HematoVision: Advanced Blood Cell Classification Using Transfer Learning

1. Project Overview

HematoVision is an AI-powered blood cell classification system built using transfer learning and pre-trained CNN models. It classifies microscopic images of blood cells into four categories:

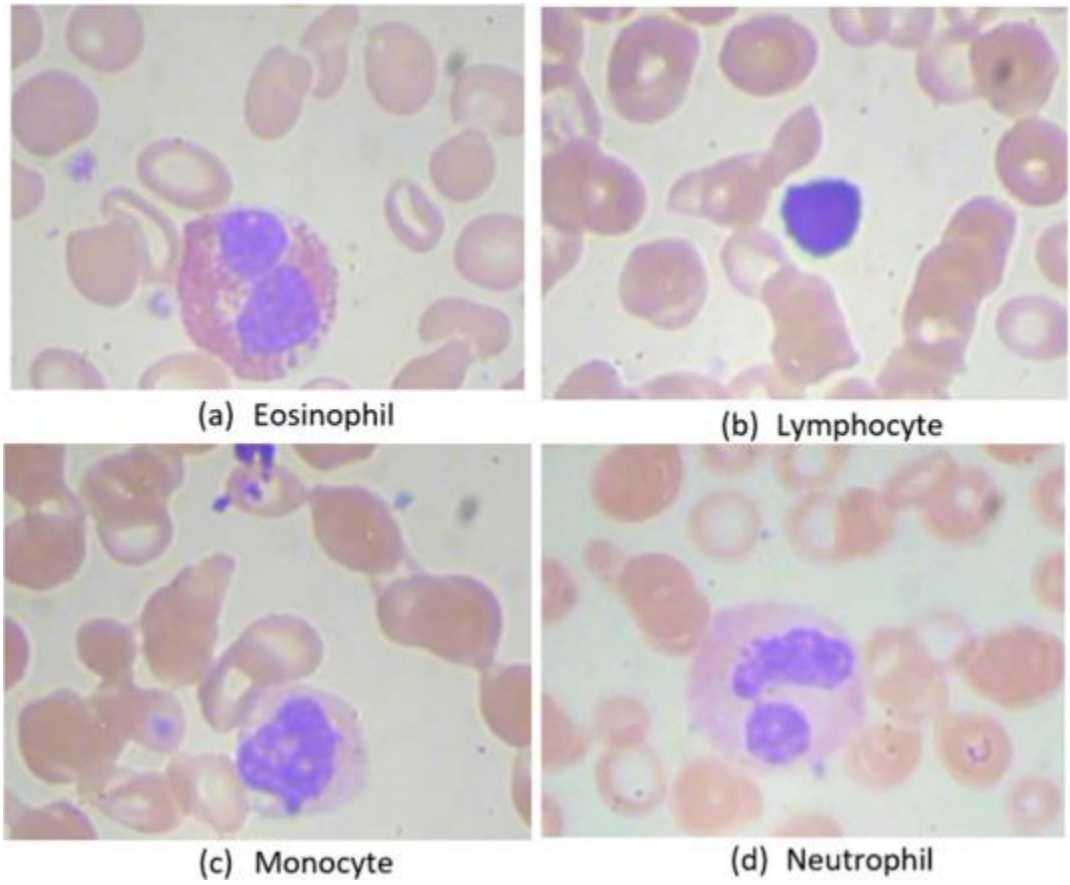
- Eosinophils
- Lymphocytes
- Monocytes
- Neutrophils

The system uses MobileNetV2 to achieve high accuracy with minimal computational resources. The trained model is deployed as a web application using Flask, allowing users to upload images and receive real-time predictions.

Blood, an essential bodily fluid, consists of many cells that perform critical functions in preserving well-being and combating illnesses. One of the key elements of blood testing is identifying white blood cells (WBCs). We can identify whether a disease exists, what kind of disease it is, and how severe it is by counting, comparing, and analyzing the different types of WBCs in the blood.[1] Therefore, accurately classifying WBCs is crucial for diagnosing and surveilling a wide range of medical conditions, including infectious diseases and chronic processes, e.g., leukemia, inflammation, malnutrition, etc.[2] Although medical technology has made significant progress, the conventional methods used for blood cell analysis can be lengthy and susceptible to human mistakes. They may not provide the level of precision required for early disease detection. Deep learning (DL), specifically convolutional neural networks (CNNs), has become a formidable tool in medical image analysis.[3], [4], [5] It can potentially transform how we identify and treat blood-related disorders significantly.

