**Project Documentation**

**🔹 Full Stack Development with Flask & TensorFlow**

 **Project Title:** **:** Hemato Vision : Advanced Blood Cell Classification Using Transfer Learning

# Introduction

**Project Title:** Hemato Vision : Advanced Blood Cell Classification Using Transfer Learning

**Team ID :** LTVIP2025TMID59611

**Team Size :** 4

**Team Leader :** Shaik Salma

**Team member :** Pulivarthi Srujana Tejaswi

**Team member :** Pinamaraju Srinivas

**Team member :** Ponugoti Harshitha

# Project Overview

* **Purpose:**

Hemato Vision is a Flask‑based web application that leverages transfer learning with pre‑trained CNNs (e.g., MobileNetV2, ResNet50) to classify four major white blood cell types—neutrophils, lymphocytes, monocytes, and eosinophils—from microscopic images

* **Features:**
  + Fast, high‑accuracy blood cell classification
  + User-friendly web interface

# Architecture

* **Frontend :** Flask with Bootstrap

based templates, offering image upload forms, confidence score display, and results

**Backend** **:**

User uploads image → Flask/FastAPI server → Preprocess image (OpenCV/TensorFlow) → Run it through pre-trained CNN → Return classification results

* **Model:**

The system uses a pre-trained ResNet50 as its backbone

# Setup Instructions

 **Prerequisites:**

Python 3.8+

Deep Learning Frame

CNN Models

**Installation Steps:**

bash

CopyEdit

# Clone the repository

git clone https://github.com/YOUR\_USERNAME/HematoVision.git cd HematoVision

# Install required packages pip install -r requirements.txt

# Run the application python app.py

# Folder Structure

cpp

CopyEdit

HematoVision/

├── app.py

├── requirements.txt

├── trained\_model.h5

├── static/

│ └── uploaded\_images/

├── templates/

│ ├── index.html

│ └── result.html

├── dataset/

├── model/

├── plots/

└── README.md

# Running the Application

* **Frontend:** Flask serves the HTML
* **Backend:** Run this 0in the terminal:

Bash Copy Edit python app.py

* **Access:**

Visit http://127.0.0.1:5000/ in the browser

# API Documentation

* **Endpoint:** /(root URL)
* **Method:** POST
* **Input:** Image (JPG/PNG)
* **Output:** Rendered HTML with predicted Cell Type (e.g., “Eosinophil, Lymphocyte, Monocyte, or Neutrophil”)

# Authentication

 Token-Based Auth (JWT)

# User Interface

* Clean Web-Based Dashboard
* Result Presentation
* Model Control & Exploration

# Testing

* Model Evaluation & Validation
* Integration & API Testing

# Screenshots or Demo

upload a blood cell image, and the Flask‑powered backend—using

Prediction Result Page

# Known Issues

* Limited, imbalanced datasets
* Lack of external validation

# Future Enhancements

* Explainable AI Integration
* Advanced Image Preprocessing & Architecture Optimization
* Self‑Supervised & Incremental Training