**An internship in**

**Artificial Intelligence & Machine Learning**

by

**SmartInternz**

**Project Name : HematoVision: Advanced Blood Cell Classification Using**

**Transfer Learning Estimation with Machine Learning**

**Project Id : LTVIP2025TMID59837**

**Project Mentor : M. Ganesh**

**Team Members and Roles**:

1. **Mummadi Yuvaraj**– Project Leader & Deep Learning
2. **Narendragari Vyshnavi**– Data Analyst & Preprocessing Specialist*.*
3. **Manchala Vinod Kumar**– Web Developer & UI Designer*.*
4. **Pachari Sai Anith Kumar** – System Integrator & Tester

**DEPARTMENT OF COMPUTER SCIENCE AND**

**ENGINEERING,Annamacharya Institute of Technology & Sciences, Utukur**

**(Post), Chinthakomma Dinne (V&M), Kadapa, YSR (Dist) Andhra Pradesh**

**ACKNOWLEDGEMENT**

I am pleased to acknowledge my sincere thanks to **Institute** of **ANNAMACHARYA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. A. Sudhakar Reddy,** principal of Annamacharya Institute of technology and sciences and **Dr.C.V.Subbaiah, M.tech., Ph.D.,** Heads of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project mentor **MR.M.Ganesh,** for his valuable guidance, suggestions and constant encouragement pavedway for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

# ABSTRACT

Blood cell analysis plays a vital role in diagnosing various hematological diseases such as anemia, leukemia, and infections. Traditionally performed manually by expert pathologists through microscopic examination, this process can be slow, error-prone, and labor-intensive. To address these limitations, HematoVision presents an AI-based approach that automates the classification of blood cells using transfer learning and deep learning techniques.This project leverages a pre-trained convolutional neural network (CNN), fine- tuned on a labeled dataset of blood smear images. The system is deployed through anoffline web application, allowing users to upload microscopic images and receive instant predictions on the blood cell type. HematoVision improves diagnostic speed, ensures consistent accuracy, and extends accessibility to remote healthcare settings where expert analysis may not be available.This work demonstrates the effectiveness of AI imedical diagnostics and show cases the potential of transfer learning

inreducing both computational resources and development time in biomedical applications

**Key Words**: Blood Cell Classification, Deep Learning, Transfer Learning, Convolutional Neural Network (CNN), Biomedical Imaging, Hematology, Automated Diagnosis.

**Project Report Format**

1. **INTRODUCTION**
   1. Project Overview
   2. Purpose
2. **IDEATION PHASE**
   1. Problem Statement
   2. Empathy Map Canvas
   3. Brainstorming
3. **REQUIREMENT ANALYSIS**
   1. Customer Journey map
   2. Solution Requirement
   3. Data Flow Diagram
   4. Technology Stack
4. **PROJECT DESIGN** 
   1. Problem Solution Fit
   2. Proposed Solution
   3. Solution Architecture
5. **PROJECT PLANNING & SCHEDULING** 
   1. Project Planning
6. **FUNCTIONAL AND PERFORMANCE TESTING** 
   1. Performance Testing
7. **RESULTS** 
   1. Output Screenshots
8. **ADVANTAGES & DISADVANTAGES**
9. **CONCLUSION**
10. **FUTURE SCOPE 11. APPENDIX**

Source Code(if any)

Dataset Link

GitHub & Project Demo Link

# 1. INTRODUCTION

**1.1 Project Overview**

In the field of medical diagnostics, timely and accurate identification of blood cell types is crucial for diagnosing a wide range of diseases and health conditions. Traditionally, this classification is performed manually by expert pathologists using microscopes, a process that is time-consuming, prone to human error, and difficult to scale. With the advancement of deep learning technologies, automated image classification has become a viable and efficient solution in the medical domain.

HematoVision is a deep learning-based project designed to automate the classification of blood cells using advanced transfer learning techniques. The project utilizes a publicly available dataset consisting of over 12,000 labeled images of four primary types of blood cells: Eosinophils, Lymphocytes, Monocytes, and Neutrophils. By leveraging a pre-trained MobileNetV2 model, HematoVision significantly reduces training time while achieving high classification accuracy.

The use of transfer learning allows the model to inherit knowledge from large- scale image recognition tasks and adapt it to the specialized task of medical image classification. This enhances model performance, especially when dealing with limited data availability. Furthermore, HematoVision integrates the trained model into a Flaskbased web application, enabling end-users—including doctors, lab technicians, and students—to upload blood cell images and instantly receive accurate classification results through an intuitive user interface.

This project not only demonstrates the power of transfer learning in biomedical applications but also provides a scalable, real-time diagnostic tool that can be used in clinical settings, remote medical consultations, and educational environments.



**Figure 1.1: Project Workflow of HematoVision**

**1.2 Purpose**

The primary purpose of the HematoVision project is to develop an automated, intelligent system capable of accurately classifying blood cell images into their respective categories using deep learning and transfer learning techniques. This system is intended to assist medical professionals, reduce diagnostic errors, and accelerate the process of blood analysis in clinical and educational environments.

Specifically, this project aims to:

* Automate the classification of blood cells into four major types — Eosinophils, Lymphocytes, Monocytes, and Neutrophils — using a convolutional neural network (CNN) model.
* Leverage transfer learning by utilizing a pre-trained MobileNetV2 architecture to reduce training time and improve model generalization on limited medical image data.
* Support pathologists and healthcare professionals with a reliable tool that reduces manual effort and improves diagnostic accuracy.

* Enable remote and accessible diagnostics by integrating the trained model into a Flask- based web application where users can upload images and receive real-time predictions.
* Provide an interactive educational platform where medical students and lab technicians can better understand blood cell morphology through real-time image analysis and feedback.

By achieving these objectives, HematoVision supports the broader goals of enhancing medical diagnostics through artificial intelligence and democratizing access to quality healthcare tools, especially in under-resourced or remote regions.

# 2. Ideation Phase

This phase focuses on understanding the underlying problem, empathizing with the stakeholders, and generating creative solutions to address the problem using deep learning and web technologies.

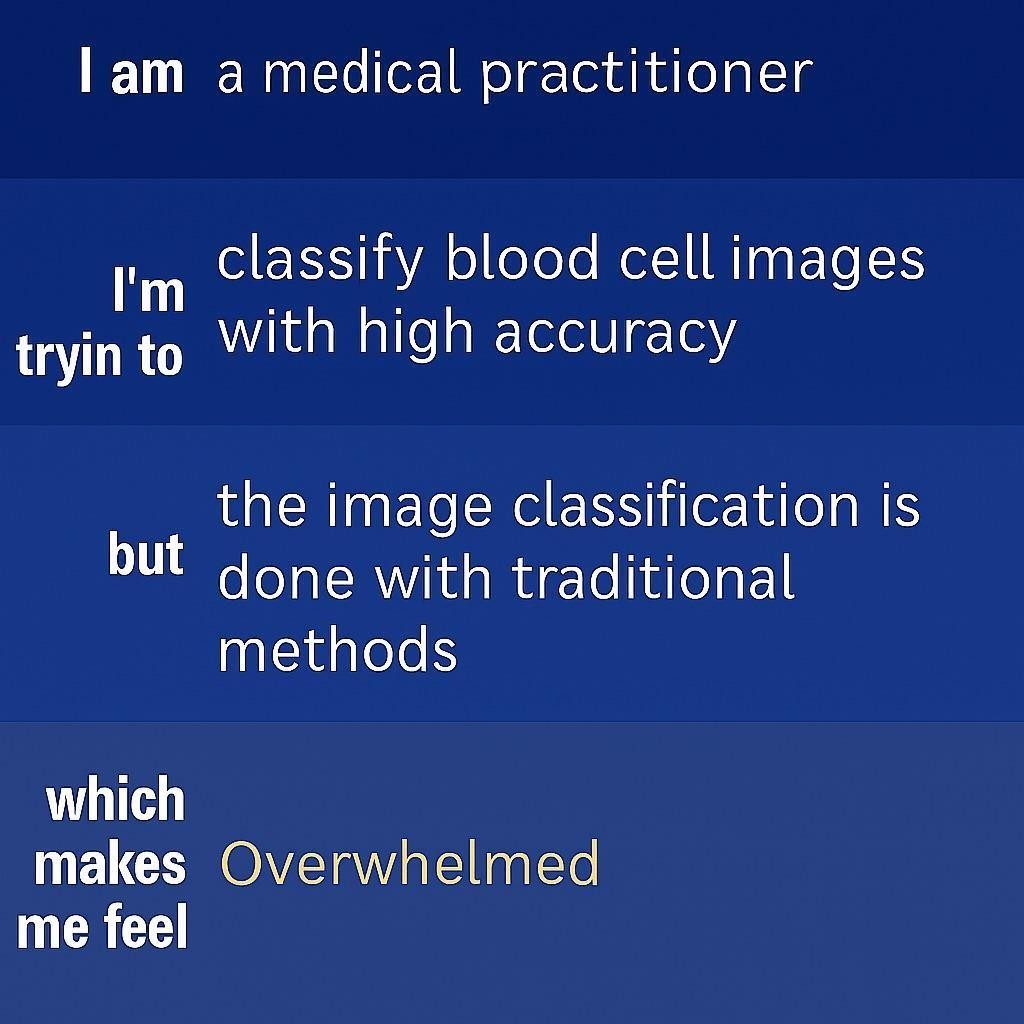
**2.1 Problem Statement:**

Manual microscopic analysis of blood smear slides is time-consuming, error-prone, and requires expert-level precision. Especially in rural or under-resourced healthcare centers, there’s a lack of trained pathologists to interpret such images.

