Project Design Phase – II

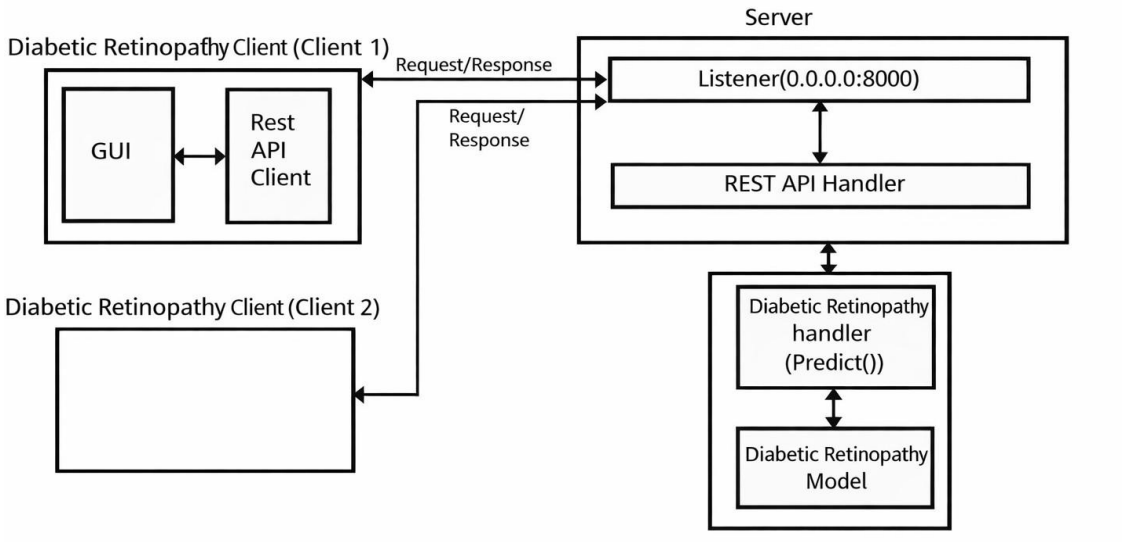
Data Flow Diagram & User Stories

Date:	02 Feb 2026
Team ID:	LTVIP2026TMIDS75194
Project Name:	Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy
Maximum Marks:	5 Marks

Data Flow Diagrams

A **Data Flow Diagram (DFD)** is a graphical representation of how data flows within a system. It illustrates how data enters the system, how it is processed, stored, and finally transformed into meaningful output. In this project, the DFD clearly represents the flow of retinal fundus images from the user interface through preprocessing, deep learning prediction, and result generation.

The DFD helps in understanding the internal working of the system and ensures that all functional requirements are properly addressed.



Data Pipeline and Processing

The data pipeline and processing components are responsible for transforming **raw retinal fundus images** into **model-ready inputs** for diabetic retinopathy detection. The pipeline ensures consistency, accuracy, and reliability of the data used for training and inference.

Key Responsibilities of the Data Pipeline

- Image ingestion from user uploads or datasets
- Image preprocessing and normalization
- Feature extraction through deep learning layers
- Validation of processed images
- Preparation of inputs for model training and prediction

Overview and Data Flow

The data pipeline transforms raw retinal fundus images through multiple stages to create standardized inputs suitable for deep learning algorithms. The pipeline processes medical image data by performing cleaning, resizing, normalization, and augmentation to enhance model performance.

Data Flow Description

1. Retinal fundus images are uploaded by the user
2. Images are validated and preprocessed
3. Cleaned images are passed to the deep learning model
4. The model analyzes retinal patterns
5. Prediction results are generated and returned to the user

This structured flow ensures accurate and efficient diabetic retinopathy detection.

Integration with Deep Learning Pipeline

The deep learning pipeline is tightly integrated with the data processing workflow to ensure consistent prediction behavior for both training and inference.

Pipeline Stages

- **Image Loading:** Retinal images are loaded into the system
- **Initial Exploration:** Image dimensions and quality are checked
- **Image Cleaning:** Noise reduction and contrast enhancement

- **Preprocessing:** Resizing and normalization
- **Feature Extraction:** Automatic feature learning using CNN layers
- **Model Inference:** Prediction using trained deep learning model

The output of this pipeline is a classification result indicating the presence or absence of diabetic retinopathy.

Backend Application Architecture

The backend architecture is designed to support seamless communication between the frontend interface and the deep learning model.

Backend Data Flow

- Preprocessed image data is passed to the prediction module
- The trained CNN model processes the image
- Prediction results are generated
- Results are returned to the frontend through REST APIs

The backend ensures fast inference, scalability, and secure handling of medical image data.

