Project Report

Title: Hematovision : advanced blood cell classification using transfer learning

Team members:

1) Dontanhaktuni Venkata surya deepak.(Lead)
2) Achanta Lakshmi Prasanna
3) Bayye Manasa
4) Durga Charishma Lingolu

Date	29 June 2025
Team ID	LTVIP2025TMID46471
Project Name	Hematovision : advanced blood cell classification using transfer learning

Index / Table of Contents

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. IDEATION PHASE

2.1 Problem Statement

2.2 Empathy Map Canvas

2.3 Brainstorming

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

3.2 Solution Requirement

3.3 Data Flow Diagram

3.4 Technology Stack

4. PROJECT DESIGN

4.1 Problem–Solution Fit

4.2 Proposed Solution

4.3 Solution Architecture

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

7. RESULTS

7.1 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**: Provide a brief summary of your project, highlighting the goal of automating blood cell classification using deep learning.
- **1.2 Purpose**: Explain why the project is important, emphasizing its potential impact on medical diagnostics and the need for an automated solution.

2. IDEATION PHASE

2.1 Problem Statement

Manually identifying and classifying different types of blood cells under a microscope is time-consuming, error-prone, and requires skilled professionals. In resource-constrained settings, timely and accurate diagnostics are difficult to achieve. There is a need for an automated solution that assists in blood cell identification to support medical diagnostics.

2. Proposed Solution:

HematoVision is a deep learning-based web application that uses transfer learning to classify microscopic images of blood cells into categories like Neutrophils, Lymphocytes, Monocytes, and Eosinophils. The trained model is integrated into a user-friendly Flask interface, enabling users to upload an image and receive an instant prediction of the blood cell type.

3. Target Users:

- Medical laboratories and diagnostic centers.

- Healthcare professionals and pathologists
- Rural health workers and clinics with limited access to expert pathologists
- Medical students for educational purposes

4. Expected Outcome:

- A fully functional AI-powered blood cell classification tool
- Reduced manual workload and improved accuracy in blood cell identification
- Faster diagnosis support in medical setups
- A deployable web-based tool that can be integrated into existing lab workflows.

2.2 Empathy Map Canvas

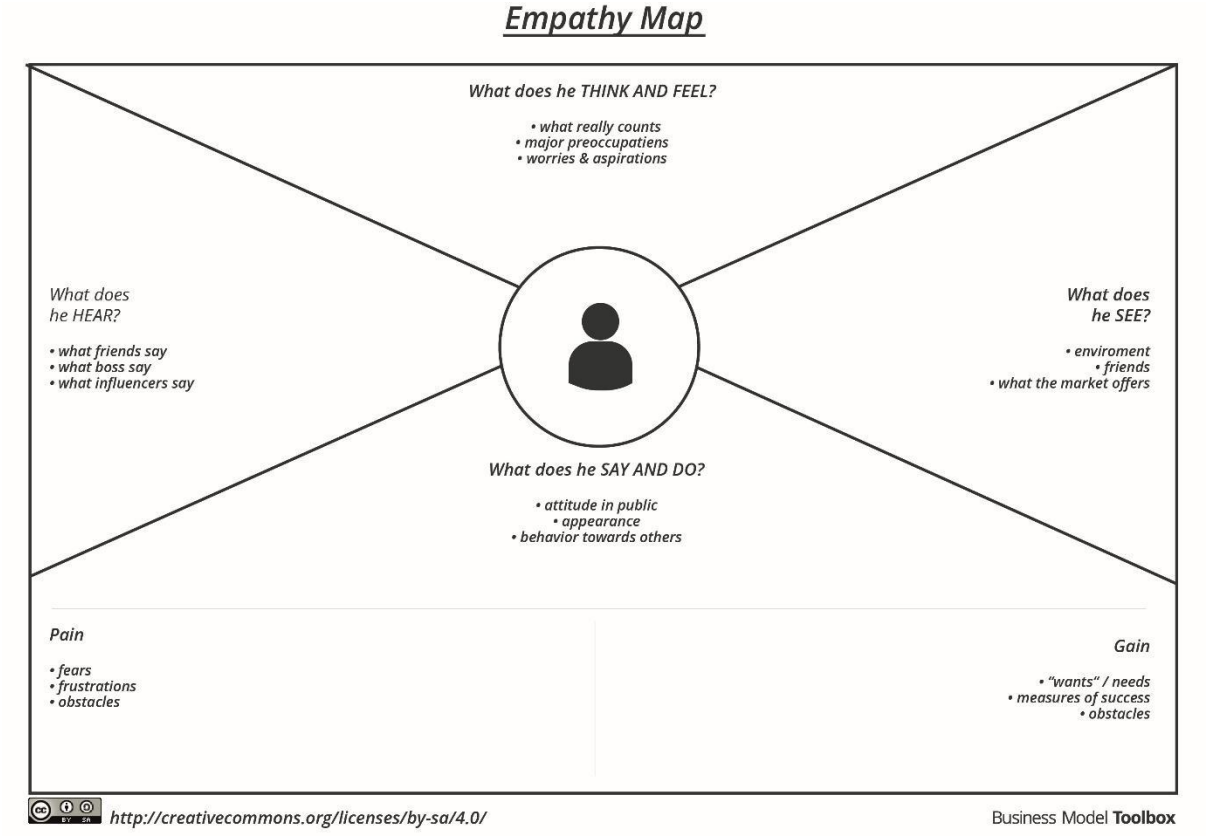
Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