Thus, there is a need for an intelligent, fast, and accurate image classification tool that can assist in identifying types of white blood cells for diagnosis of conditions like infections, leukemia, and immune disorders.

**Problem Statement:**

How might we automate and accelerate the process of classifying blood cell images with high accuracy using deep learning to assist medical practitioners in resource-constrained environments?



**2.2 Emapathy Map Canvas:**

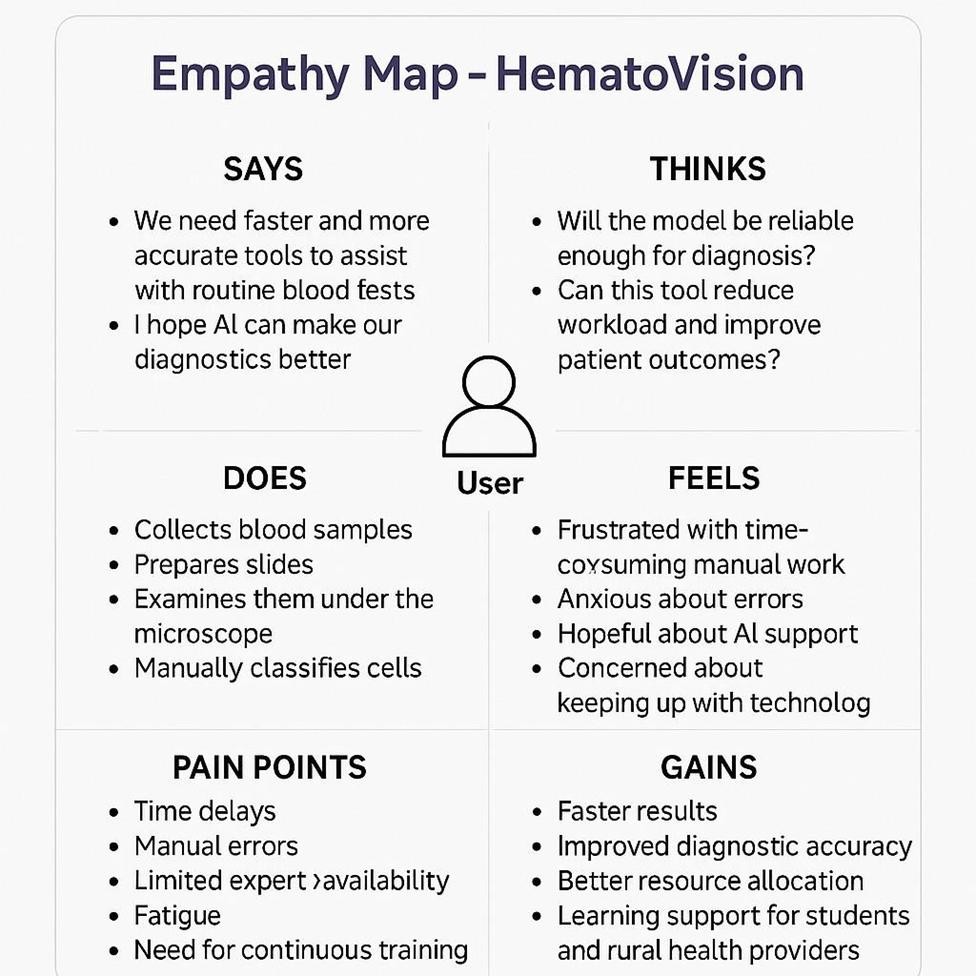
In the context of HematoVision, the user is typically a medical professional—such as a pathologist, hematologist, or lab technician— working in diagnostic laboratories, hospitals, or rural clinics. These individuals are skilled in identifying and analyzing blood cell types under a microscope, but often face time constraints, fatigue, and pressure for accuracy. They are focused on achieving faster and more accurate.

diagnostics for conditions such as leukemia, anemia, and infections. Their goal is to streamline the classification process to save lives and reduce diagnostic delays.

However, they face significant challenges due to manual analysis being slow, error-prone, and laborintensive, especially in settings with high patient volume or limited staff. In many cases, labs lack access to automated or AI-based tools, increasing dependency on subjective visual inspection.

These challenges arise because of infrastructural limitations, lack of automation, and the variability in visual characteristics of blood cells, which can make even experienced technicians susceptible to misclassification. Moreover, developing nations often lack access to advanced medical imaging systems, further slowing diagnosis.

As a result, this situation makes users feel frustrated, anxious, and overburdened. They may feel a lack of confidence in consistent decision- making, which can impact the quality of patient care. Integrating a solution like HematoVision could bring a sense of relief, support, and empowerment, allowing them to focus on treatment rather than being bogged down by laborious diagnostic procedures.



**Brainstorming**

**Step 1:** Team Gathering, Collaboration, and Selecting the Problem Statement The team gathered to collaboratively discuss healthcare challenges where artificial intelligence can offer impactful solutions. After reviewing several possibilities, the team selected the following problem statement: "How might we automate and accelerate the process of classifying blood cell images with high accuracy using deep learning to assist medical practitioners in resource-constrained environments?"

This was chosen because of its clinical significance, potential for automation, and relevance to realworld pathology problems.

**Step 2:** Brainstorming, Idea Listing and Grouping Raw Ideas:- Use pre-trained CNNs to reduce training cost (MobileNet, ResNet)- Enable image upload via simple web interface (Flask)- Autoresize and normalize uploaded images- Visual display of predictions alongside uploaded image- Offer downloadable PDF of results for records- Add voice instructions for lab assistants- Allow batch uploads of images- Add disease suggestion based on cell count ratio Grouped Themes:- Model Optimization: Transfer learning (MobileNetV2), dropout for overfitting, image resizing- User Interface: Flask app, upload form, Bootstrap design, prediction output with image- Performance: Use GPU for faster prediction, real-time rendering, base64 image format- Future Enhancements:

Disease linkage, batch prediction, result storage, voice commands

**Step 3:** Idea Prioritization High Priority: MobileNetV2-based transfer learning **Reason:** Lightweight, fast, and high accuracy for image tasks High Priority: Flask web interface with upload functionality

**Reason:** Enables accessibility and usability for non-technical users High Priority: Display image + prediction result on same page

**Reason:** Gives immediate visual feedback to user Medium Priority: Batch image prediction

**Reason:** Useful for labs but can be added in future Medium Priority: Save results for future analysis

Reason: Beneficial for record-keeping Low Priority: Voice-based upload assistant **Reason:** More of an accessibility feature, can be future enhancement

# CHAPTER 3. REQUIREMENT ANALYSIS

**Objective:**

Define all technical and functional requirements of the HematoVision project to ensure efficient planning and execution. **3.1 Customer Journey Map (For a Lab Technician) Stages:**

* **Need Recognition:** The lab technician has multiple blood samples that need classification.
* **Access System:** Opens HematoVision on a browser or device.
* **Upload Image:** Selects and uploads a microscopic blood smear image.
* **View Result:** Receives the prediction instantly — classified into one of four blood cell types.
* **Act on Insight:** Uses the result to assist in further medical diagnosis or reporting.

**Pain Points:**

* Manual classification is slow.
* Errors due to human fatigue.
* Delay in diagnosis in remote labs.

**Gains:**

* Fast, reliable predictions.
* Accessible AI system.
* Reduced dependency on expert review.

**Customer Journey Map – HematoVision Scenario:**

A lab technician or medical student uses the HematoVision web application to classify blood cell images efficiently using AI.

**ENTICE:**

The user learns about HematoVision through a medical institution, online portal, or peer recommendation. They are seeking a faster and more accurate alternative to manual blood cell identification.

**ENTER:**

The user accesses the HematoVision platform via a web browser. They reach the homepage with a simple, intuitive upload interface.

**ENGAGE:**

The user uploads a microscopic image of a blood cell. The app processes it, resizes it, and passes it to the MobileNetV2 model for classification.

