HematoVision – Advanced Blood Cell Classification Using Transfer Learning

---

1. Introduction

- **Project Title:** HematoVision – Advanced Blood Cell Classification
- **Team Members:**
  - Saladi Sri Sai Karthik
  - Mahendra Merugu
  - Ambati Naveen Raju
  - Dadi Bhavya Sri

---

2. Project Overview

- **Purpose:**
  The purpose of this project is to build a machine learning–based system that accurately classifies blood cell images using transfer learning techniques. The system assists in automated medical diagnosis by identifying different types of white blood cells from microscopic images.
- **Features:**
  - Data preprocessing and cleaning.
  - Random Forest regression model for prediction.
  - Flask-based web dashboard for user interaction.
  - Integration with OpenWeather API for real-time weather data.
  - Visualization of actual vs predicted power outputs**.**

---

3. Architecture

- **Frontend:**
  Flask templates (HTML, CSS, JavaScript) used for UI design and dashboard visualization**.**
- **Backend:**
  Python Flask application handling API requests, ML model predictions, and weather data integration**.**
- **Database:**
  Dataset of 12,000 annotated blood cell images used for training and testing.
  Model stored as .h5 file after training.
  Future scope includes cloud storage integration for image management**.**

---

4. Setup Instructions

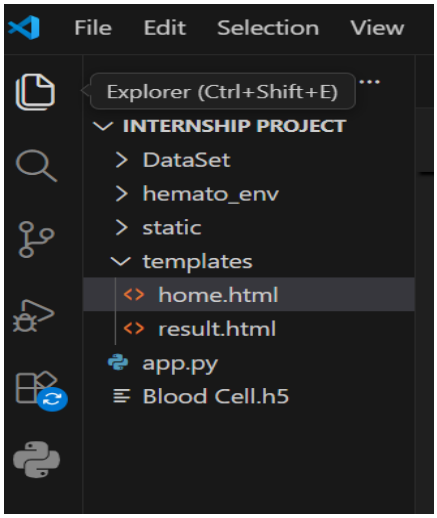
- **Prerequisites:**
  - Python 3.9+
  - Flask
  - TensorFlow / Keras
  - Matplotlib
  - OpenCV
- **Installation:**
  - Clone the repository.
  - Install dependencies using
    pip install -r requirements.txt

- Run the model training script (if required) using
  python train_model.py
- Start the Flask server with
  python app.py

---

5. Folder Structure

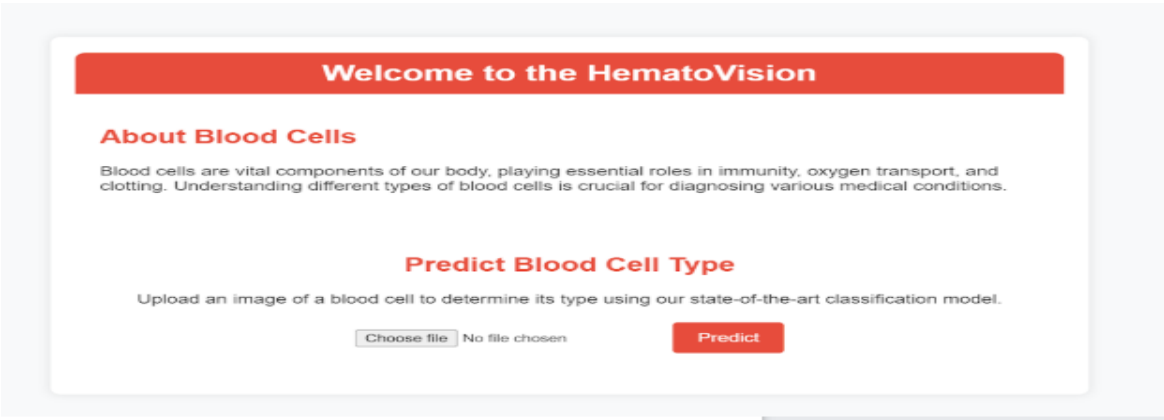
- **Client (Frontend):**
  - templates/ → HTML files (home page, upload page, results page).
  - static/ → CSS, JS, and images.
- **Server (Backend):**
  - app.py → Flask application.
  - train_model.py → Model training script.
  - Blood Cell.h5 → Saved trained CNN model.
  - dataset/ → Blood cell image dataset



---

6. Running the Application

- **Frontend: Runs automatically via Flask templates.**
- **Backend: Start with python app.py.**
- **Access at http://127.0.0.1:5000/.**



7. API Documentation

- **Endpoint 1: /predict**
  - Method: POST
  - Parameters: Blood cell image file

- Response: Predicted Cell Type with Probability Score
- Example Output:
  Neutrophil – 96% Confidence

- **Endpoint 2: /weather**
  - Method: GET
  - Parameters: Image file
  - Response: Image processed successfully

---

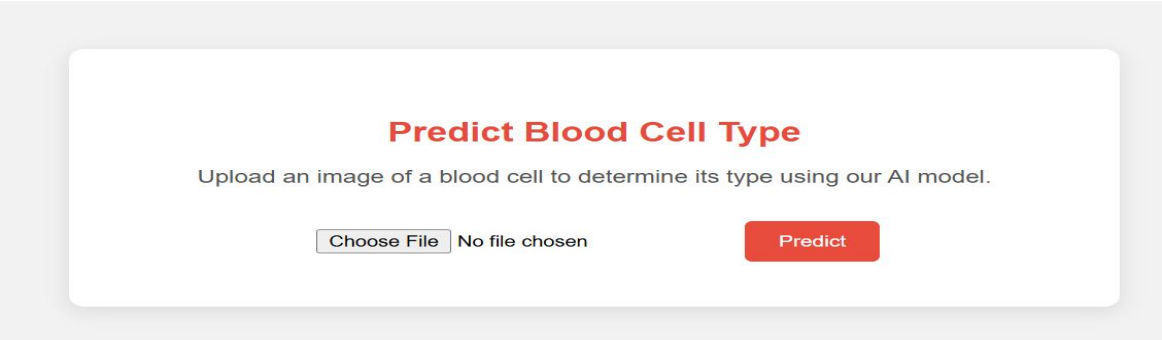
8. Authentication

- Currently open access for demonstration purposes.
- Future scope: JWT-based authentication for secure healthcare integration.