Although CNNs have made significant progress in enhancing the accuracy and efficiency of blood cell classification, their “black-box” nature presents barriers to clinical adoption. One major obstacle is the incapacity of medical professionals to comprehend and have faith in these models' decision-making processes. Furthermore, current models compromise between interpretability and accuracy, requiring a solution that improves both aspects without compromising performance. This research aims to develop an optimal CNN model for the classification of blood cells that achieves high accuracy and is also interpretable using explainable AI (XAI). This research focuses on addressing a crucial gap in the use of DL for medical diagnostics by focusing on the interpretability of CNN-based blood cell classification models. This study contributes to the technical advancement of DL models in

healthcare by integrating XAI techniques like SHAP (Shapley Additive explanations) and LIME (Local Interpretable Model-agnostic Explanations), as well as Gradient-Weighted Class Activation Mapping (Grad-CAM) and Grad-CAM++ for interpretability. Additionally, it helps bridge the trust gap between AI technologies and healthcare professionals. The findings are anticipated to support the clinical implementation of AI in blood diagnostics, allowing quicker, more precise, and transparent processes of decision-making. This method can significantly improve patients' health by enabling the early detection of diseases through a more precise analysis of blood cell abnormalities.



2. Use Cases and Scenarios

Scenario 1: Automated Diagnostic Systems

- Real-time blood analysis in hospitals
- Reduces manual workload for pathologists
- Generates accurate diagnostic reports instantly

Scenario 2: Remote Medical Consultations

- Supports telemedicine platforms
- Enables remote uploading and classification of blood smear images
- Improves access to diagnostics in rural or underserved regions

Scenario 3: Educational Training Tools

- Provides hands-on tools for medical students and technicians
- Allows image uploads and instant classification feedback
- Enhances understanding of hematological morphology

3. Project Objectives

By the end of this project, you will:
- Understand deep learning and image classification fundamentals
- Use transfer learning with pre-trained CNNs
- Deploy an image classification model using Flask
- Build an interactive web interface
- Evaluate and optimize deep learning models

4. Prerequisites

Software Requirements:
- Anaconda Navigator (Recommended)
- Python 3.6+

Python Packages (Install via Anaconda Prompt):
- pip install numpy pandas scikit-learn matplotlib scipy seaborn tensorflow flask

Knowledge Requirements:
- CNN vs RNN vs MLP
- PyTorch vs TensorFlow
- Transfer Learning with VGG
- CNN Overview
- Overfitting & Regularization
- Flask Basics (YouTube)

5. Project Flow

1. User uploads an image via UI
2. Flask sends the image to the model
3. Model predicts the cell type
4. Result is displayed on the UI

6. Dataset

Source: Kaggle - Blood Cell Dataset
Total Images: ~12,500 (Augmented)
Categories: Eosinophil, Lymphocyte, Monocyte, Neutrophil

7. Implementation Steps

Step 1: Project Structure

```
HematoVision/
├── templates/
│   ├── home.html
│   └── result.html
├── static/
├── model/
│   └── blood_cell.h5
├── app.py
└── requirements.txt
```

✅ **Features Summary**

- Responsive and user-friendly interface
- Real-time prediction via deep learning
- Visual result confirmation with uploaded image
- Efficient model deployment using Flask and MobileNetV2