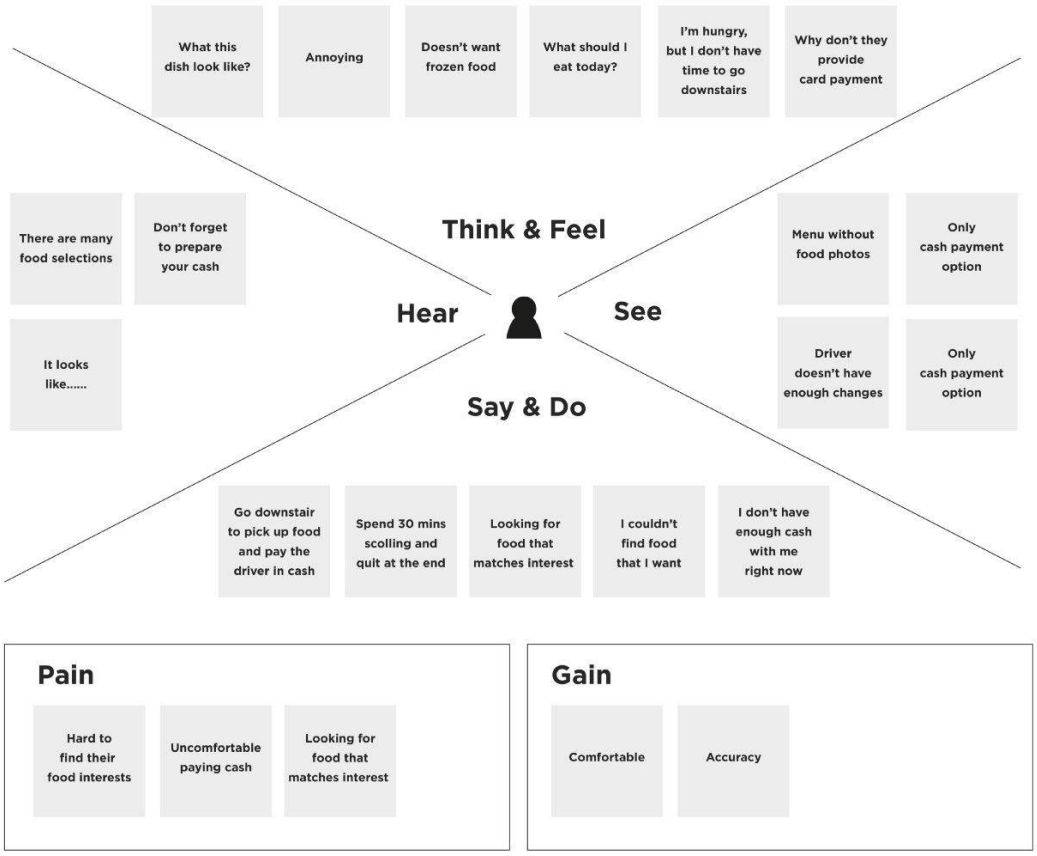
Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example:



