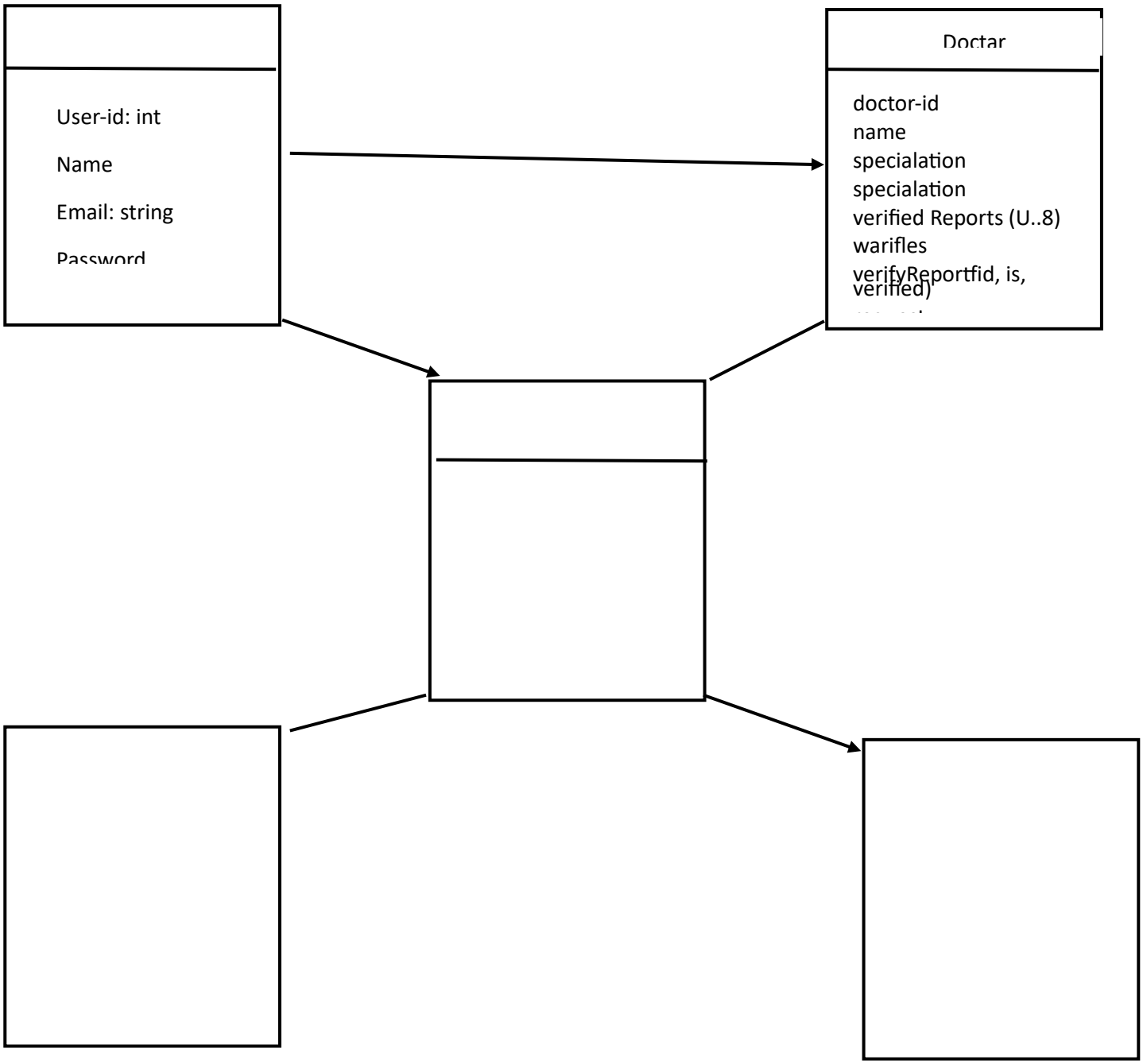


Figure 5 Activity Diagram

6.2.2 Class Diagram



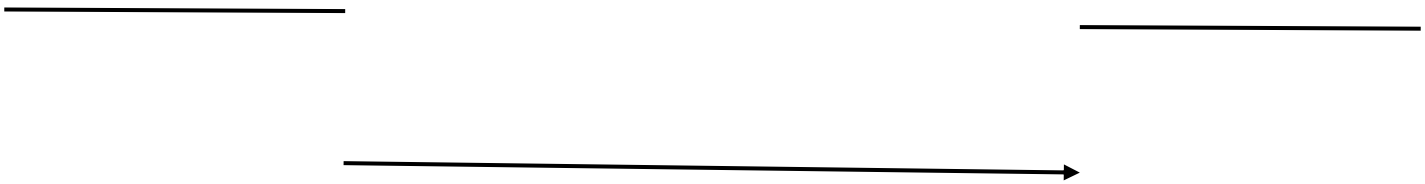


Figure 6 Class Diagram

6.2.3 Sequence Diagram

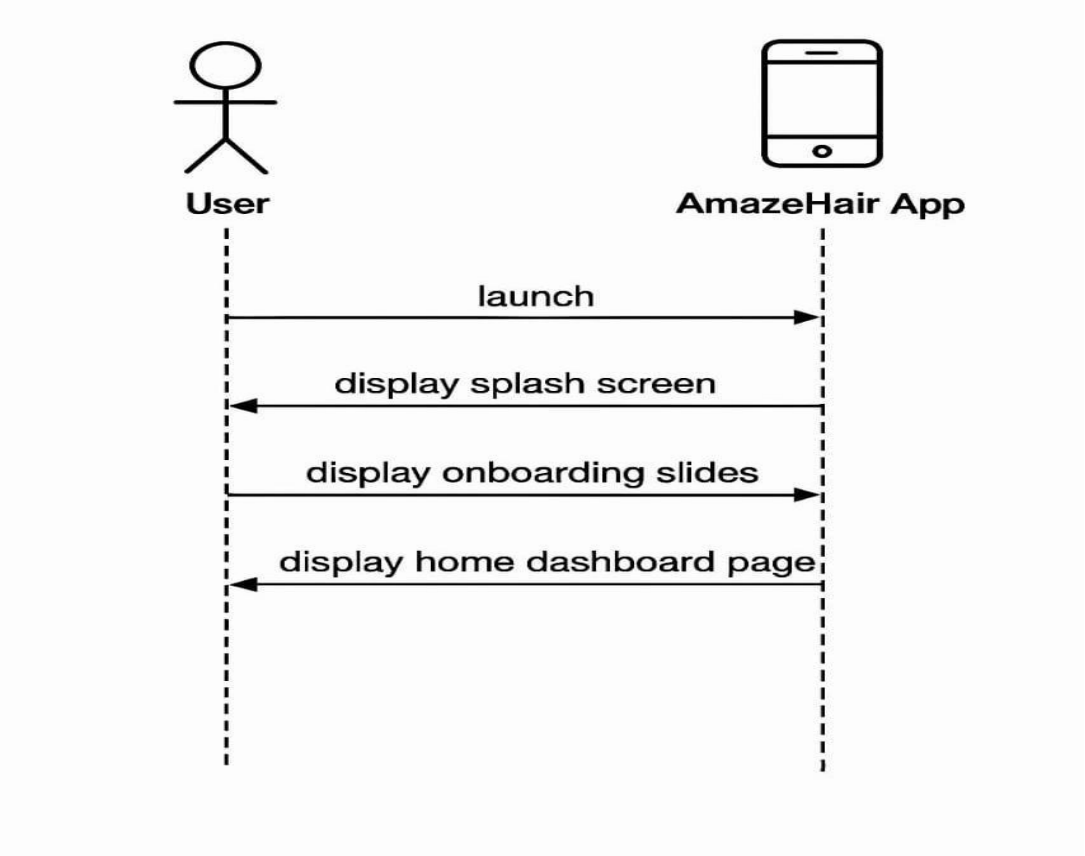


Figure 7 Sequence Diagram

6.2.4 ER Diagram

AmazeHair ER Diagram

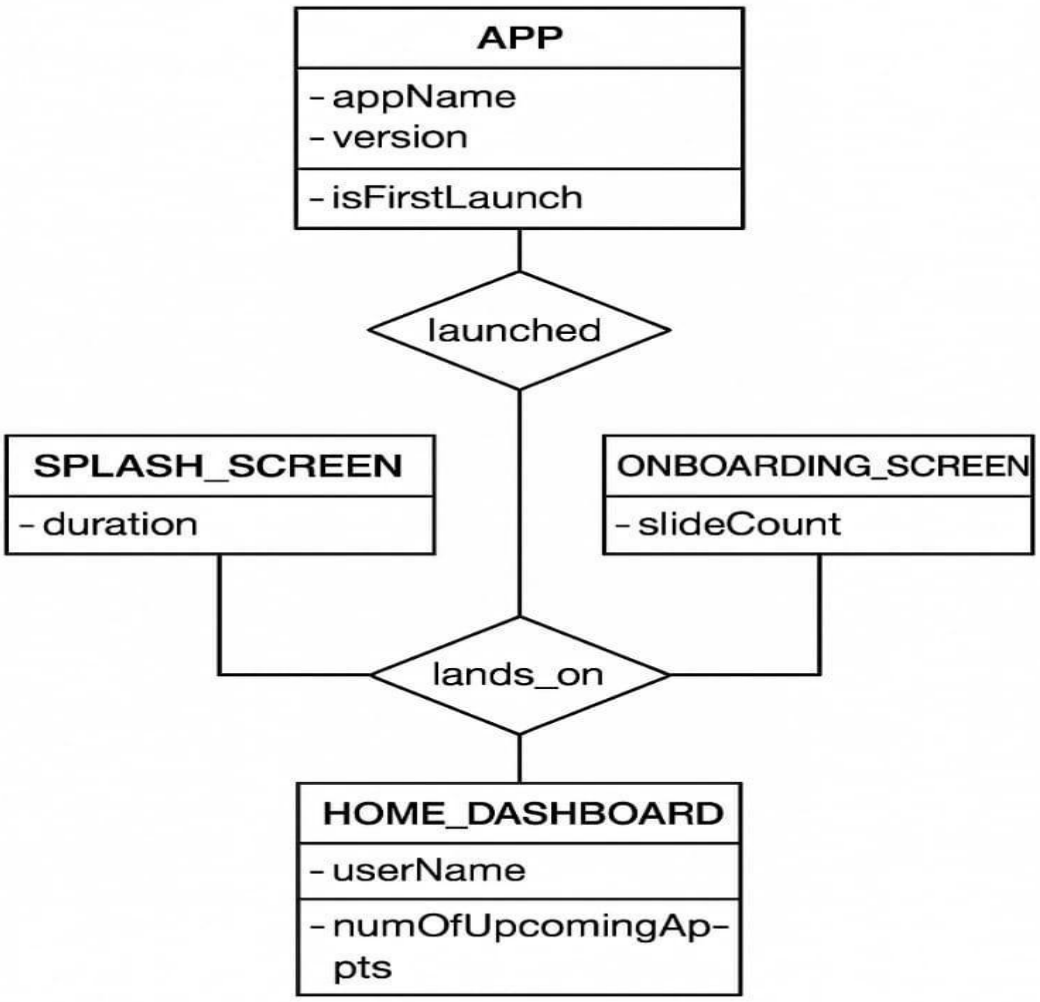


Figure 8 ER Diagram

Chapter 7

System Implementation

```
from fastapi import FastAPI, File, UploadFile, Form
from fastapi.middleware.cors import CORSMiddleware
import pdfplumber
import os
from groq import Groq
from dotenv import load_dotenv
load_dotenv()

app = FastAPI()

app.add_middleware(
   CORSMiddleware,
   allow_origins=["*"],  # Allow all origins (fine for local dev)
   allow_credentials=True,
   allow_methods=["*"],
   allow_headers=["*"],
)
```

```
client = Groq(api_key=os.getenv("GROQ_API_KEY"))
UPLOAD_FOLDER = "uploads"
os.makedirs(UPLOAD_FOLDER, exist_ok=True)

def extract_text_from_pdf(file_path: str) -> str:
    """Extract all text from a PDF file using pdfplumber."""
    text = ""
    with pdfplumber.open(file_path) as pdf:
        for page in pdf.pages:
            page_text = page.extract_text() or ""