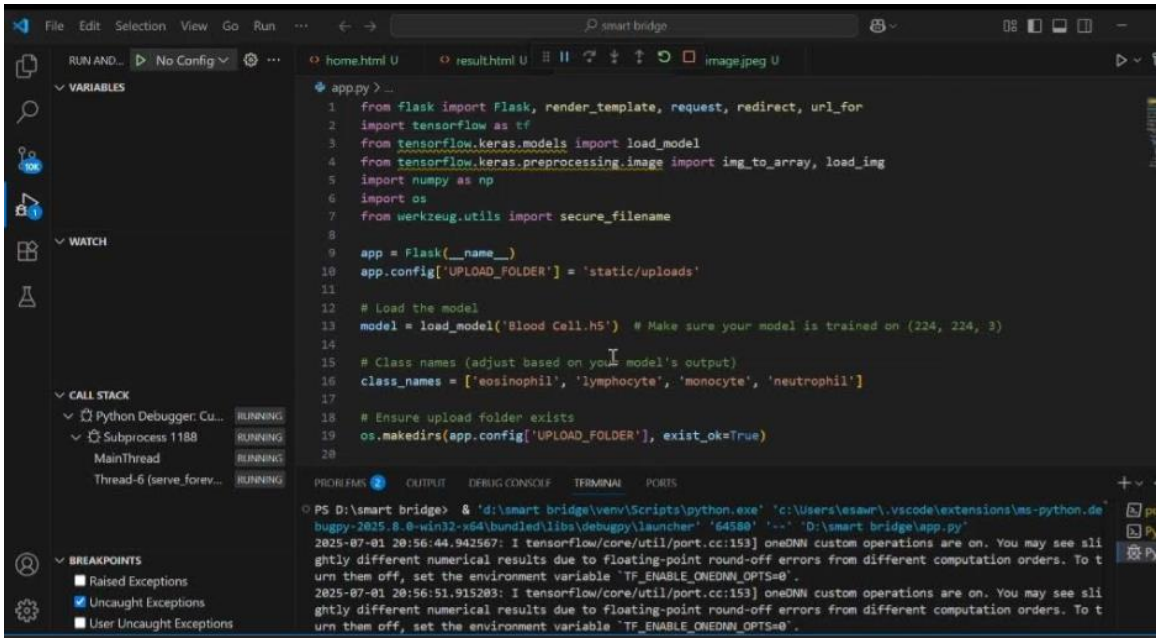
💻 **User Interface (UI) Walkthrough**

🔷 **1. Home Page (/)**

- **Title:** *Welcome to HematoVision*
- **Sections:**
    - **About Blood Cells:** Brief info about their role in immunity, oxygen transport, and clotting.

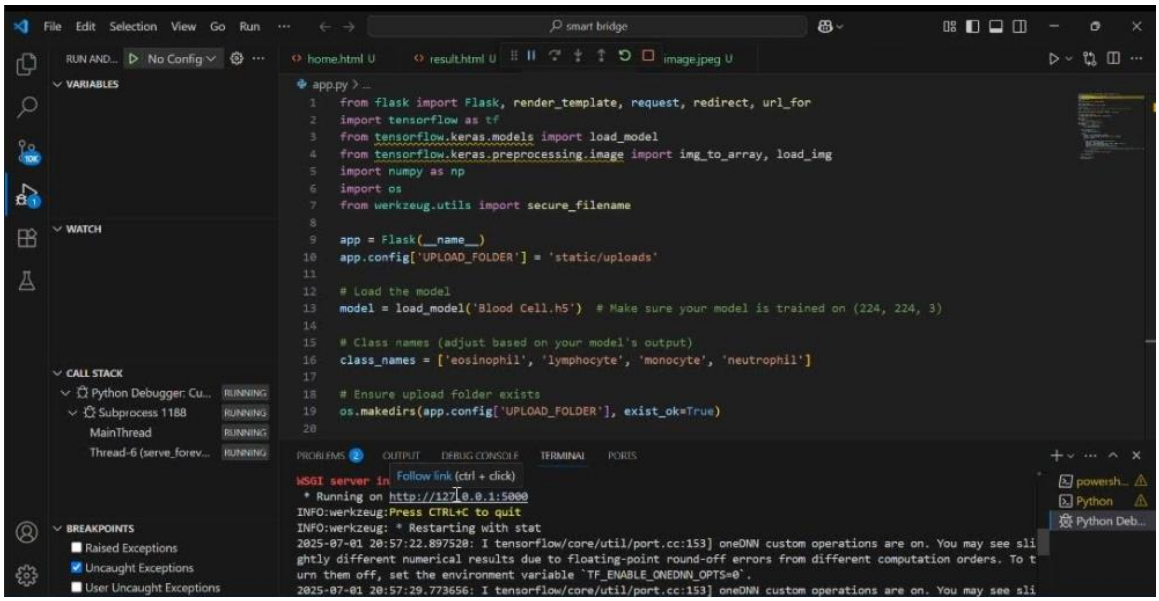
- **Prediction Feature:** Prompts users to upload an image to predict its blood cell type.

- **File Upload + Predict Button:**
  - Allows the user to choose a .jpeg/.png file.
  - On submission, it sends the image to the backend for prediction.