Reference: https://www.mural.co/templates/empathy-map-canvas

Example: Food Ordering & Delivery Application



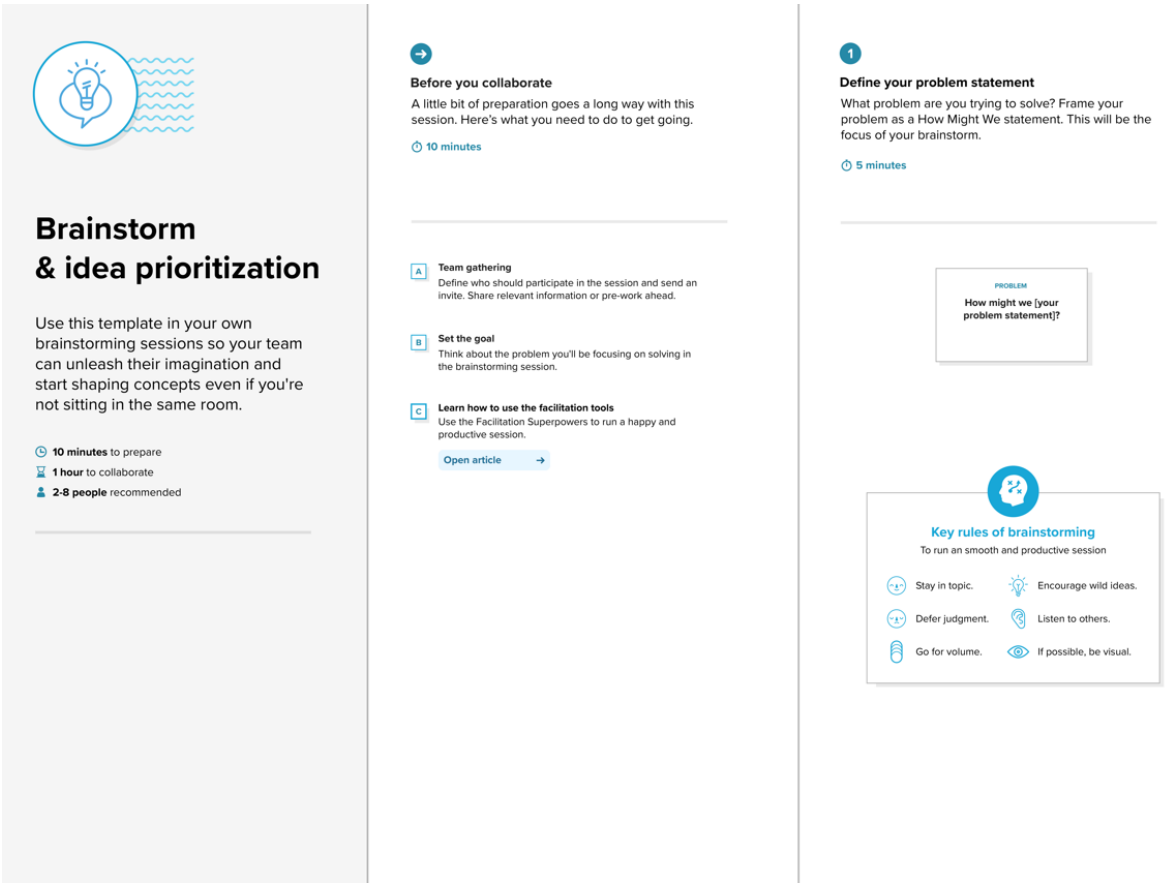
2.3 Brainstorming & Idea Prioritization

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

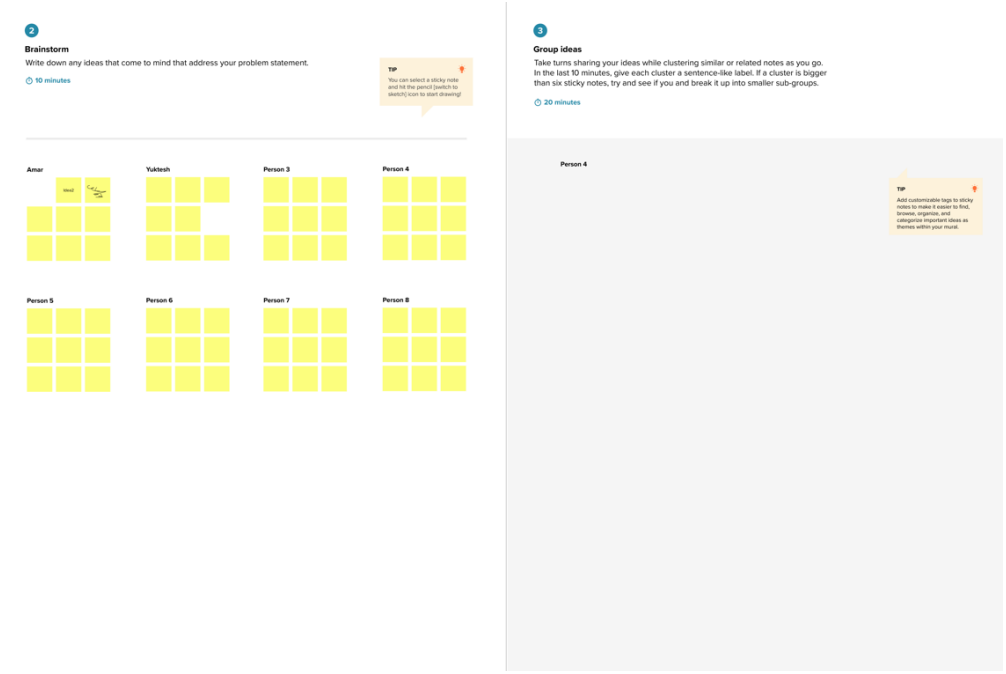
Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Reference: https://www.mural.co/templates/brainstorm-and-idea-prioritization

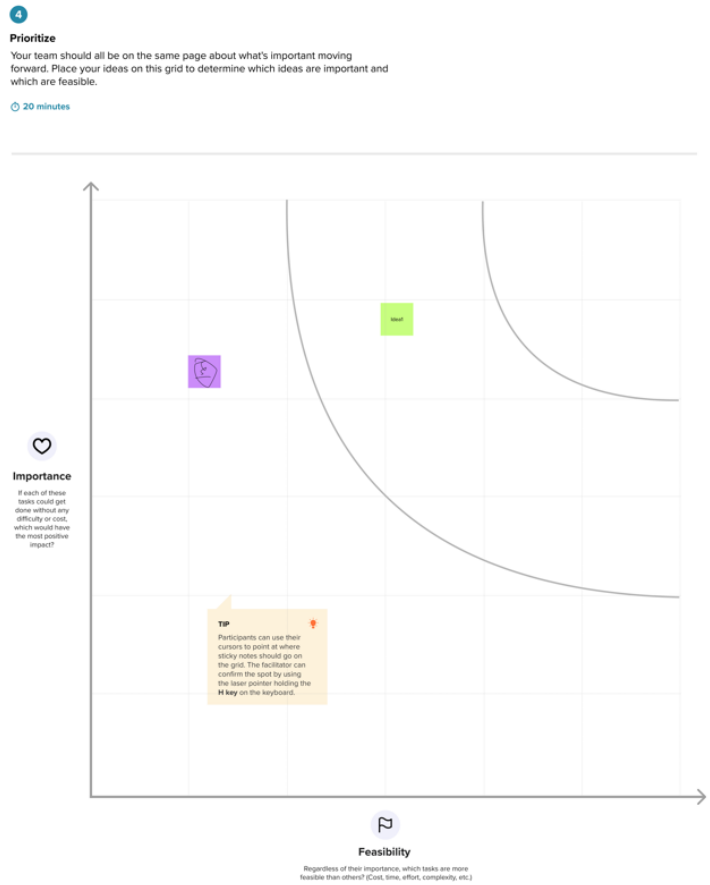
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