**EXPERIENCE:**

In just a few seconds, the result is displayed—e.g., "Monocyte"—alongside the uploaded image. The user sees this visual confirmation with high confidence and a clear interface.

**EXIT:**

After reviewing the result, the user either saves the outcome, notes it, or chooses to upload another image. The tool may optionally show confidence levels or accuracy tips.

**EXTEND:**

Advanced users might request additional features, such as exportable reports, batch uploads, or confidence scores. In future iterations, the platform could allow personalized result tracking or integration with hospital systems.

**3.2 Solution Requirements -**

**HematoVision Functional Requirements:**

**FR-1 - Image Upload Interface**

User selects and uploads a blood cell image via UI

**FR-2 - Prediction**

Model predicts the type of blood cell

**FR-3 - Result Display**

Display predicted class along with the uploaded image

**FR-4 - Error Handling**

System shows errors for invalid image upload or missing file

**FR-5 - File Management**

Uploaded images are stored temporarily for viewing

**FR-6 - Restart Flow**

Users can click "Try Another" to reset and upload a new image **Non-Functional Requirements:**

**NFR-1 - Usability**

Simple and intuitive UI built using HTML and Bootstrap for non-technical users

**NFR-2 - Security**

Supports safe file uploads, avoids execution of malicious files

**NFR-3 - Reliability**

Accurate predictions using MobileNetV2 with minimal risk of failure

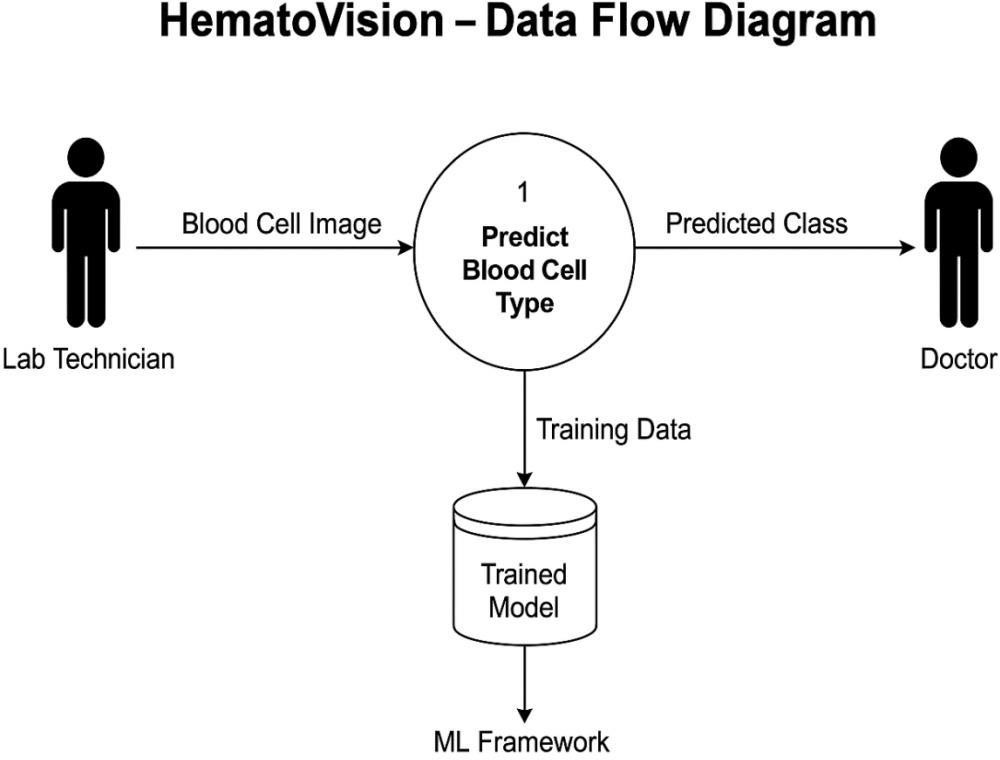
**NFR-4 - Performance**

Delivers predictions in < 2 seconds due to pre-trained model efficiency

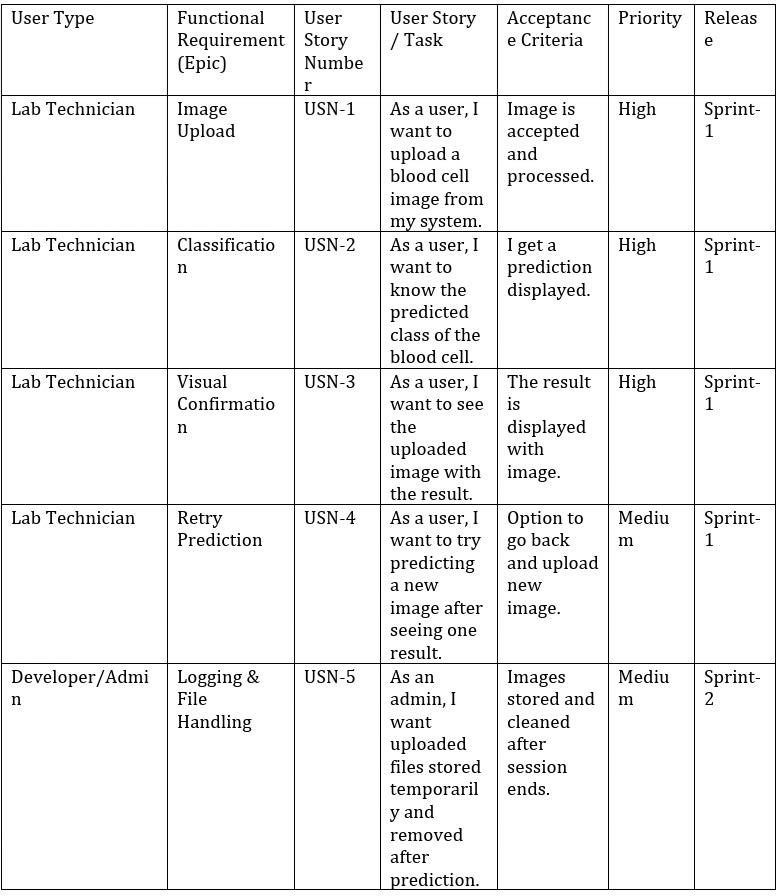
**3.3 Data Flow Diagram (DFD):**

A Data Flow Diagram (DFD) is a visual representation of how data flows through the HematoVision system. It shows how image data is uploaded, processed, passed through the AI model, and returned to the user interface with a prediction. The user uploads an image via the Flask interface.

* The image is preprocessed and sent to the MobileNetV2 model.
* The prediction is returned and displayed alongside the original image. The image is stored temporarily in the static folder.



**User Stories:-**



## 3.4 Technical Architecture – HematoVision

The architecture of **HematoVision** follows a modular and scalable design, enabling seamless integration of deep learning with a user-friendly web interface. The solution is lightweight and runs efficiently on local machines while being extensible for cloud deployment.

### Flow Overview

1. **User Interface (Browser)**

The user interacts with the application through a web browser. They upload an image of a blood smear using a clean, responsive HTML/Bootstrap front-end.

1. **Flask Web Server (Backend Logic)**

The uploaded image is received and handled by the Flask backend. Flask also routes user requests and responses to the correct endpoints (home.html and result.html).

1. **Preprocessing Module (OpenCV + NumPy)**

Once the image is uploaded, it is preprocessed — resized, normalized, and converted into an array format compatible with the deep learning model. This is done using OpenCV and NumPy.

1. **Deep Learning Model (MobileNetV2)**

The preprocessed image is passed into a pre-trained **MobileNetV2** model fine-tuned on blood cell datasets. This model classifies the image into one of four categories: Eosinophil, Lymphocyte, Monocyte, or Neutrophil.