Flask Application and Model Integration

The Flask backend application acts as the core server that manages model loading, prediction requests, and response handling.

Key Components

- **Model Integration:**
    - Trained model loaded using serialization techniques
- **Flask Application Core:**
    - REST API configuration
    - Cross-Origin Resource Sharing (CORS) support
- **API Endpoints:**

- `/` – Landing page
- `/api/predict` – Diabetic retinopathy prediction
- `/api/contact` – User contact and feedback

The backend efficiently routes prediction requests and ensures reliable responses.

User Flow

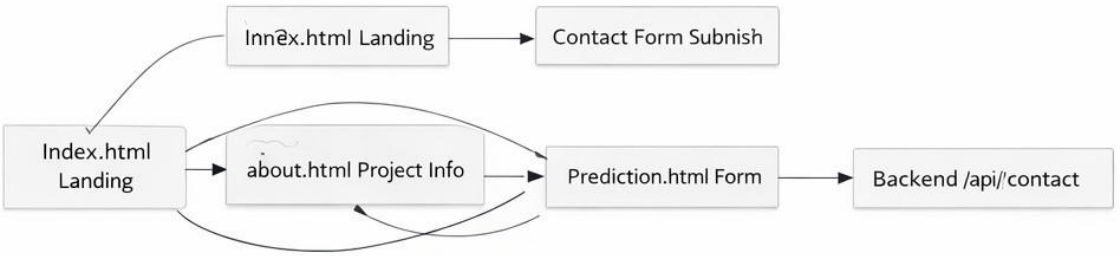
The user flow represents the step-by-step interaction between the user and the system.

User Interaction Flow

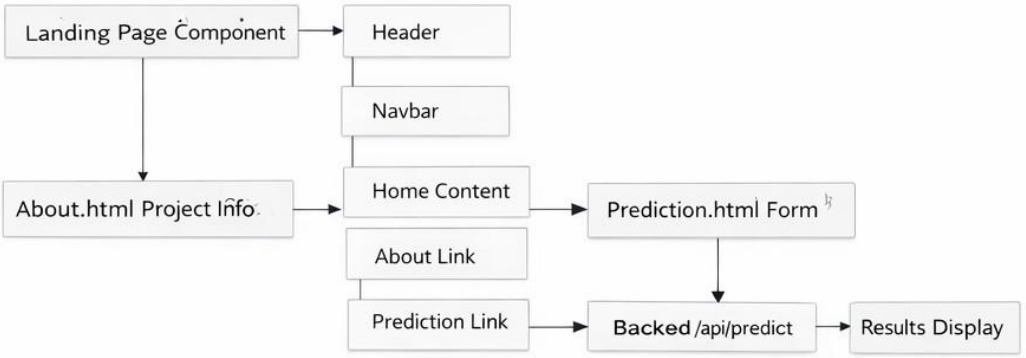
1. User accesses the landing page
2. Navigates to the prediction page
3. Uploads retinal fundus image
4. Submits prediction request
5. Backend processes image
6. Deep learning model generates prediction
7. Result is displayed to the user

This flow ensures a simple and intuitive diagnostic process.

3.2.1 Entry Point – Landing Page



3.2.2 About Page – Project Information



User Stories

User stories describe the functional requirements from the developer's perspective and define acceptance criteria for each sprint.

User Stories Table

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
Developer	Data Collection & Preprocessing	USN-1	As a developer, I want to collect and preprocess retinal fundus images so that clean and valid images are available for analysis	Images are cleaned, normalized, and ready for processing	High	Sprint-1

Developer	Feature Engineering	USN-2	As a developer, I want to perform image preprocessing and feature preparation so that the model receives standardized inputs	Images are resized, normalized, and augmented	High	Sprint-2
Developer	Model Development	USN-3	As a developer, I want to train and evaluate a CNN model so that diabetic retinopathy can be detected accurately	Model achieves high accuracy and F1-score	High	Sprint-3
Developer	Model Deployment	USN-4	As a developer, I want to integrate the trained model with a Flask API and frontend UI	API returns correct predictions and UI is responsive	High	Sprint-4
Developer	Testing & Final Deployment	USN-5	As a developer, I want to test and deploy the system on the cloud and document the workflow	System runs end-to-end with complete documentation	High	Sprint-5

Summary of Project Design Phase – II

- Clear data flow is defined using DFD concepts
- Robust image processing and deep learning pipeline implemented
- Backend architecture supports scalable deployment
- User stories ensure requirement traceability

- System design aligns with Agile development methodology