Step-3: Idea Prioritization



3. REQUIREMENT ANALYSIS

3.1 **Customer Journey Map**: Outline the steps users (pathologists, lab technicians) take from uploading a blood cell image to receiving a classification result.

3.2**Solution Requirement**:

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	Functional Requirement (Epic)	Registration through Form Registration through Gmail Registration through LinkedIN
FR-2	Image Upload Interface	User uploads microscopic image through web interface

FR-3	Prediction Trigger	App sends image to deep learning model for classification
FR-4	Result Display	Predicted class displayed on result page
FR-5	Model Management	Load pretrained Blood_Cell.h5 model into backend
FR-6	Session Reset	Option to upload another image for fresh prediction

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	Simple and intuitive UI with clear instructions
NFR-2	**Security**	No external storage of uploaded images
NFR-3	**Reliability**	Model should give consistent results on repeat inputs
NFR-4	**Performance**	Prediction response in under 3 seconds
NFR-5	**Availability**	Web app always available on localhost
NFR-6	**Scalability**	Model can be upgraded with new data or architectures

3.3 Data Flow Diagram

4 Data Flow Diagram (DFD)

5 Level 0: System Overview

6 This Data Flow Diagram (DFD) represents the main entities and their interactions within the blood cell classification system.

Entities:

1. User: Uploads the blood cell image.
2. Web Interface: Displays the image upload page and result.
3. Flask Backend: Handles requests, invokes the trained model, and sends the result.
4. Blood Cell Classifier (MobileNetV2): Processes the image and classifies the blood cell type.
5. Model Storage: Stores the trained model (Blood_Cell.h5).

Data Flows:
1. User uploads image → Web Interface → Flask Backend.
2. Flask Backend sends image → Blood Cell Classifier.
3. Blood Cell Classifier returns prediction → Flask Backend.
4. Flask Backend returns result → Web Interface → User.

7 Level 1: Flask Backend Process

8 This diagram zooms into the Flask Backend, detailing its data processing.
Entities:
1. User: Initiates the image upload and receives the classification result.
2. Flask Backend: Receives the image, invokes the model, and returns the result.

Processes:
1. Upload Image: User uploads the image to Flask.
2. Process Image: Flask sends the image to the classifier.
3. Return Result: Flask sends the result back to the Web Interface.

Data Stores:
1. Model Storage: Stores Blood_Cell.h5.

Data Flows:
1. User uploads image → Flask Backend.
2. Flask Backend sends image to the model → Blood Cell Classifier.
3. Blood Cell Classifier sends the classification result → Flask Backend.
4. Flask Backend returns result to Web Interface → User.

9 User Stories

10 User Stories

User Story Number	User Story / Task	Acceptance Criteria	Priority	Release
USN-1	As a user, I can upload an image of a blood cell to the application for classification.	The image is uploaded successfully. The system sends the image for classification.	High	Sprint-1
USN-2	As a user, I want to receive a prediction of the blood cell type after	The prediction is displayed correctly. The user can view and interpret the result.	High	Sprint-1

	uploading an image.			
USN-3	As an administrator, I can access the system logs to monitor the prediction system's performance.	Logs are available and display prediction statistics. The system records errors or exceptions.	Medium	Sprint-2
USN-4	As a user, I can upload another image without needing to refresh the page.	The page clears previous results and allows uploading a new image.	Medium	Sprint-1

3.4 Technology Stack

Technical Architecture:

This project implements an AI-based blood cell classification system that uses deep learning and transfer learning to automate the identification of different types of white blood cells. The architecture supports image ingestion, preprocessing, model prediction, and Flask-based deployment to assist medical professionals with diagnostics.