            text += page_text + "\n"
    return text.strip()

def generate_summary(text: str) -> str:
    """Generate a medical summary from the extracted lab report text."""
    prompt = f"""
You are a medical expert AI. Summarize this lab report for a patient in simple, clear language.
Include:
1. Key test results (with values and reference ranges)
2. Meaning or interpretation of each result
3. Overall health summary and recommendations.

Lab Report:
{text}
"""
    try:
        response = client.chat.completions.create(
            model="llama-3.3-70b-versatile",  # ☑ Updated latest model
            messages=[
                {"role": "system", "content": "You are a helpful medical assistant."},
                {"role": "user", "content": prompt},
```

```
            ],
            temperature=0.4,
        )
        summary = response.choices[0].message.content.strip()
        return summary
    except Exception as e:
        print("Summary Error:", e)
        return f"Summary generation failed: {e}"

def translate_summary(text: str, target_lang: str) -> str:
    """Translate text into the target language using Groq LLM."""
    if target_lang == "en":
        return text  # No translation needed

    translation_prompt = f"Translate the following medical lab report summary into
{target_lang}. Make it sound natural and medically accurate:\n\n{text}"
    try:
        response = client.chat.completions.create(
            model="llama-3.3-70b-versatile",  # ☑ Updated latest model
            messages=[
                {"role": "system", "content": "You are a professional medical translator."},
                {"role": "user", "content": translation_prompt},
            ],
            temperature=0.3,
        )
        translated_text = response.choices[0].message.content.strip()
        return translated_text
    except Exception as e:
        print("Translation Error:", e)
        return f"Translation failed: {e}"
```

```
@app.post("/upload-report")
async def upload_report(file: UploadFile = File(...), language: str = Form("en")):
    """Main endpoint to upload, summarize, and translate lab reports."""
    try:
        # Save uploaded file temporarily
        file_path = os.path.join(UPLOAD_FOLDER, file.filename)
        with open(file_path, "wb") as f:
            f.write(await file.read())

        pdf_text = extract_text_from_pdf(file_path)
        if not pdf_text.strip():
            return {"summary": "No readable text found in the PDF file."}
        english_summary = generate_summary(pdf_text)

        final_summary = translate_summary(english_summary, language)

        return {"summary": final_summary}

    except Exception as e:
        print("Error:", e)
        return {"summary": f"An error occurred: {str(e)}"}

@app.get("/")
def root():
    return {"message": "☑ Multilingual Lab Report Summarizer API running successfully with
LLaMA 3.3!"}
```

7.2 Algorithm

1. OCR Text Extraction Algorithm

Purpose:
To extract textual data (test names, values, and units) from scanned or PDF medical reports.

Algorithm Steps:

1. Input the report image or PDF file.
2. Convert the report into grayscale for noise reduction.
3. Apply **Optical Character Recognition (OCR)** using **Tesseract** or **EasyOCR**.
4. Detect text blocks, align text orientation, and segment into lines.
5. Extract text and store it in structured form (e.g., "Hemoglobin: 11.2 g/dL").
6. Pass the extracted text to the NLP processing module for interpretation.