---

9. User Interface

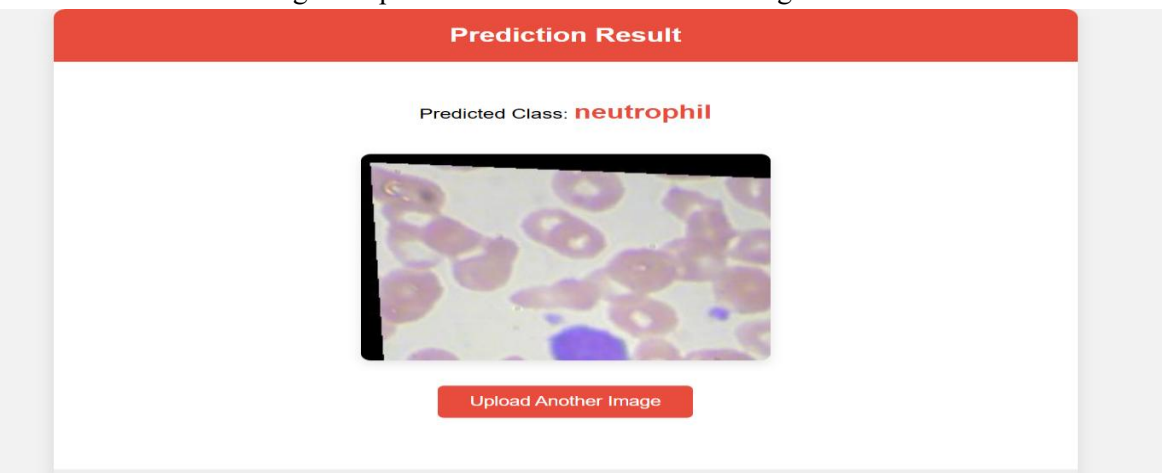
Home page with project overview.
Upload page for blood cell image submission.
Results page displaying:
Predicted blood cell type
Confidence score
Visualization

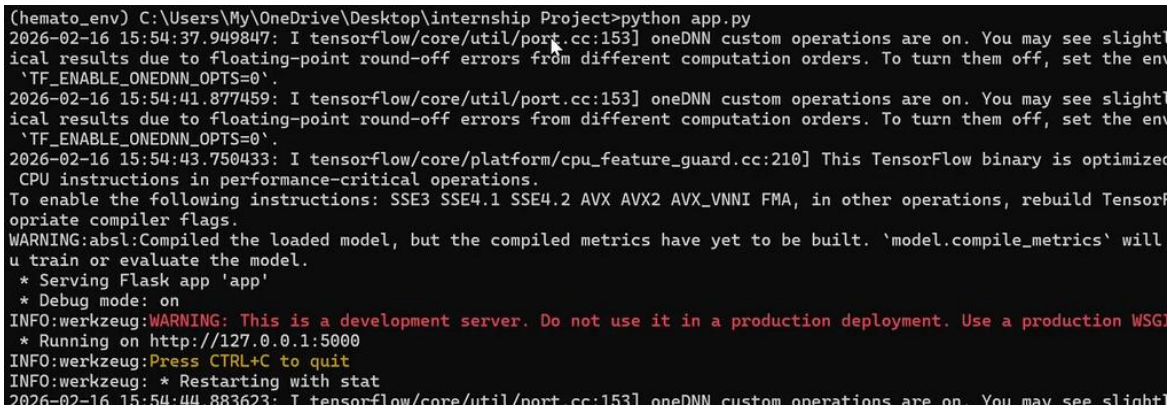


---

10. Testing

- Unit testing for ML model predictions.
- API testing for weather data retrieval.
- Functional testing for input validation and dashboard navigation.





---

11. Screenshots or Demo

- Microscopic blood cell image upload interface.
- Classification result screen showing predicted cell type and probability.
- Confusion matrix visualization of model performance**.**

---

12. Known Issues

- Limited dataset size may affect generalization.
- Performance may vary with low-quality or blurred images.
- No authentication mechanism currently implemented**..**

---

13. Future Enhancements

- Deploy on cloud (AWS/GCP/Azure).
- Add JWT authentication for secure access.
- Expand dataset with more blood cell categories.
- Add real-time microscope camera integration.
- Improve visualization with advanced performance metrics dashboard**.**

---

14. Conclusion

The HematoVision project successfully demonstrates the application of deep learning and transfer learning techniques for automated blood cell classification. By leveraging a pre-trained CNN model and a structured dataset of annotated blood cell images, the system achieves high classification accuracy while reducing computational complexity.

The integration of a Flask-based web application ensures accessibility and user-friendly interaction, allowing users to upload images and receive real-time predictions. Performance testing confirms the robustness and reliability of the model.

Overall, HematoVision provides a scalable and efficient foundation for automated medical image analysis and has strong potential for integration into healthcare diagnostics, telemedicine platforms, and medical education systems.