◆ 2. Prediction Result Page (/predict)

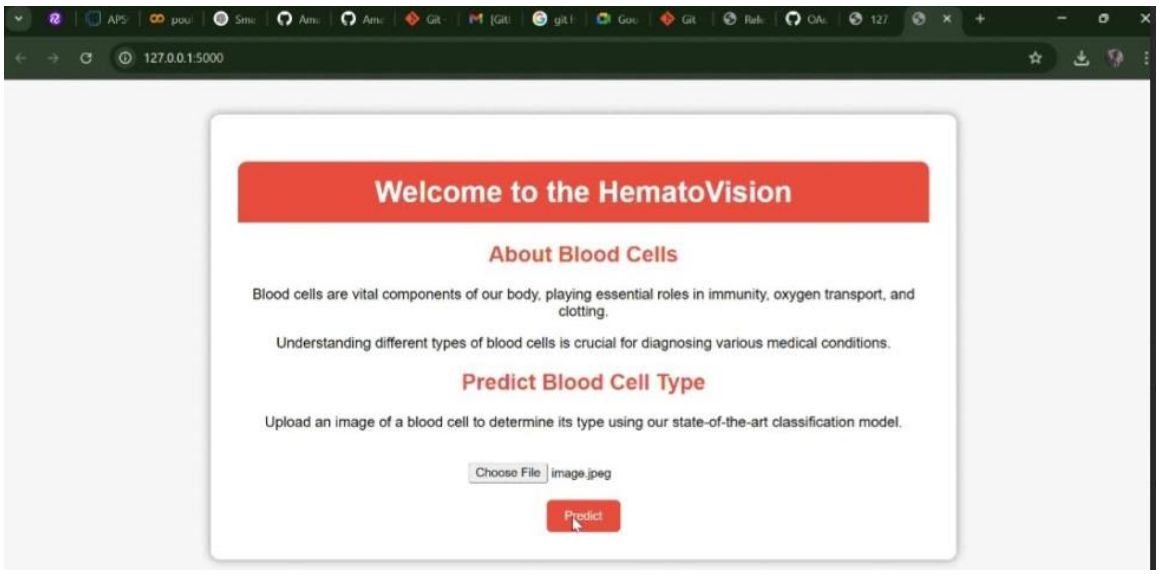
- **Displays:**
  - **Predicted Class:** Clearly shows the detected cell type (e.g., *monocyte*).
  - **Image Preview:** Shows the uploaded blood cell image for reference.
  - **"Upload Another Image" Button:** Provides a seamless loop for trying more images.