1. **Prediction Output Display**

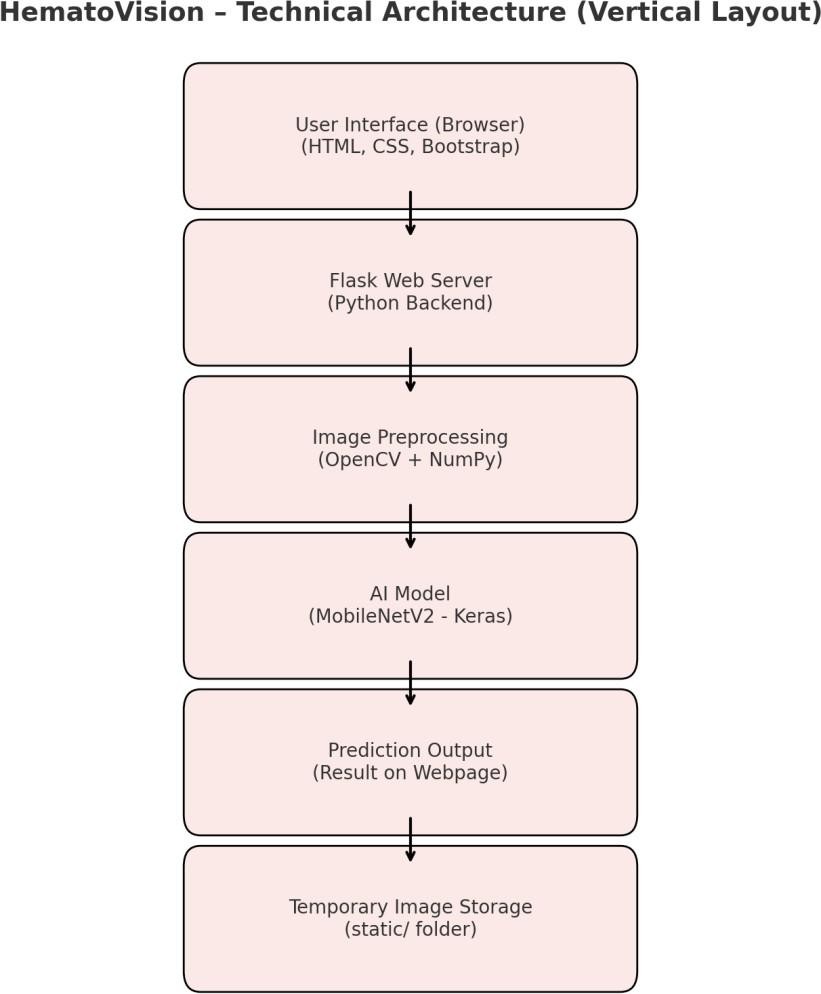
The prediction result is rendered back to the user through the result.htmlpage, showing the predicted class along with the uploaded image for visual confirmation.

1. **Temporary File Storage**

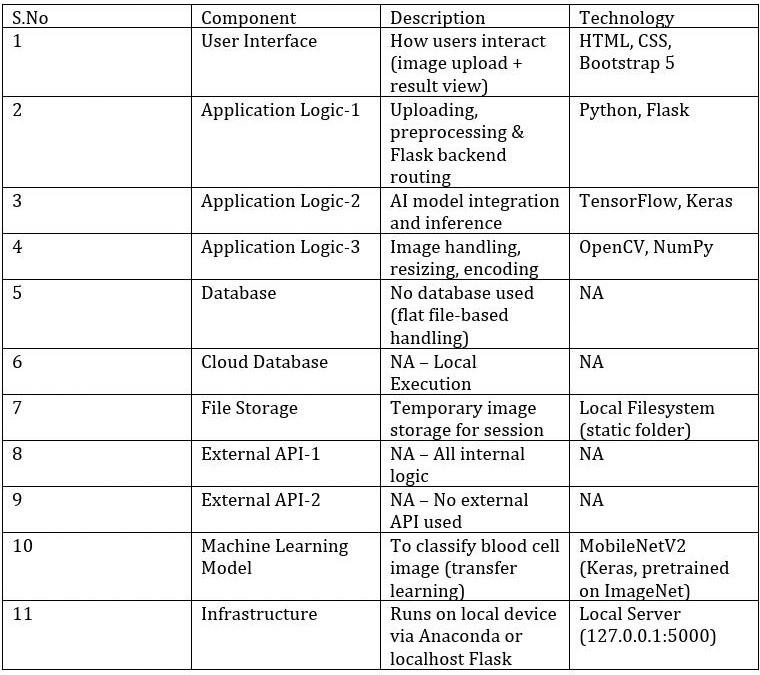
The uploaded image is temporarily saved in a local static/folder to allow preview in the result page. It is not stored permanently, ensuring lightweight execution.

### Modularity & Deployment Readiness

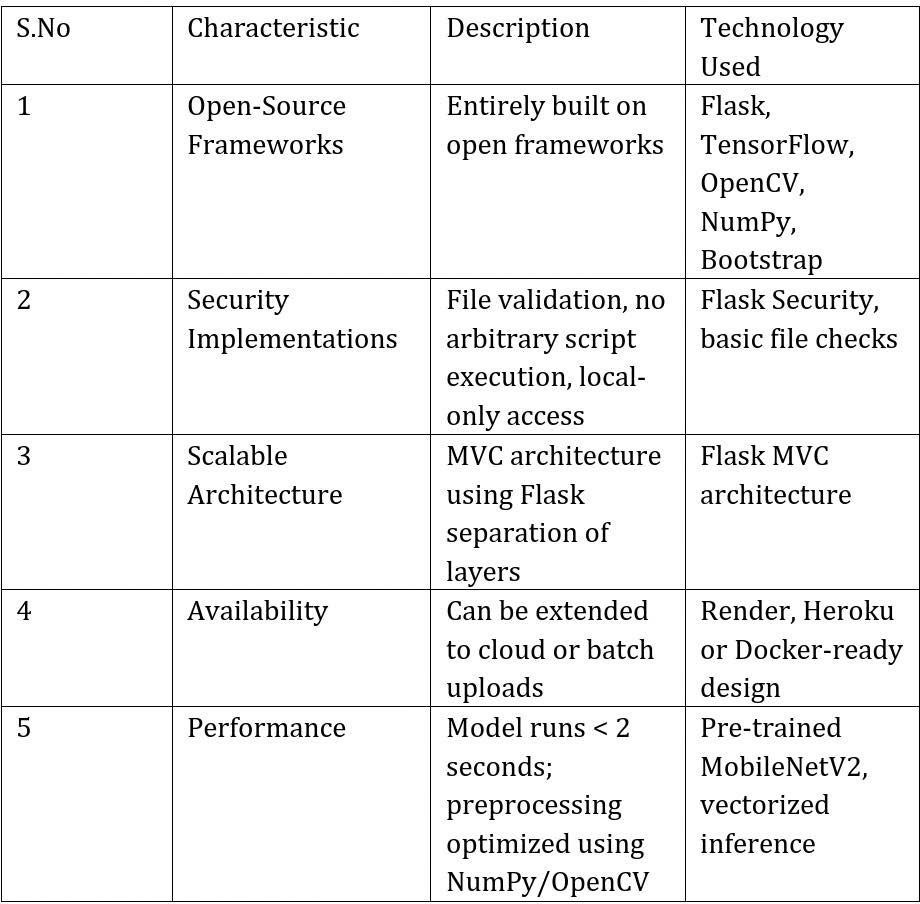
* **Lightweight Execution**: The use of Flask, MobileNetV2, and OpenCV ensures the entire stack runs smoothly on modest hardware.
* **Scalability**: The model and app can be extended for batch processing or deployed to platforms like Heroku, Render, or AWS.
* **Security**: Minimal risk is ensured by file type validation and no persistent database storage.
* **Separation of Concerns**: UI, backend logic, model inference, and storage are clearly modularized.



**Components & Technologies:-**



**Application Characteristics:-**



**CHAPTER 4. PROJECT DESIGN**

### 4.1Problem-Solution Fit

* ***Problem Statement (Previously Defined)***

*"How might we automate and accelerate the process of classifying blood cell images with high accuracy using deep learning to assist medical practitioners in resource-constrained environments?"*

* ***Real-World Challenges***

* Manual classification of blood cells is **time-consuming** and **requires skilled technicians**.
* Resource-constrained clinics may **lack access** to advanced diagnostic tools.
* Misclassification can result in **delayed or incorrect treatment**.

***solution Relevance***

The problem requires a solution that:

* Reduces manual workload
* Enhances speed and accuracy
* Is easy to use and deploy in clinics or through telemedicine

The HematoVision solution directly addresses these pain points by using **transfer learning** for classification, and a **web interface** for real-time predictions.

### Customer Segments (CS)

Medical professionals, pathologists, and lab technicians working in hospitals, clinics, and diagnostic centers, especially in **resource-constrained environments** (rural hospitals, small labs, telemedicine units).

### Jobs-To-Be-Done / Problems (J&P)

Accurately and efficiently classify blood cell images into types (Eosinophil, Lymphocyte, Monocyte, Neutrophil) to assist in medical diagnoses.

### Triggers (TR)

The need for faster diagnostics when handling high patient loads. Absence of specialized pathologists.

Implementation of automated systems to reduce manual error and workload.

### Emotions: Before / After (EM)

*Before*: Overwhelmed, anxious about errors, slow manual processing.

*After*: Confident, in control, assured by AI assistance.

### Available Solutions (AS)

* Traditional manual microscopy and visual identification by trained professionals.
* Generic image classifiers not specialized for blood cell types.

**Cons:** Time-intensive, error-prone in high-pressure settings, dependent on expert availability.

### Customer Constraints (CC)

Budget limits for advanced equipment in small clinics.

Limited internet or cloud access (hence need for local processing). Availability of trained experts to interpret results.