Output:
Structured medical data ready for NLP-based analysis.

◇ 2. NLP-Based Report Interpretation Algorithm

Purpose:
To identify medical parameters and generate simplified interpretations from extracted text.

Algorithm Steps:

1. Receive structured text from OCR output.
2. Preprocess text (remove units, special characters, and redundant spaces).
3. Tokenize sentences and apply **named entity recognition (NER)** to identify medical terms and numeric values.
4. Match extracted terms with medical reference ranges from a knowledge base.
5. Detect abnormal values (e.g., higher or lower than normal).
6. Generate contextual explanations (e.g., "Low hemoglobin may indicate anemia").

Output:
Interpreted report data with condition analysis and medical insight.

◇ 3. Text Summarization Algorithm (Transformer-Based)

Purpose:
To summarize the entire medical report into a short, patient-friendly explanation.

Algorithm Steps:

1. Input the interpreted data from the NLP module.
2. Use a **Transformer-based summarization model** (e.g., **T5**, **BioGPT**) trained on biomedical datasets.
3. Encode input text into embeddings and pass through attention layers.
4. Decode relevant information into concise, natural-language sentences.

5. Structure output summary under predefined sections — “Findings,” “Causes,” “Precautions,” and “Doctor Advice.”

Output:
Readable summary of the report in simple medical language.

◇ 4. Translation Algorithm

Purpose:
To convert the English summary into regional languages (Marathi and Hindi) for patient accessibility.

Algorithm Steps:

1. Input the English summary from the summarization module.
2. Pass it to a **multilingual translation model** such as **MarianMT** or **Google Translate API**.
3. Ensure domain-specific medical terminology remains unchanged.
4. Verify translated output for context consistency.
5. Display the multilingual summary on the user interface.

Output:
Multilingual report summary preserving medical meaning and readability.

◇ 5. Doctor Verification Algorithm

Purpose:
To ensure the accuracy of AI-generated summaries through medical professional validation.

Algorithm Steps:

1. Store the AI-generated summary in a "Pending Verification" database.
2. Allow the doctor to access the summary via the verification interface.
3. Doctor reviews, edits, or adds remarks to ensure correctness.
4. If approved → mark the report as **"Doctor Verified."**
5. Update the verification status in the main database and make it visible to the user.

Output:
Doctor-approved report summary with verification log and timestamp.

7.3 Methodologies

The development of Lab2Life follows the Agile Software Development Model, ensuring flexibility, continuous testing, and feedback at every stage.

Phases:

1. Requirement Analysis:
   Identified user needs (patients & doctors).
   Defined modules: OCR, NLP, Translation, Doctor Verification, Web UI.
2. Design Phase:
   Prepared architecture, DFDs, UML diagrams, and database schema.

Designed wireframes for user and doctor dashboards.

3. Development Phase:

   Implemented OCR using *Tesseract*, NLP summarization using *BioGPT/T5*.

   Built frontend with *React.js* and backend with *FastAPI*.

4. Testing Phase:

   Conducted unit, integration, and user testing.

   Verified accuracy, translation quality, and doctor verification flow.

5. Deployment Phase:

   Deployed system on *Google Cloud / Render*.

   Integrated secure APIs and database (MongoDB).

6. Maintenance Phase:

   Collected feedback, improved AI model accuracy, and enhanced multilingual support.

7.4 Protocols Used

The **Lab2Life** system uses several communication and security protocols to ensure safe data exchange and reliable operation between modules.

1. **HTTP / HTTPS Protocol:**
   Used for secure client-server communication between frontend and backend using SSL/TLS encryption.

2. **RESTful API Protocol:**
   Enables interaction between React.js frontend and FastAPI backend with lightweight JSON data transfer.

3. **OAuth 2.0 / JWT (JSON Web Token):**
   Handles user authentication and session management to prevent unauthorized access.