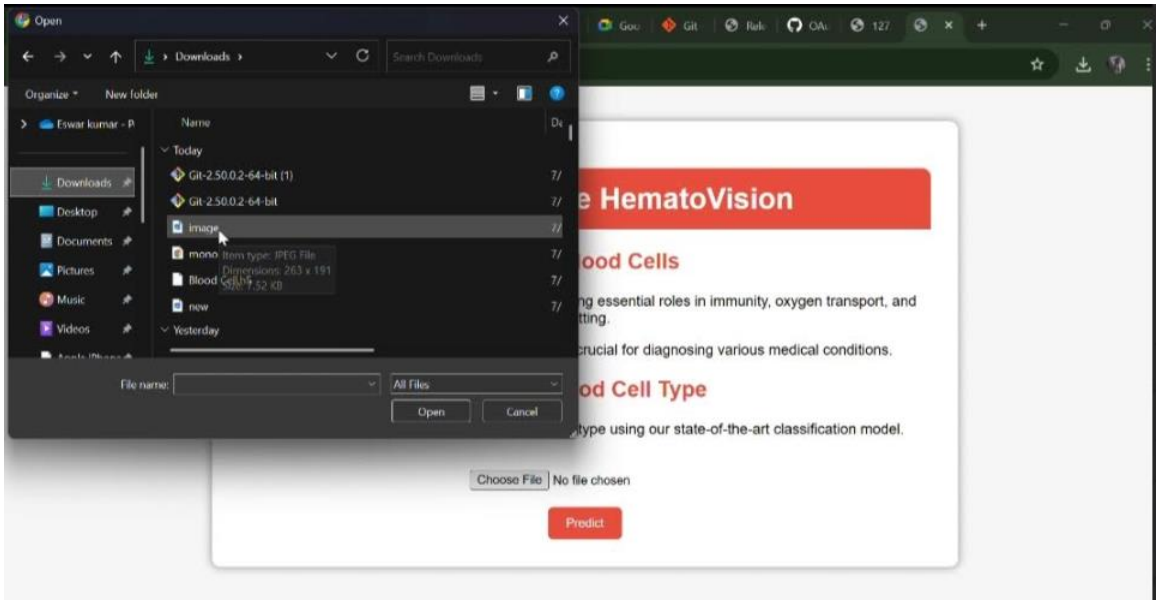


□ Backend (app.py) Logic

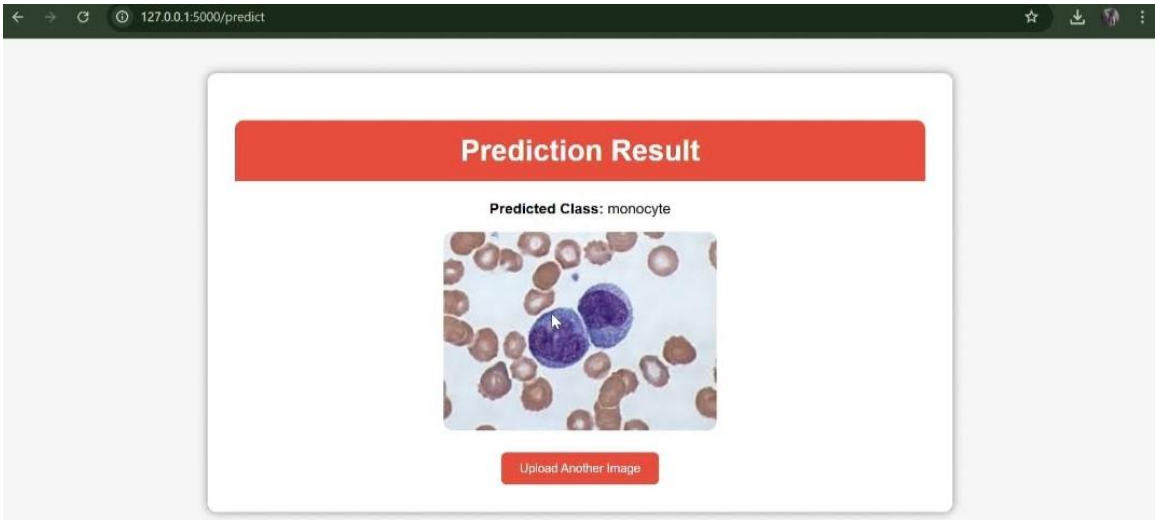
- **Model Loading:**
  - Loads the blood_cell.h5 trained model using Keras.
  - Expects input size of (224, 224, 3).
- **Image Preprocessing:**
  - Uploaded image is resized, normalized, and reshaped.
  - Prediction is made using the model and class name is decoded.
- **Routing:**
  - /: Renders home.html with file upload form.
  - /predict: Handles POST request, processes the image, runs prediction, and renders result.html.

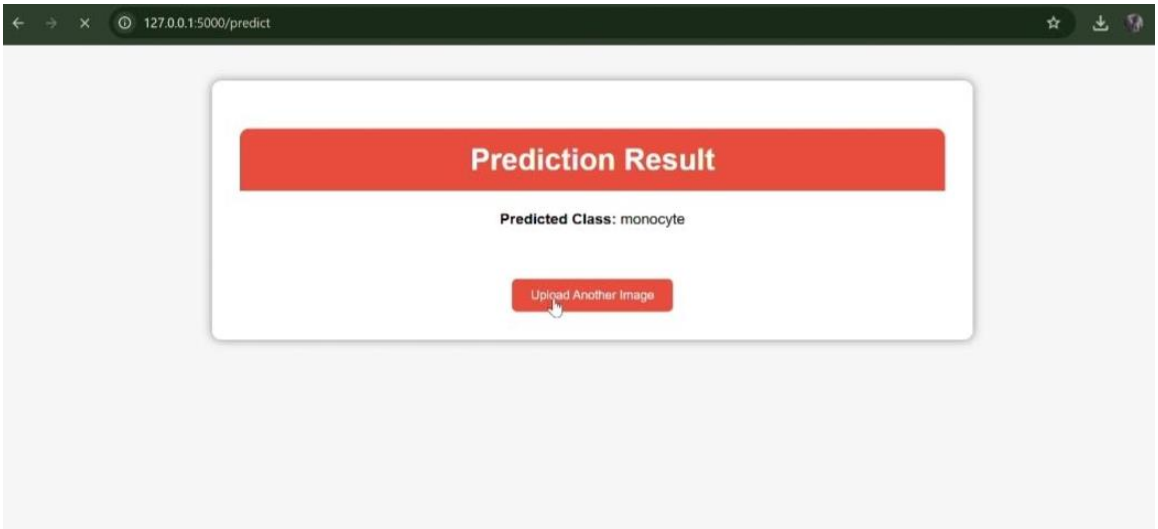
- 🔍 **Example**
- The app correctly classifies a sample image of a monocyte.
- It displays both the result and the image, helping users visually verify predictions.