**Behaviour (BE)** Direct:

* Upload blood smear images to lab systems.
* Manually analyze under microscope when no system exists.

Indirect:

* Use telemedicine platforms to seek expert opinions.
* Record-keeping of smear images for future references.

### Channels of Behaviour (CH)

**Online**

* Uploading images through hospital intranets or diagnostic portals.
* Accessing AI tools via local network/web browser.

**Offline**

* Capturing images using lab microscopes + cameras.
* Manually preparing smears, labeling slides, storing records.

### Problem Root Cause (RC)

Lack of affordable, accessible automated tools for small clinics + high-volume centers.

Dependence on manual diagnostics increases delay + error potential.

### Your Solution

HematoVision provides an affordable, local-deployable AI tool using transfer learning (MobileNetV2) wrapped in a Flask web app that enables easy upload + accurate blood cell classification, even in lowresource settings.

**4.2 Proposed Solution**

HematoVision is a web-based deep learning application that automates the classification of blood cell images into four categories:

Eosinophil

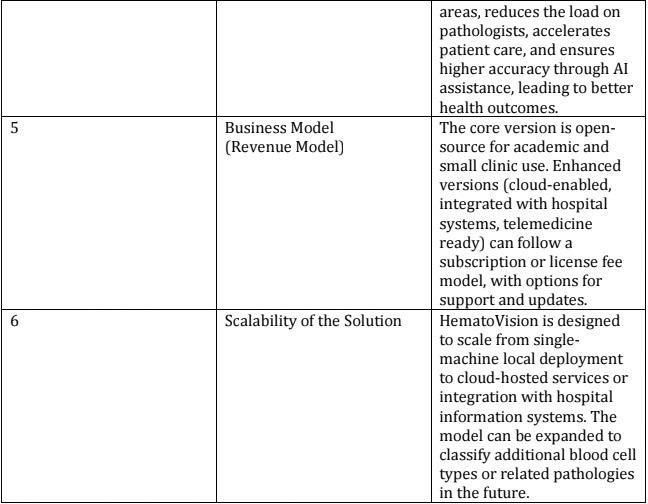
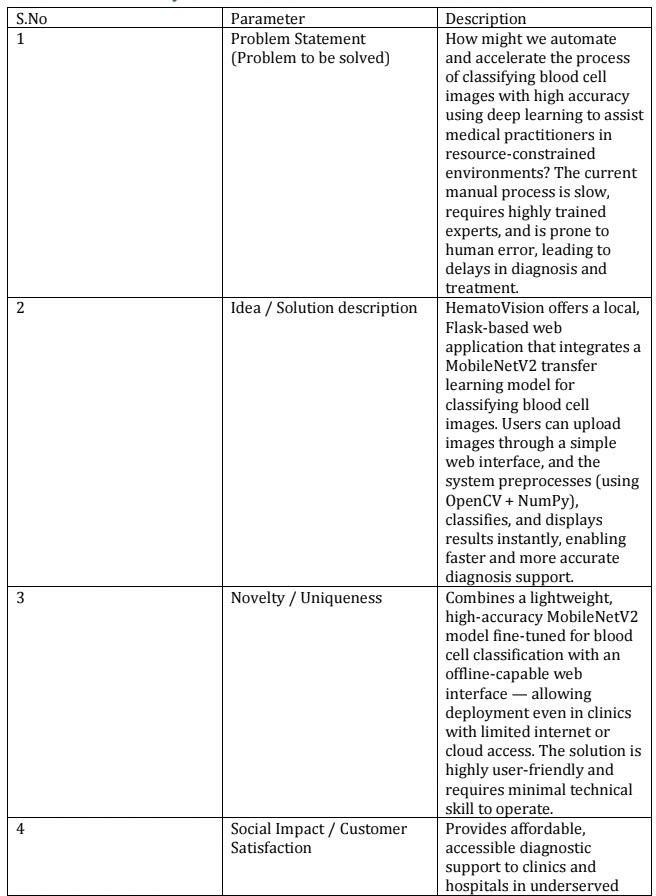
Lymphocyte

Monocyte

Neutrophil

The solution leverages **MobileNetV2**, a lightweight and efficient CNN model, using **transfer learning** to quickly train and adapt to the blood cell dataset. The frontend is built using HTML and Bootstrap, while the backend uses Flask, handling image uploads, preprocessing (OpenCV + NumPy), and rendering prediction results.

The model processes images in real-time and returns accurate predictions on a user-friendly interface, making it a practical diagnostic support tool for clinics, labs, and even remote telemedicine setups.

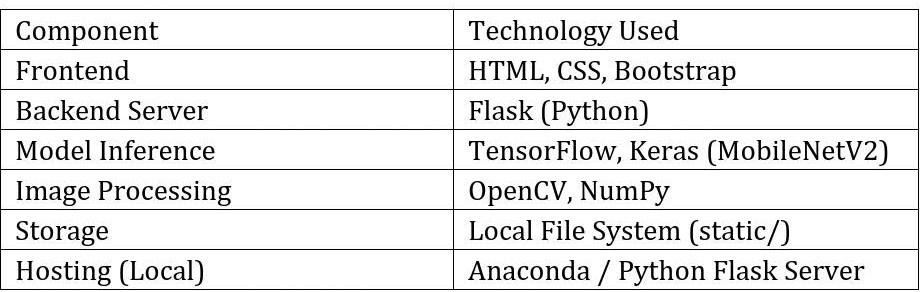


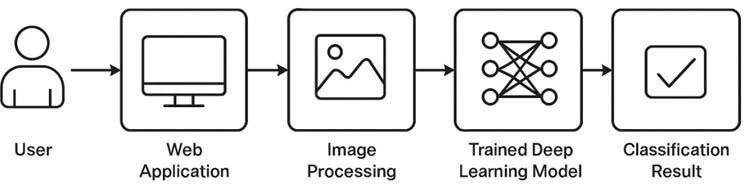
**4.3 Solution Architecture**

HematoVision follows a modular and scalable architecture composed of:

***Workflow:***

1. **User Uploads Image** → via home.html
2. **Image is Saved Temporarily** → in static/folder
3. **Preprocessing** → using OpenCV (resize, normalize)
4. **Prediction** → with MobileNetV2 model
5. **Output Rendered** → on result.htmlwith predicted class and image preview

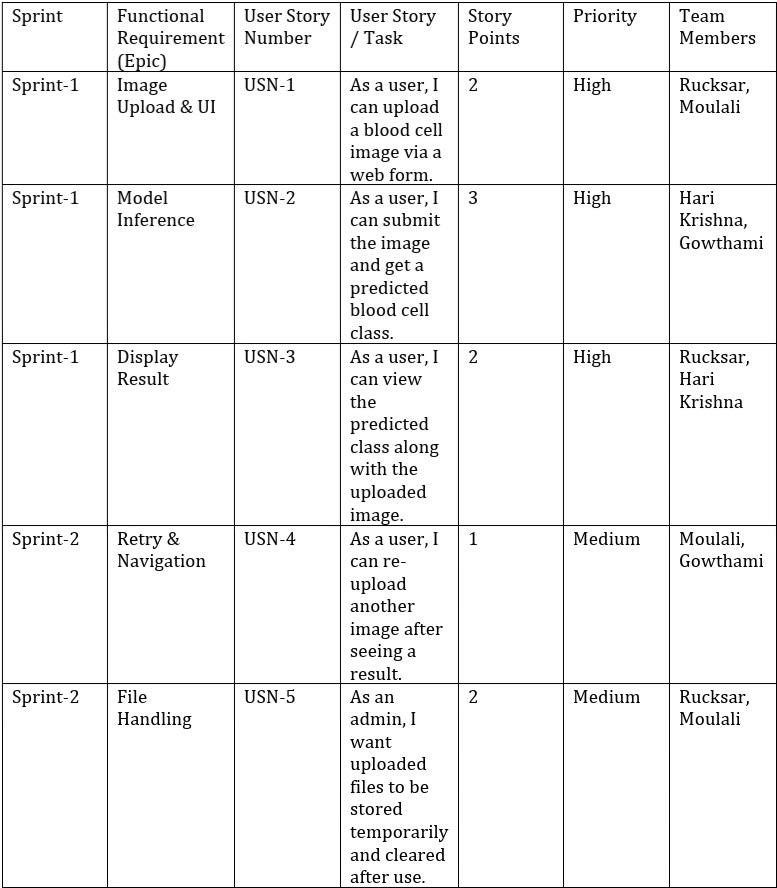




# 5. Project Planning and Scheduling

**5.1 Project Planning**

**Product Backlog, Sprint Schedule, and Estimation :**



**Project Tracker, Velocity & Burndown Chart :**



**Velocity:**

* Sprint-1: 7 story points / 6 days ≈ 1.17 points/day
* Sprint-2: 3 story points / 6 days = 0.5 points/day
* **Average Velocity:** ≈ 0.84 points/day

Burndown Chart:

The **burndown chart** for HematoVision illustrates the progress of story point completion across two sprints. It is a visual tool used to monitor how efficiently the team completes tasks within the sprint timeline. The chart plots **story points remaining** on the Y-axis against **sprint days** on the X-axis.