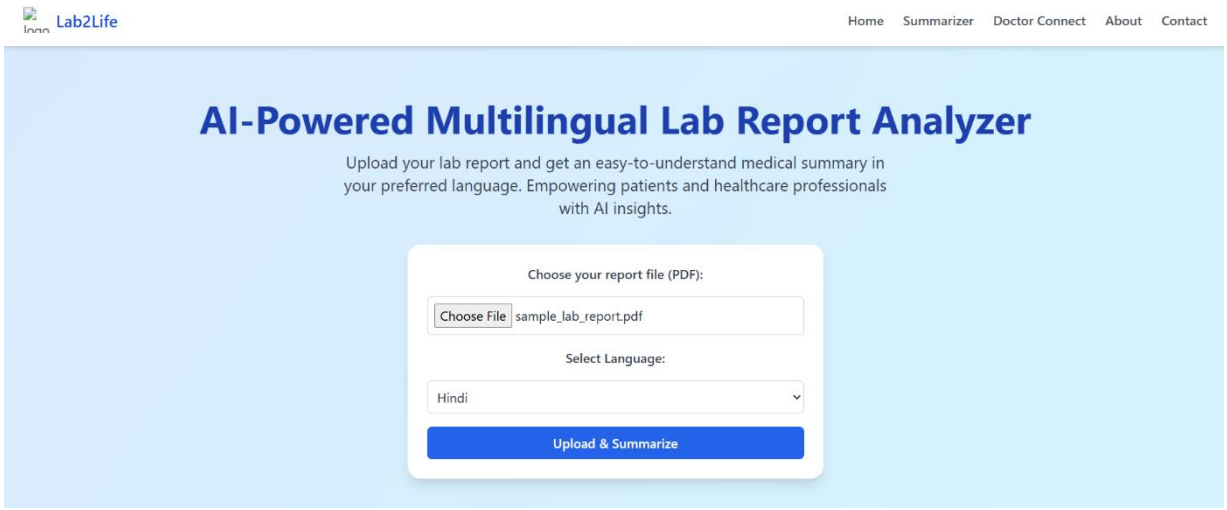
4. **Database Communication Protocol:**
   FastAPI communicates with **MongoDB** over TCP/IP for reliable data storage and retrieval.

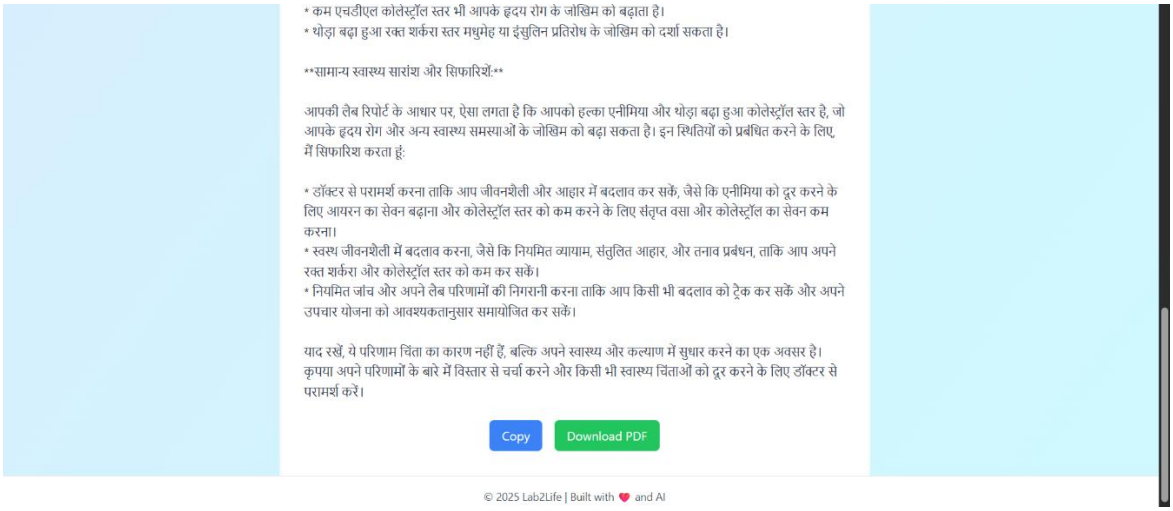
5. **Encryption Protocols:**
   All sensitive information (reports, login data) is encrypted using **AES / SHA-256** algorithms.

