Artificial Intelligence and Machine Learning

Project Documentation

1. Introduction

- **Project Title:** HematoVision:Advanced blood cell Classification Using Transfer Learning
- **Team Members:** 1.G.Geethanjali(Team leader)
  2.K.Hemalatha
  3.M.Chandrika
  4.Y.DilliSravani

2. Project Overview

- **Purpose:** HematoVision is a deep learning-based web application designed to automatically classify different types of white blood cells—specifically eosinophils, lymphocytes, monocytes, and neutrophils—from microscopic images. The main goal is to assist medical professionals, researchers, and students in rapid, accurate identification of blood cell types, potentially aiding in disease diagnosis and hematological research.
- **Features:**
  - Deep Learning Integration
  - Organized Project Structure
  - Image Upload & Prediction
  - Modern UI Design

3. Architecture

```
HematoVision/
├── app.py
├── model_training/
│   └── Blood_Cell_Classification.ipynb
│   └── Blood_Cell.h5
├── static/
│   └── style.css
│   └── (user-uploaded images will go here)
└── templates/
    ├── home.html
    └── result.html
```

4. SetUp Instructions

- **Prerequisites:**
  1. Python 3.7 or above

2. Required Libraries: Flask, TensorFlow/PyTorch, OpenCV, NumPy
3. Project folders: static/, templates/, model/
4. Install dependencies: pip install -r requirements.txt

5. Running The Application

After setting up the dependencies and environment variables as described earlier, follow the steps below to run the HematoVision application locally:

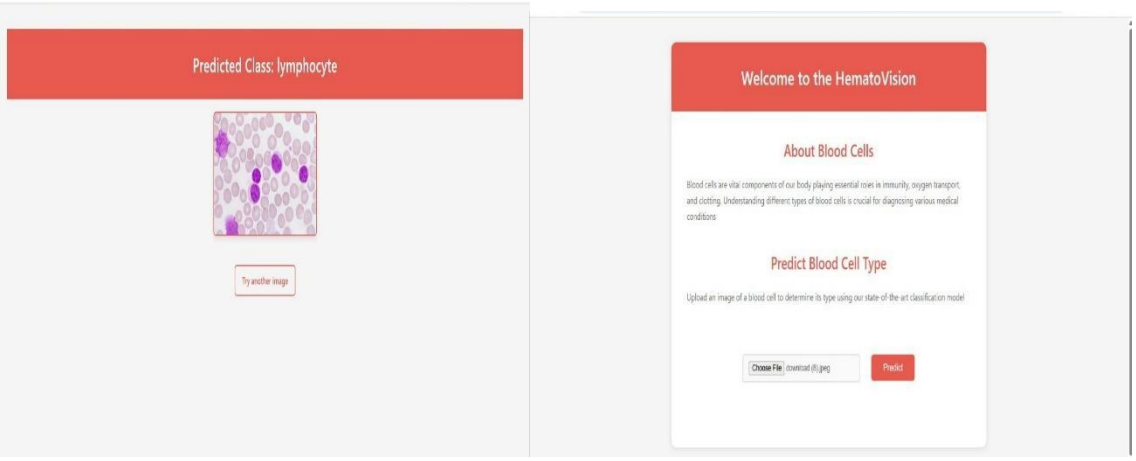
🖼️ Frontend :

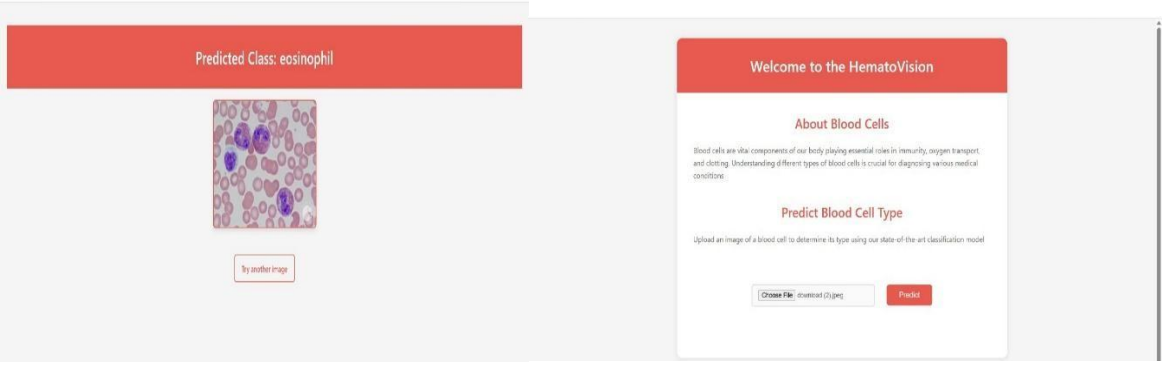
1. Hypertext Markup Languauge
2. Client Slide Scripting

🪚 Backend (app.py server):

1. Python
2. Flask
3. TensorFlow
4. OpenCV

6. User Interface





7. Testing

Purpose of Testing:

To ensure that all parts of the HematoVision system—ML model, backend APIs, and frontend UI—function correctly, reliably, and efficiently before deployment.

Testing Strategy Overview:

- ML Model Testing
- Backend API Testing
- Frontend (React) Testing
- End-to-End Testing

8. Known Issues:

While HematoVision is functional and reliable for most standard use cases, there are a few known limitations and issues that developers and users should be aware of.

- Model Accuracy on Blurry or Noisy Images
- No Input Validation for File Types (Backend)
- Large File Uploads
- No Authentication Implemented
- No Rate Limiting or Abuse Protection

9. Future Enhancements

- Machine Learning & Model Enhancements
- Frontend Features (React)
- Authentication & User Management
- Backend & Infrastructure
- Database Improvements
- Mobile & API Integration

10. Conclusion

- HematoVision simplifies cell classification for healthcare use
- It combines deep learning and web technologies for real-time inference
- Future improvements can include multi-class support and training interface