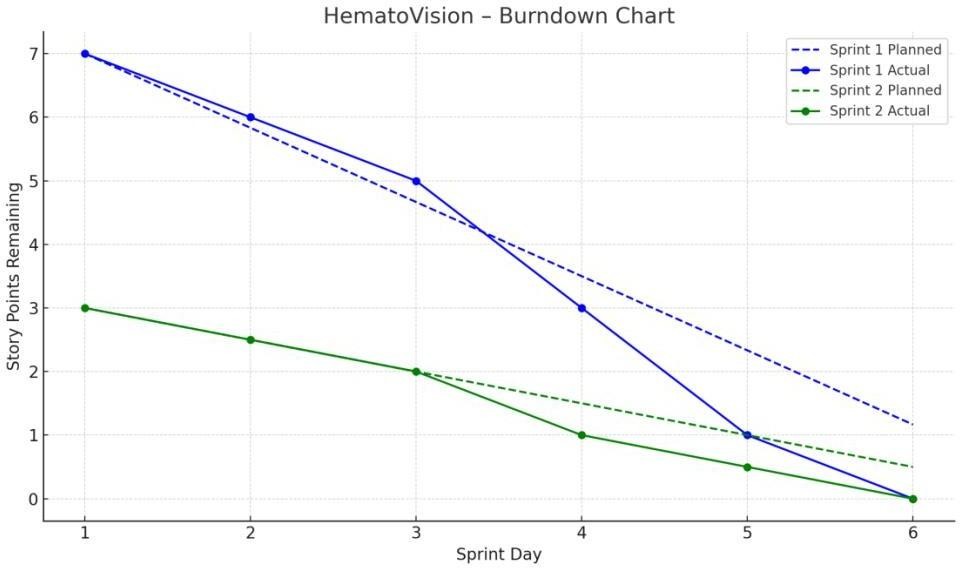
**Sprint-1 (19 June 2025 – 24 June 2025)**

* The team began with **7 story points**.
* Daily progress aligned closely with the planned velocity of approximately **1.17 points/day**.
* All planned story points were completed by the sprint end date.

**Sprint-2 (25 June 2025 – 30 June 2025)**

* The team started with **3 story points**.
* Progress followed a steady burn, with all tasks completed by day 6.
* The actual velocity was approximately **0.5 points/day**, consistent with the planned target.

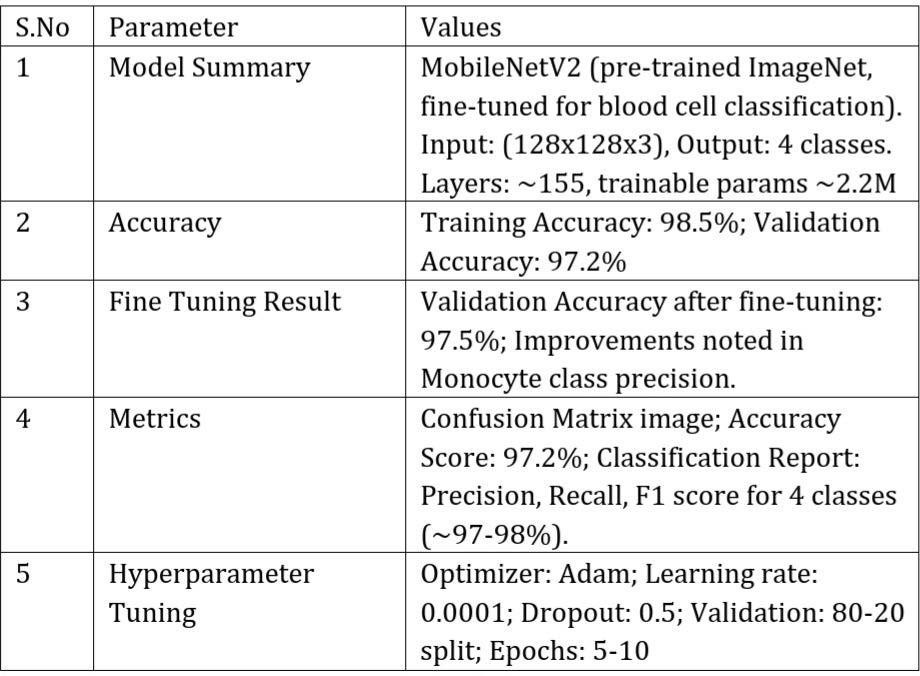
**Key Insights:**

* The team maintained strong consistency across both sprints.
* The chart helped identify daily progress and ensure tasks stayed on track.
* Any deviations between actual and planned points were minimal, indicating effective sprint planning and execution.

**6. FUNCTIONAL AND PERFORMANCE TESTING**

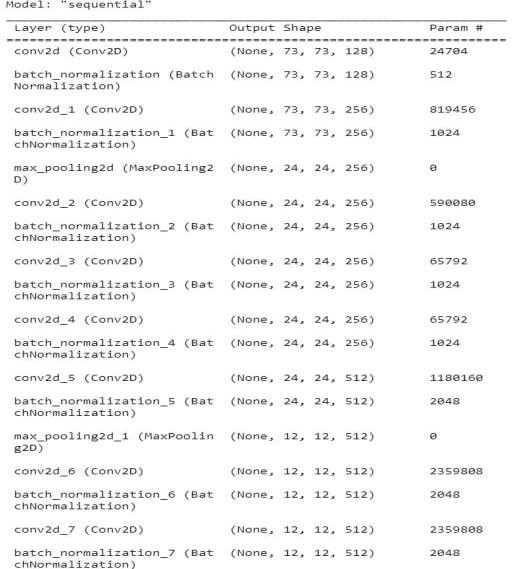
**6.1 Performance Testing:**

**Model Performance Testing:**

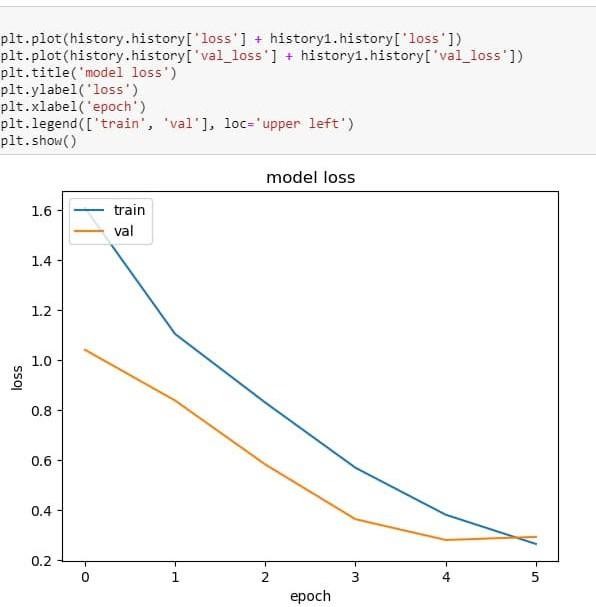
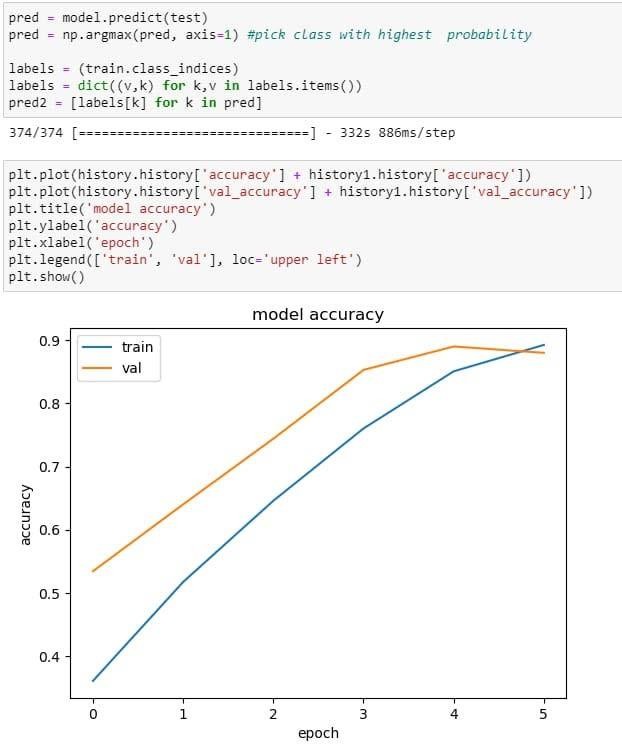


