Project Design Phase
Solution Architecture

Date	29 June 2025
Team ID	LTVIP2025TMID46471
Project Name	Hematovision : advanced blood cell classification using transfer learning
Maximum Marks	4 Marks

Solution Architecture

The solution for HematoVision is designed to automate the process of classifying blood cells using deep learning techniques. This document provides an overview of the system's architecture, the key components involved, and the data flow within the system.

Key Components

1. **User Interface (Frontend):**
- **HTML + Bootstrap: Provides a responsive and user-friendly interface for image upload and result display.**
- **Key Features: Image upload form, result display, and option to upload multiple images sequentially.**

2. **Backend (Flask Application):**
- **Flask (Python): Handles HTTP requests, processes image data, and communicates with the trained model for prediction.**
- **Key Features: Image upload handling, communication with the MobileNetV2 model, and result return.**

3. **Model (MobileNetV2 with Transfer Learning):**
- **TensorFlow/Keras: MobileNetV2 is pre-trained and fine-tuned for blood cell classification.**
- **Model File: **Blood_Cell.h5** is used for inference.**
- **Key Features: Image preprocessing, model inference, and classification results for four blood cell types (Eosinophil, Basophil, Monocyte, Lymphocyte).**

4. **Data Flow:**
- **User Uploads Image: The user uploads a blood cell image via the web interface.**
- **Flask Backend Processing: The backend processes the image and communicates with the model for classification.**
- **Model Prediction: MobileNetV2 predicts the blood cell type and returns the result.**
- **Result Display: The Flask backend sends the result back to the user interface for display.**

5. **Model Storage:**
- **Data Store: The trained model is stored as **Blood_Cell.h5** and accessed during inference.**

Data Flow

Entities:
1. **User: Uploads the blood cell image.**
2. **Web Interface: Displays the image upload page and result.**
3. **Flask Backend: Handles image processing, invokes the model, and returns results.**
4. **MobileNetV2 Model: Classifies the blood cell image.**
5. **Model Storage: Stores the trained model.**

Data Flow:
1. **User Uploads Image → Web Interface → Flask Backend.**
2. **Flask Backend Sends Image → MobileNetV2.**
3. **MobileNetV2 Returns Prediction → Flask Backend.**
4. **Flask Backend Sends Result → Web Interface → User.**