📌 **Architecture Overview:**

1. Image Ingestion (Microscopic cell images in JPEG format)

2. Data Preprocessing & Augmentation (TensorFlow, Keras)

3. Model Building using Transfer Learning (MobileNetV2 + Dense layers)

4. Model Evaluation & Export (Saved as bloodell.h5)

5. Flask Web App for Image Upload & Prediction (HTML, Flask, TensorFlow)

6. Deployment Readiness (GitHub + Local Testing)

Table-1: Components & Technologies

S.No	Component Description	Technology
1	User Interface – image upload & result view	HTML, Bootstrap

2	Application Logic – image preprocessing & prediction	Python, TensorFlow, Keras
3	Model – blood cell classification	MobileNetV2 (Transfer Learning), Dense layers
4	Storage – model and sample images	Local .h5 file, static/uploads folder
5	Framework – backend web server	Flask
6	Development Environment	Google Colab, Jupyter Notebook
7	Hosting & Deployment	Localhost (Flask), GitHub (project source)

Table-2: Application Characteristics

S.No	Characteristics Description	Technology / Tools Used
1	Open-Source Frameworks	TensorFlow, Keras, Flask
2	Security Considerations	Local image storage, no external API exposure
3	Scalable Design	Easily extendable with new model versions and datasets
4	Availability	Web interface available 24/7 on localhost or web host
5	Performance	Efficient MobileNetV2 architecture with GPU support

4. PROJECT DESIGN

Problem–Solution Fit

1. Problem Statement

Manual classification of blood cells is a time-consuming and error-prone task that pathologists and lab technicians perform. This results in inefficiencies, errors in diagnosis, and delays in patient treatment plans. The need for an automated system that can classify blood cells accurately and rapidly is evident. The existing process is prone to fatigue and human error, leading to a potential risk in medical decision-making.

2. Solution Description

HematoVision leverages a deep learning-based solution to automate the classification of blood cells using MobileNetV2, a pre-trained model fine-tuned for blood cell images. The system allows users to upload images of blood cells, which the model processes and classifies into four types: Eosinophil, Basophil, Monocyte, and Lymphocyte. This solution reduces manual errors and significantly speeds up the diagnostic process, enabling faster and more accurate diagnoses.

3. Customer/Target Audience

The primary target audience for HematoVision includes pathologists, lab technicians, and medical professionals who rely on blood cell classification for diagnostic purposes. The solution also caters to medical institutions that require an efficient, scalable way to classify blood cells with accuracy.

4. Customer Needs and Behaviors

The customers face the challenge of manual and time-consuming blood cell classification, which is prone to errors and inefficiencies. They need a fast, accurate, and reliable method to classify blood cells without the heavy reliance on human labor. The solution should fit into their current workflow and help them speed up diagnostics, reduce errors, and increase operational efficiency.

5. Solution Fit

HematoVision addresses these customer needs by providing an automated solution for blood cell classification using advanced machine learning techniques. The web-based tool is user-friendly and does not require significant changes to existing workflows. It automates the image classification process, offering a quick and reliable alternative to manual methods, and ultimately improves the speed and accuracy of diagnoses.

6. Key Metrics for Success

Success for HematoVision will be measured by the following key metrics:
1. **Reduction in Classification Time**: Faster image processing compared to manual methods.
2. **Accuracy Improvement**: Increased accuracy in blood cell classification.
3. **Customer Adoption Rate**: The rate at which pathologists, lab technicians, and medical facilities adopt the solution.
4. **Operational Efficiency**: Improved efficiency in medical workflows and diagnostics.

4.2 Proposed Solution

Project team shall fill the following information in the proposed solution template.

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Manual classification of blood cells is a time-consuming and error-prone task typically performed by pathologists and lab technicians. This process often results in delays and inaccuracies, affecting the overall diagnosis and treatment plan. There is a growing need for an automated system that can classify blood cells quickly and accurately.
2.	Idea / Solution description	Manual classification of blood cells is a time-consuming and error-prone task typically performed by pathologists and lab technicians. This process often results in delays and inaccuracies, affecting the overall diagnosis and treatment plan. There is a growing need for an automated system that can classify blood cells quickly and accurately. HematoVision proposes a deep learning-based solution to automate the classification of blood cells. The system leverages MobileNetV2, a pre-trained deep learning model, fine-tuned on a dataset of blood cell images. The solution is designed as a web-based tool where users can upload blood cell images, and the system will classify them into one of four categories: Eosinophil, Basophil, Monocyte, and Lymphocyte. The tool offers quick and accurate predictions, significantly

		reducing manual efforts and errors in diagnosis.
3.	Novelty / Uniqueness	The uniqueness of this solution lies in its use of MobileNetV2 with transfer learning for classifying blood cells. By leveraging a lightweight pre-trained model, the solution can provide accurate classifications while requiring minimal computational resources. This approach not only speeds up the process but also makes the solution accessible to medical facilities with limited infrastructure. Additionally, the web-based interface makes it easy to use in various healthcare settings.
4.	Social Impact / Customer Satisfaction	HematoVision has a significant social impact by providing a tool that enhances diagnostic efficiency and accuracy in medical labs. Pathologists and technicians can now classify blood cells faster, leading to quicker diagnoses and improved patient outcomes. By automating a critical yet time-consuming task, HematoVision can help reduce human error, optimize workflow, and free up medical professionals for more complex tasks. The system is user-friendly, enabling professionals with limited technical knowledge to use it effectively, thereby improving customer satisfaction and trust in the healthcare system
5.	Business Model (Revenue Model)	The proposed solution can be offered as a Software-as-a-Service (SaaS) platform, where medical facilities and laboratories can subscribe to the system for a monthly or annual fee. Alternatively, it could be integrated into lab equipment sales as a bundled

		offering. Another revenue stream could include licensing the technology to large-scale hospitals and diagnostic centers. Additionally, a freemium model can be introduced, offering basic features for free with premium services such as advanced analytics and integration with lab management systems available through paid plans.
6.	Scalability of the Solution	HematoVision is designed to be scalable in terms of both data volume and application usage. The model can be trained on a larger dataset to improve its accuracy and robustness over time. The system is also capable of handling a high number of users concurrently, as it is built on a web-based architecture with Flask. As the adoption of the solution grows, additional servers or cloud hosting can be integrated to scale the solution. Furthermore, the system's flexibility allows for the addition of more blood cell classes or integration with other medical imaging tools, expanding its applicability to other areas of healthcare.