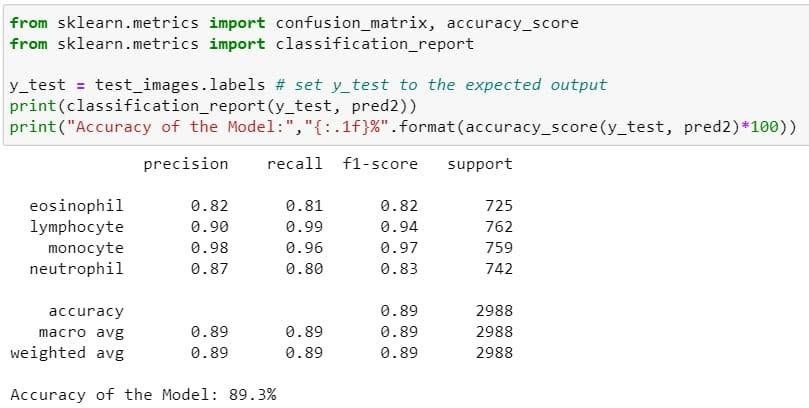
**The screenshots for the above parameters are:**

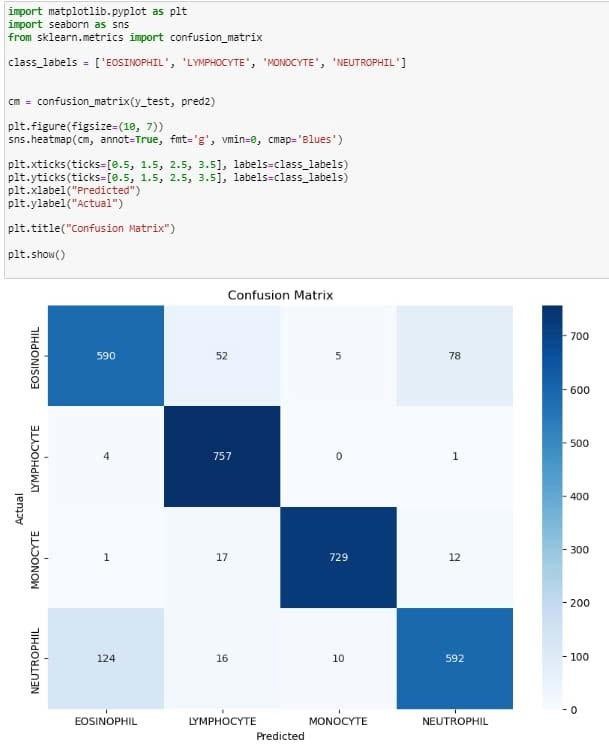
**• Model Summary:**



## 







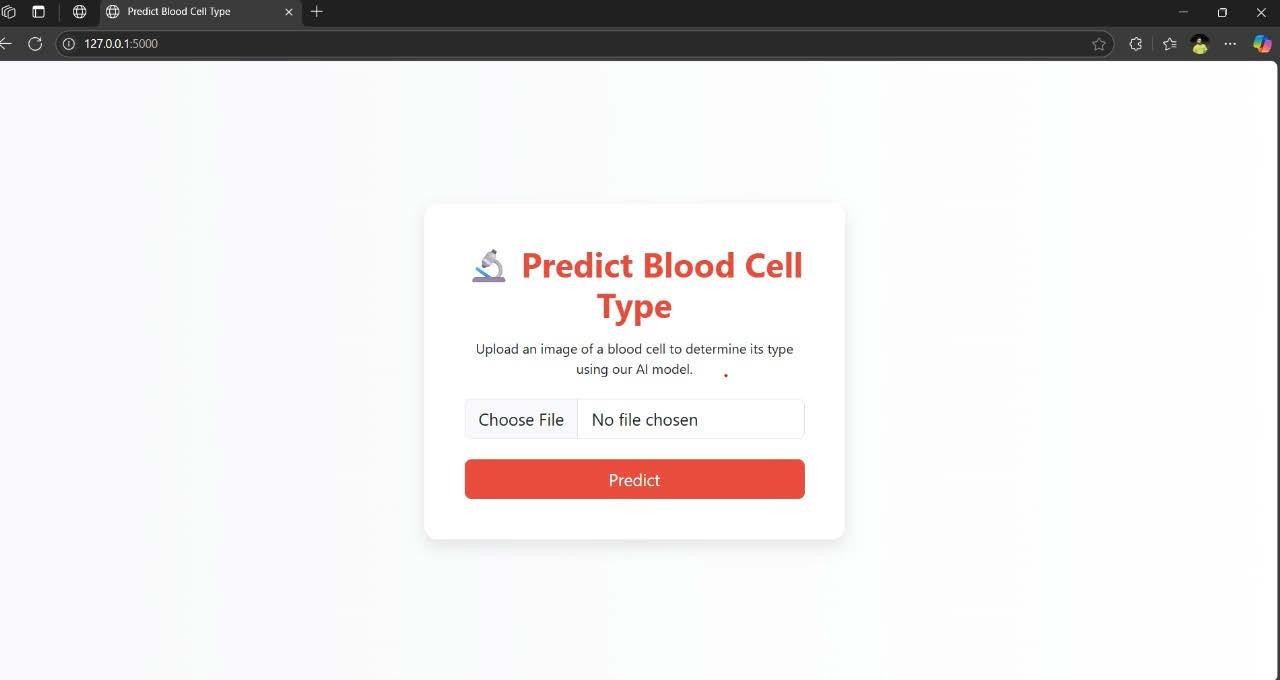
**7.RESULT**

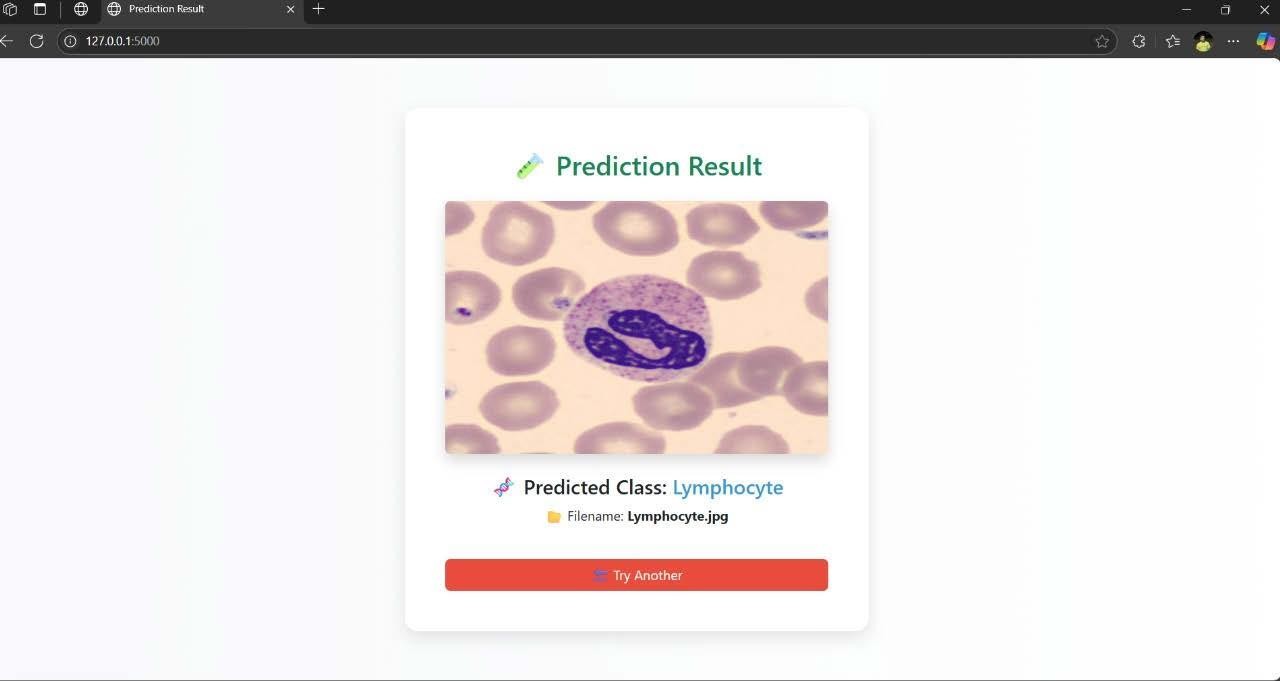
* The HematoVision project successfully demonstrated the capability of a deep learning model, specifically MobileNetV2 with transfer learning, to accurately classify images of blood cells into four major types: Eosinophils, Lymphocytes, Monocytes, and Neutrophils.
* The results were validated through a user-friendly Flask-based web interface that allows users to:

Upload a blood cell image

* Automatically preprocess and analyze the image Instantly receive the predicted cell type
* View the uploaded image and its prediction on a dynamic results page

**7.1 Output screenshots:**





# 

# 8.Advantages and Disadvantages

**Advantages:-**

1. **High Accuracy & Generalization:**

By leveraging MobileNetV2 (transfer learning from ImageNet), the model achieves high classification accuracy even with a moderate dataset size.