Chapter 8

Working Modules

8.1 GUI of Working Module





Chapter 9

9.1 Project Plan Gantt Chart:

Month	Phase / Task Description	Activities	Expected Outcome / Deliverables
Month 1 (July)	Project Selection & Topic Finalization	Finalize domain, topic “Lab2Life”, team formation, and feasibility study.	Approved project proposal and group formation.
Month 2 (August)	Requirement Analysis & Literature Survey	Study related research papers, define scope, and prepare SRS document.	SRS document and literature survey report.
Month 3 (September)	System Design	Prepare architecture, DFDs, UML diagrams, and database schema.	System design diagrams and architecture document.
Month 4 (October)	Dataset Preparation & OCR Module	Collect sample medical reports, implement OCR using Tesseract/EasyOCR.	Working OCR model for text extraction.
Month 5 (November)	NLP Model Development	Build and train summarization model (T5/BioGPT).	AI model for report summarization.
Month 6 (December)	Explanation & Contextual Insights	Generate causes, precautions, and recommendations.	Context-aware medical explanation module.
Month 7 (January)	Multilingual Translation Module	Integrate translation (Marathi/Hindi) using MarianMT/Google Translate API.	Multilingual explanation module.
Month 8 (February)	Doctor Verification Module	Develop interface for doctors to review and approve reports.	Functional doctor verification system.
Month 9 (March)	Web Application Development	Build frontend (React.js) and backend (FastAPI), integrate all modules.	Fully functional web platform.
Month 10 (April)	System Integration & Security Implementation	Combine modules, add authentication and encryption.	Secure integrated prototype.
Month 11 (May)	Testing & Evaluation	Conduct unit and user testing; gather feedback from doctors.	Tested and verified system.

Month 12 (June)	Documentation & Final Presentation	Compile report, prepare presentation, and submit final project.	Final project report and viva presentation.

Conclusion

The **Lab2Life** project successfully demonstrates how Artificial Intelligence (AI) and Natural Language Processing (NLP) can simplify complex medical reports into clear, understandable summaries for patients.

By integrating **OCR** for text extraction, **Transformer-based NLP models** for summarization, and **multilingual translation**, the system bridges the communication gap between medical data and patient comprehension.

The addition of a **Doctor Verification Module** ensures that every AI-generated report is accurate, trustworthy, and medically approved before being shared with users. This approach not only enhances transparency in healthcare communication but also empowers patients to take informed decisions about their health without relying entirely on technical expertise.

In real-world use, **Lab2Life** acts as a smart healthcare assistant—helping patients interpret their reports, understand test results, and follow preventive recommendations. It contributes to improved **health awareness, accessibility, and inclusivity**, especially for non-medical and regional-language users.

Through this project, we have demonstrated the potential of combining **AI, web technologies, and multilingual communication** to make healthcare more understandable, reliable, and patient-centered.

.

References

[1] V. Gulshan, et al., "Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs," *JAMA*, vol. 316, no. 22, pp. 2402–2410, 2016.

[2] G. Quellec, K. Charrière, Y. Boudi, B. Cochener, and M. Lamard, "Deep image mining for diabetic retinopathy screening," *Medical Image Analysis*, vol. 39, pp. 178–193, 2017.

[3] A. Rakhlin, A. Shvets, V. Iglovikov, and A. A. Kalinin, "Deep convolutional neural networks for diabetic retinopathy detection," *arXiv preprint arXiv:1712.07019*, 2017.