4.3 Solution Architecture

The solution for HematoVision is designed to automate the process of classifying blood cells using deep learning techniques. This document provides an overview of the system's architecture, the key components involved, and the data flow within the system.

Key Components

1. **User Interface (Frontend)**:
- **HTML + Bootstrap**: Provides a responsive and user-friendly interface for image upload and result display.
- **Key Features**: Image upload form, result display, and option to upload multiple

images sequentially.

2. **Backend (Flask Application)**:
- **Flask (Python)**: Handles HTTP requests, processes image data, and communicates with the trained model for prediction.
- **Key Features**: Image upload handling, communication with the MobileNetV2 model, and result return.

3. **Model (MobileNetV2 with Transfer Learning)**:
- **TensorFlow/Keras**: MobileNetV2 is pre-trained and fine-tuned for blood cell classification.
- **Model File**: **Blood_Cell.h5** is used for inference.
- **Key Features**: Image preprocessing, model inference, and classification results for four blood cell types (Eosinophil, Basophil, Monocyte, Lymphocyte).

4. **Data Flow**:
- **User Uploads Image**: The user uploads a blood cell image via the web interface.
- **Flask Backend Processing**: The backend processes the image and communicates with the model for classification.
- **Model Prediction**: MobileNetV2 predicts the blood cell type and returns the result.
- **Result Display**: The Flask backend sends the result back to the user interface for display.

5. **Model Storage**:
- **Data Store**: The trained model is stored as **Blood_Cell.h5** and accessed during inference.

Data Flow

Entities:
1. **User**: Uploads the blood cell image.
2. **Web Interface**: Displays the image upload page and result.
3. **Flask Backend**: Handles image processing, invokes the model, and returns results.
4. **MobileNetV2 Model**: Classifies the blood cell image.
5. **Model Storage**: Stores the trained model.

Data Flow:
1. **User Uploads Image** → Web Interface → Flask Backend.
2. **Flask Backend Sends Image** → MobileNetV2.
3. **MobileNetV2 Returns Prediction** → Flask Backend.
4. **Flask Backend Sends Result** → Web Interface → User.

5. **PROJECT PLANNING & SCHEDULING**
6. **Product Backlog, Sprint Schedule, and Estimation (4 Marks)**
7. **Product Backlog, Sprint Schedule, and Estimation**
8. **Product Backlog**

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Preparation	USN-1	Upload blood cell images for classification.	3	High
Sprint-1	Model Integration	USN-2	Integrate the pre-trained MobileNetV2 model for classification.	4	High
Sprint-1	Backend Setup	USN-3	Set up Flask backend to handle image upload and inference.	5	High
Sprint-2	Frontend Integration	USN-4	Design the web interface for blood cell image upload and result display.	4	High
Sprint-2	Prediction Display	USN-5	Display classification result (blood cell type) on the frontend.	3	High
Sprint-2	UI Testing	USN-6	Test the frontend for proper display and interaction.	3	High
Sprint-3	Model Evaluation	USN-7	Evaluate the model accuracy with test data and optimize.	4	Medium
Sprint-3	Dashboard Setup	USN-8	Set up final result dashboard for predictions	4	Medium

			and performance metrics.		
Sprint-3	Testing & Optimization	USN-9	Perform performance testing of the backend and model integration.	4	Medium
Sprint-4	Documentation	USN-10	Create documentation for setup, usage, and results.	6	High
Sprint-4	Final Review	USN-11	Review and finalize the project for submission.	6	High

9. Sprint Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	20 June 2025	25 June 2025	25 June 2025
Sprint-2	18	6 Days	26 June 2025	1 July 2025	1 July 2025
Sprint-3	16	6 Days	2 July 2025	7 July 2025	7 July 2025
Sprint-4	12	6 Days	8 July 2025	13 July 2025	13 July 2025