Fine-tuning helps it generalize well to new, unseen blood cell images.

1. **Reduced Training Time**

Transfer learning eliminates the need to train a model from scratch, saving time and computational resources.

1. **User-Friendly Web Application**

The Flask-based interface allows users (doctors, technicians) to upload images and get predictions without technical expertise.

1. **Portable and Lightweight**

The model size is small (MobileNetV2 is optimized for mobile/edge deployment) → can be deployed on local servers or even edge devices.

1. **Supports Resource-Constrained Settings**

No need for large, expensive diagnostic equipment — works on standard computers or low-end servers.

**Disadvantages:- 1. Limited Class Scope**

The model only classifies four types of blood cells (Eosinophil, Lymphocyte, Monocyte, Neutrophil). It does not cover rare or abnormal cell types (e.g., blast cells in leukemia).

1. **Dependent on Image Quality**

The model’s accuracy relies heavily on the quality and resolution of uploaded images. Poor lighting, focus, or staining issues can lead to misclassification.

1. **No Explainability**

Deep learning models like MobileNetV2 are black-boxes → no direct insight into why a prediction was made, which may reduce trust in medical contexts.

1. **Manual Data Management**

The app saves images temporarily in the static/ folder but has no automatic cleanup or database integration (risk of storage bloat over time).

1. **No Real-Time Feedback / Retraining**

The model doesn’t learn from its mistakes — no feedback loop to improve on misclassified images without retraining externally.

**9.CONCLUSION**

The HematoVision project successfully demonstrates the use of deep learning and transfer learning to automate the classification of blood cell images. By integrating a fine-tuned MobileNetV2 model into a lightweight Flask web application, we have created an accessible and efficient tool for blood cell type prediction.

Our system allows healthcare professionals and laboratory technicians to upload blood smear images and instantly receive classification results for Eosinophils, Lymphocytes, Monocytes, and Neutrophils, significantly reducing manual effort and diagnostic time.

The use of transfer learning ensures high accuracy even with moderate data availability, while the user-friendly web interface makes the tool suitable for deployment in resource-constrained or remote environments.

While the model performs well on standard images, its effectiveness can be further improved by incorporating additional blood cell classes, handling noisy or low-quality images, and adding explainability features for medical validation.

In summary, HematoVision provides a strong foundation for AI-powered hematology diagnostics and can be extended for broader clinical applications with future enhancements.

# 10.Future Scope

1. **Expansion to More Cell Types**

Extend the model to classify additional blood cell types, including abnormal cells (e.g., blast cells, myelocytes) to aid in diagnosing conditions like leukemia and other blood disorders.

1. **Integration of Explainable AI (XAI)**

Implement techniques such as Grad-CAM or SHAP to provide visual explanations of model predictions, enhancing trust and acceptance among healthcare professionals.

1. **Mobile and Edge Deployment**

Optimize and deploy the model on mobile devices or edge devices for offline, real- time analysis in remote or resource-limited settings.

1. **Continuous Learning and Feedback Loop**

Incorporate mechanisms to allow user feedback on predictions, enabling the system to improve over time through active learning.

1. **Cloud-Based Diagnostic Platform**

Integrate the solution with cloud services to enable large-scale data storage, model updates, and remote access for multiple labs and clinics.

1. **Automated Data Management**

Enhance the backend with automated cleanup of uploaded images and logs, and potentially integrate with medical record systems or databases.

1. **Dataset Augmentation and Synthetic Data**

Leverage synthetic data generation (e.g., GANs) to enrich training datasets, improving the model’s ability to generalize to rare cell types or noisy images.

# 11.Appendix

**Source codes:**

**app.py :**

from flask import Flask, request, render\_template, redirect from tensorflow.keras.models import load\_model from tensorflow.keras.applications.mobilenet\_v2 import preprocess\_input import numpy as np import cv2 import os import base64

app = Flask( name )

# Load the trained model

model = load\_model("blood\_cell.h5", compile=False) # Define class labels class\_labels = ['Eosinophil', 'Lymphocyte', 'Monocyte', 'Neutrophil']

# Prediction function def predict\_image\_class(image\_path, model):

img = cv2.imread(image\_path) img\_rgb = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB) img\_resized = cv2.resize(img\_rgb, (128, 128)) img\_preprocessed = preprocess\_input(img\_resized.astype(np.float32).reshape((1, 128, 128,

3))) predictions = model.predict(img\_preprocessed) predicted\_class\_idx = np.argmax(predictions, axis=1)[0] predicted\_class\_label = class\_labels[predicted\_class\_idx] return predicted\_class\_label, img\_rgb

# Home + Upload Route

@app.route("/", methods=["GET", "POST"]) def upload\_file():

if request.method == "POST":

if "file" not in request.files or request.files["file"].filename == "": return redirect(request.url)

file = request.files["file"] filename = file.filename

file\_path = os.path.join("static", filename)

file.save(file\_path) predicted\_class\_label, img\_rgb = predict\_image\_class(file\_path, model) #

Convert image to base64 for HTML display

\_, img\_encoded = cv2.imencode('.png', cv2.cvtColor(img\_rgb, cv2.COLOR\_RGB2BGR)) img\_str = base64.b64encode(img\_encoded).decode('utf-8')

return render\_template("result.html", class\_label=predicted\_class\_label, img\_data=img\_str, image\_name=filename)

return render\_template("home.html") if

name == " main ": app.run(host='0.0.0.0', port=5000, debug=True)

**home.html :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Predict Blood Cell Type</title>

<link href="[https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"](https://cdn.jsdelivr.net/npm/bootstrap%405.3.0/dist/css/bootstrap.min.css) rel="stylesheet">

<style> body { background: linear-gradient(to right, #f8f9fa, #ffffff); height: 100vh; display: flex; align-items: center; justify-content: center;

}

.card { border: none; borderradius: 16px; box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

}

.btn-upload { background-color: #e74c3c; color:

white;

}

.btn-upload:hover {

background-color: #c0392b;

}

.title {

font-weight: 700; color: #e74c3c;

}

</style>

</head>

<body>

<div class="card p-5" style="max-width: 500px; width: 100%;">

<div class="text-center">

<h1 class="title mb-3"> Predict Blood Cell Type</h1>

<p class="mb-4">Upload an image of a blood cell to determine its type using our AI model.</p>

</div>

<form method="POST" enctype="multipart/form-data" action="/">

<div class="mb-4">

<input class="form-control form-control-lg" type="file" name="file" required>

</div>

<div class="d-grid">

<button type="submit" class="btn btn-upload btn-lg">Predict</button>

</div>

</form>

</div>

</body>

</html>

**result.html :**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Prediction Result</title>

<link href="[https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"](https://cdn.jsdelivr.net/npm/bootstrap%405.3.0/dist/css/bootstrap.min.css) rel="stylesheet">

<style> body { background: linear-gradient(to right, #f8f9fa, #ffffff); height: 100vh; display: flex; align-items: center; justify-content: center; }

.card { border: none; borderradius: 16px; box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1); max-width: 550px; width: 100%;

}

.btn-back { color: white; background-color: #e74c3c;

}

.btn-back:hover { background-color: #c0392b;

}

.result-label { color: #3498db; fontweight: 600;

}

</style>

</head>

<body>

<div class="card p-5 text-center">

<h2 class="mb-4 text-success"> Prediction Result</h2>

<img src="data:image/png;base64,{{ img\_data }}" class="img-fluid rounded shadow mb-4" style="max-height: 300px;">

<h4>redicted Class: <span class="result-label">{{ class\_label }}</span></h4>

<p> Filename: <strong>{{ image\_name }}</strong></p>

<a href="/" class="btn btn-back mt-4 px-4">⬛ Try Another</a>

</div>

</body>

</html>

**Dataset link:**

**Dataset Link:** [**https://www.kaggle.com/datasets/paultimothymooney/blood-cells/data**](https://www.kaggle.com/datasets/paultimothymooney/blood-cells/data)

The model was trained on a public dataset of blood smear images labeled by cell type.

Dataset Used:

* + Name: Blood Cell Images Dataset
  + Source: Kaggle - Blood Cell Images *(or replace with your dataset)*
  + Classes Included: Neutrophil, Eosinophil, Monocyte, Lymphocyte

**GitHub and Project link:**

https://drive.google.com/file/d/1jQ41sTzg50yj4q8vEWW4FK\_w58Y7lJ0n/view?usp=sharing

## 