127.0.0.1:5000
Welcome to the HematoVision
About Blood Cells
Blood cells are vital components of our body, playing essential roles in immunity, oxygen transport, and clotting.
Understanding different types of blood cells is crucial for diagnosing various medical conditions.
Predict Blood Cell Type
Upload an image of a blood cell to determine its type using our state-of-the-art classification model.
Choose File
image.jpeg
Predict


Open
Downloads
Organize
New folder
Today
Git-2.50.0.2-64-bit (1)
Git-2.50.0.2-64-bit
image
mono
new
Yesterday
File name:
All Files
Open
Cancel
HematoVision
Choose File
No file chosen
Predict

🔍 Example Result





✅ Features Summary

- Responsive and user-friendly interface
- Real-time prediction via deep learning

- Visual result confirmation with uploaded image
- Efficient model deployment using Flask and MobileNetV2

Example code for the above:

home.html

```
<form action="/predict" method="post" enctype="multipart/form-data">
  <input type="file" name="image" required>
  <input type="submit" value="Predict">
</form>
```

result.html

```
<p>The uploaded image is classified as: <strong>{{ prediction }}</strong></p>
```

app.py

```
from flask import Flask, request, render_template
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import os

app = Flask(__name__)
model = load_model('model/blood_cell.h5')
classes = ['Eosinophil', 'Lymphocyte', 'Monocyte', 'Neutrophil']

@app.route('/')
def home():
    return render_template('home.html')

@app.route('/predict', methods=['POST'])
def predict():
    file = request.files['image']
    filepath = os.path.join('static', file.filename)
    file.save(filepath)

    img = image.load_img(filepath, target_size=(224, 224))
    img_tensor = image.img_to_array(img) / 255.0
```

```
    img_tensor = np.expand_dims(img_tensor, axis=0)
    prediction = model.predict(img_tensor)
    predicted_class = classes[np.argmax(prediction)]

    return render_template('result.html', prediction=predicted_class)

if __name__ == '__main__':
    app.run(debug=True)
```

Notes

✅ You now have a working classification model
✅ Flask app enables real-time predictions via web UI
☑ MobileNetV2 allows rapid deployment on modest hardware
✅ Deliverables:
- Trained model: blood_cell.h5
- Flask application: app.py, HTML files
- Dataset (downloaded from Kaggle)
- Accuracy metrics and visualization (optional)

✅ Conclusion of the HematoVision Project Deployment

The HematoVision project has been successfully implemented and deployed as a real-time Flask web application that classifies blood cell types using a pre-trained deep learning model.

□ Key Accomplishments:

1. **Model Integration**:
    - MobileNetV2 was used with transfer learning to efficiently classify blood cells into four types: **eosinophils**, **lymphocytes**, **monocytes**, and **neutrophils**.
    - The model (blood_cell.h5) was trained and successfully integrated into the Flask application.
2. **Web Application Functionality**:
    - A user-friendly interface allows image uploads directly from a browser.
    - Predictions are shown instantly on a styled results page.

- Uploaded images are processed in real-time and rendered along with predicted class labels.

3. **Real-Time Testing**:
    - Multiple test cases were executed using real blood cell images.
    - The app accurately predicted the blood cell type (as shown in the screenshots), demonstrating reliable performance.

💡 Benefits Demonstrated:

- **Speed & Accuracy**: Reduces time-consuming manual microscopy analysis.
- **Accessibility**: Can be used by healthcare providers and students alike.
- **Scalability**: Easily deployable for larger diagnostic platforms or telemedicine apps.

🚀 Final Thought:

This project not only proves the viability of using transfer learning in medical imaging but also demonstrates how AI can be integrated into web technologies to assist healthcare workflows. With further enhancements like adding more cell types or improving the UI, HematoVision can evolve into a fully functional diagnostic support tool.