10. Project Tracker, Velocity & Burndown Chart

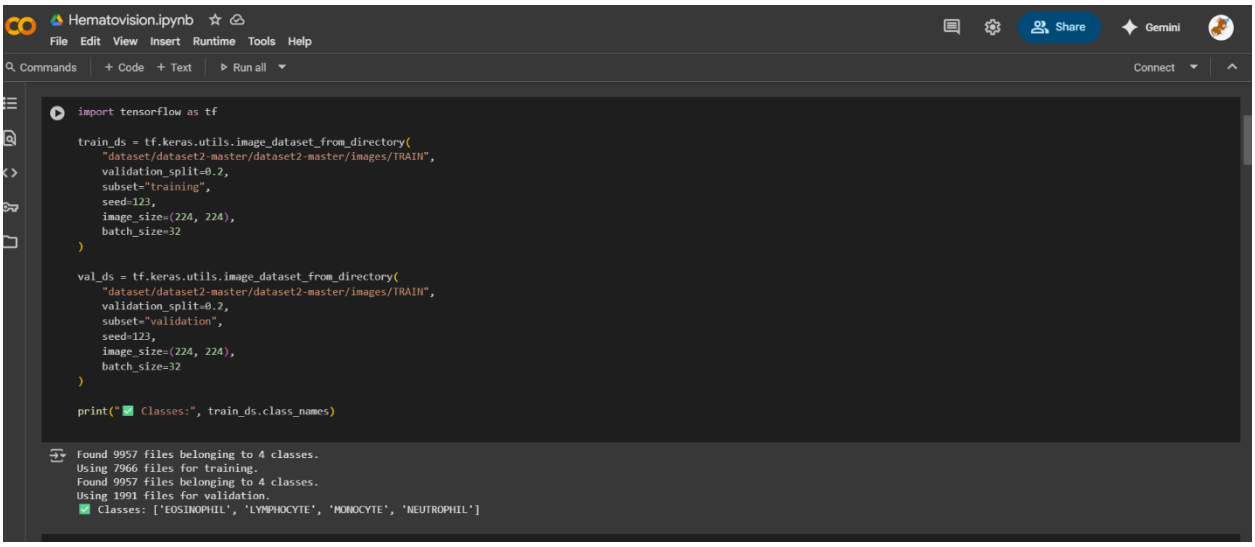
11. Assuming the team's velocity is 20 (points per sprint), the team's average velocity (AV) per iteration unit (story points per day) is calculated as follows:
    Average Velocity (AV) = Total Story Points / Sprint Duration (in days)
    AV = 20 / 10 = 2 points per day.

12. Burndown Chart

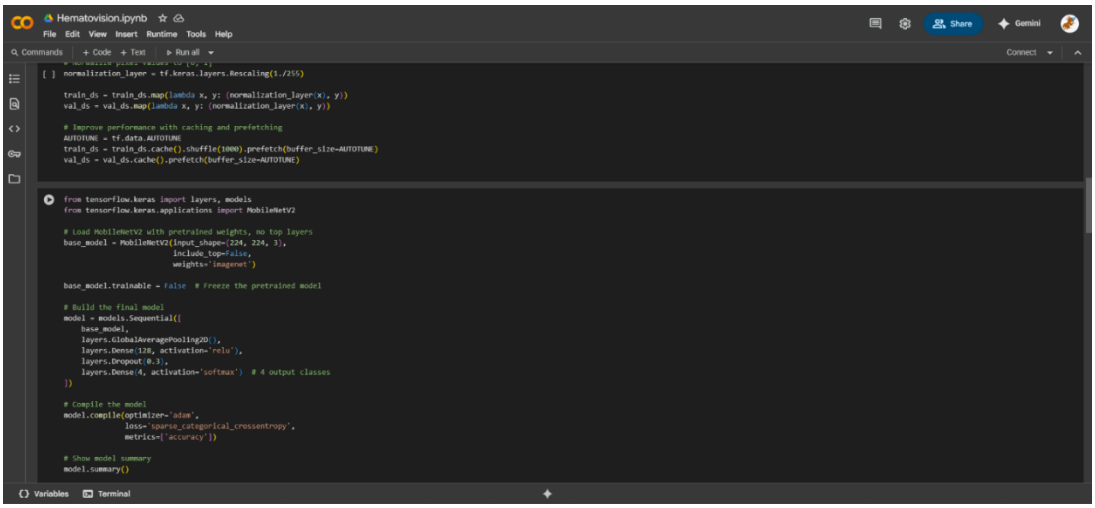
13. A burndown chart is a graphical representation of work left to do versus time. It is used to track progress over time in an agile project, showing how much work remains to be done for each sprint.
    The chart helps visualize the rate of progress, identify any issues with the project timeline, and ensure that work is completed on time.