[4] Y. Peng, S. Yan, and Z. Lu, "Transfer learning in biomedical natural language processing: an evaluation of BERT and BioBERT models," *Bioinformatics*, vol. 35, no. 14, pp. 2731–2737, 2019.

[5] Z. I. Attia, et al., "An artificial intelligence-enabled ECG algorithm for the identification of patients with atrial fibrillation during sinus rhythm: a retrospective analysis of outcome prediction," *The Lancet*, vol. 394, no. 10201, pp. 861–867, 2019.

[6] R. Gupta and R. Chhikara, "Performance evaluation of ensemble and kernel-based machine learning models for medical data classification," *Procedia Computer Science*, vol. 173, pp. 321–329, 2020.

[7] C. Raffel, et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020.

[8] R. Valarmathi and N. Vijayabhanu, "Comparative analysis of convolutional neural network architectures for medical image classification," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 4, pp. 54–62, 2021.

[9] E. Tjoa and C. Guan, “A survey on explainable artificial intelligence (XAI): towards medical transparency,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 11, pp. 4793–4813, 2021.

[10] W. L. Alyoubi, W. M. Shalash, and M. F. AbouElhamayed, “Deep learning for diabetic retinopathy: a comprehensive review,” *Diagnostics*, vol. 13, no. 1, pp. 1–20, 2023.

[11] K. Singhal, T. Tu, et al., “Large language models encode clinical knowledge,” *Nature*, vol. 620, pp. 172–180, 2023.

Appendices

Appendix A – Sample Input Reports

Contains examples of raw medical reports (in PDF or image format) used for system testing and OCR validation.

- Complete Blood Count (CBC) Report
- Thyroid Function Test (TFT) Report
- Lipid Profile Report

Appendix B – AI Model Output Samples

Displays examples of AI-generated outputs showing the transformation from raw text to summarized explanation.
Example:

- Extracted Text: “Hemoglobin: 9.8 g/dL (Normal: 12–15)”
- AI Summary: “Low hemoglobin detected. Possible anemia or iron deficiency.”
- Translated Output: “हीमोग्लोबिन कमी आहे. लोहाची कमतरता असू शकते.”

Appendix C – System Interface Screenshots

Screenshots of key interfaces from the Lab2Life Web Application:

1. Report Upload Page

2. OCR Extraction Display
3. AI Summary Output
4. Translation Selection (Marathi / Hindi / English)
5. Doctor Verification Dashboard
6. Final Verified Report View

Appendix D – Dataset and Tools Used

Datasets:

- MIMIC-IV Clinical Notes Dataset (PhysioNet)
- Manually collected sample lab reports

Tools & Frameworks:

- Python (FastAPI, Transformers, Tesseract, PyTorch)
- React.js (Frontend)
- MongoDB (Database)
- Google Translate API / MarianMT (Translation)

Future Scope

The Lab2Life project has significant potential for further enhancement and real-world application in healthcare technology.
Although the current system efficiently extracts, summarizes, and explains medical reports, several advancements can be introduced to expand its utility and impact.

1. Integration with Hospital Systems (EHR/EMR):

Lab2Life can be integrated with Electronic Health Record (EHR) or Hospital Management Systems to automatically process and summarize lab reports in real time for both doctors and patients.

2. Mobile Application Development:

Creating an Android and iOS app version of Lab2Life would make it more accessible, allowing users to scan reports directly from their phones and receive instant summaries.

3. Voice-Based Interaction:

Incorporating speech recognition and voice assistant features would help visually impaired or elderly users interact with the system using simple voice commands.

4. AI Chatbot Integration:

An AI-powered health chatbot can be added to provide conversational explanations, answer user queries about results, and offer personalized health suggestions.

5. Predictive Health Analytics:

Future versions can include machine learning models that analyze report trends over time and predict possible health risks or deficiencies before symptoms appear.

6. Multi-Domain Expansion:

The same AI framework can be extended to interpret radiology, pathology, and genomic reports, expanding Lab2Life beyond basic laboratory diagnostics.