6. FUNCTIONAL AND PERFORMANCE TESTING & RESULTS (output screen shots)

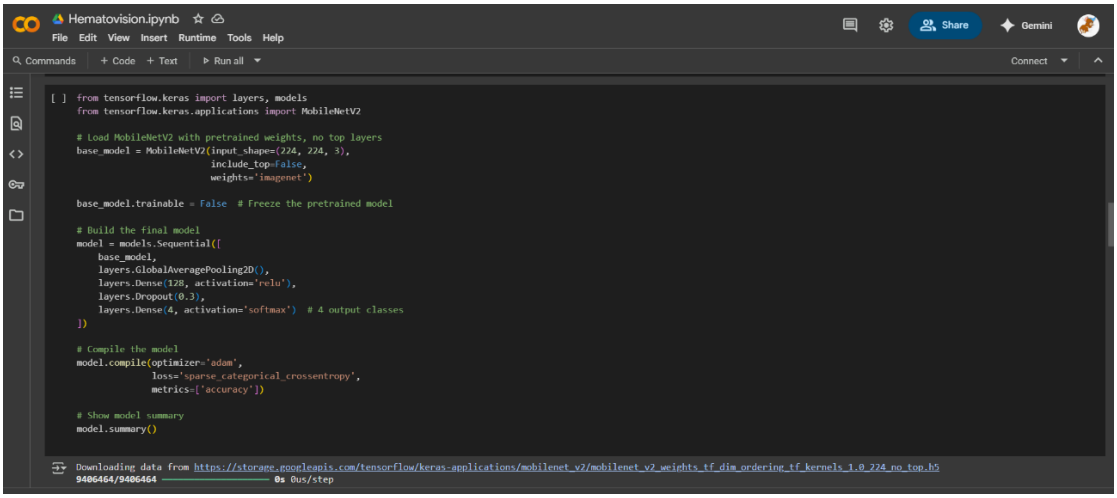
1.Data rendered



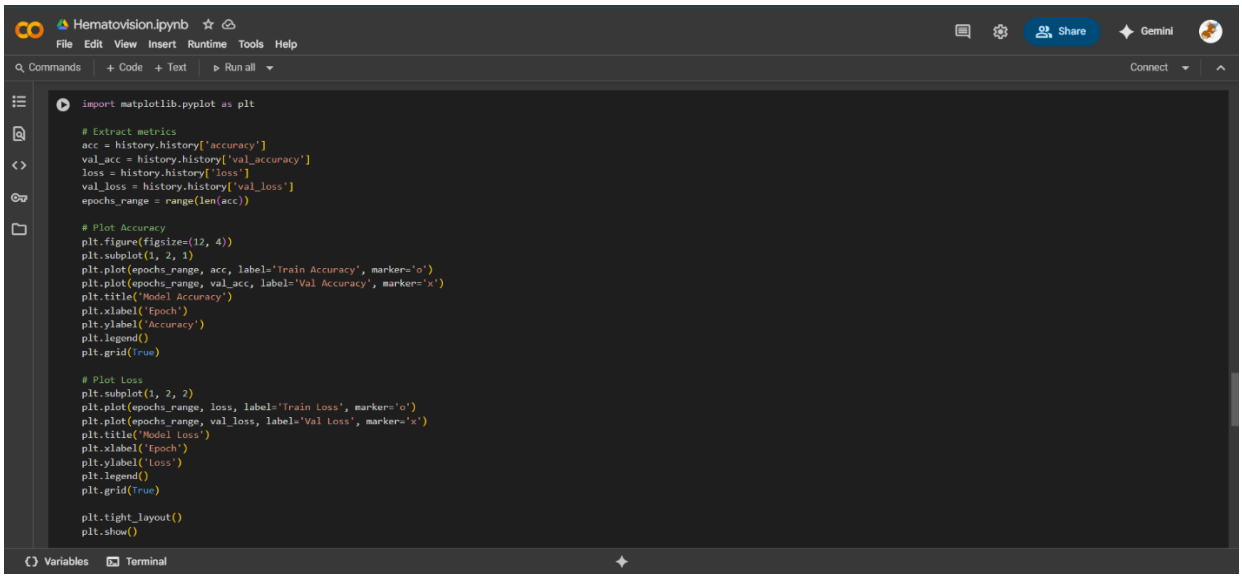
2.Data processing



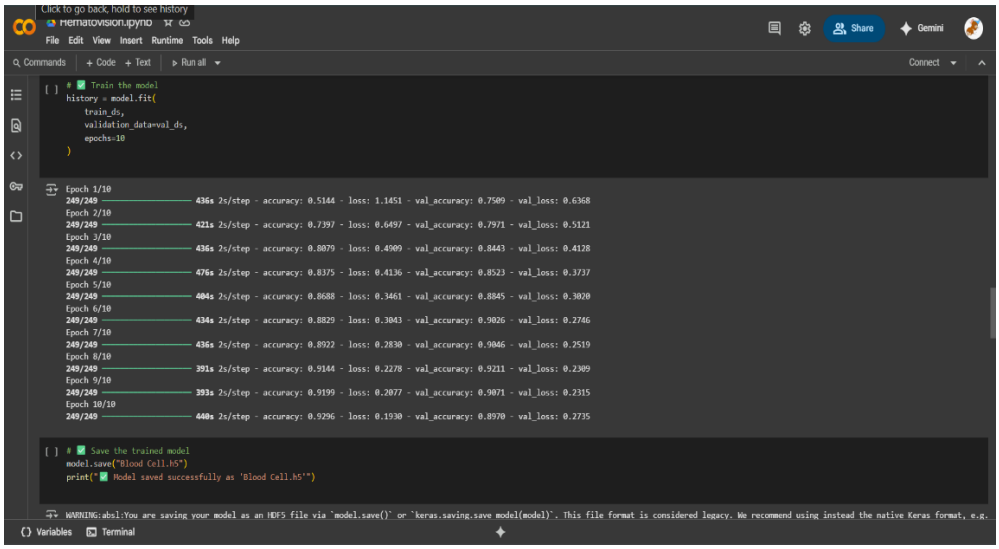
3. Utilization of Filters



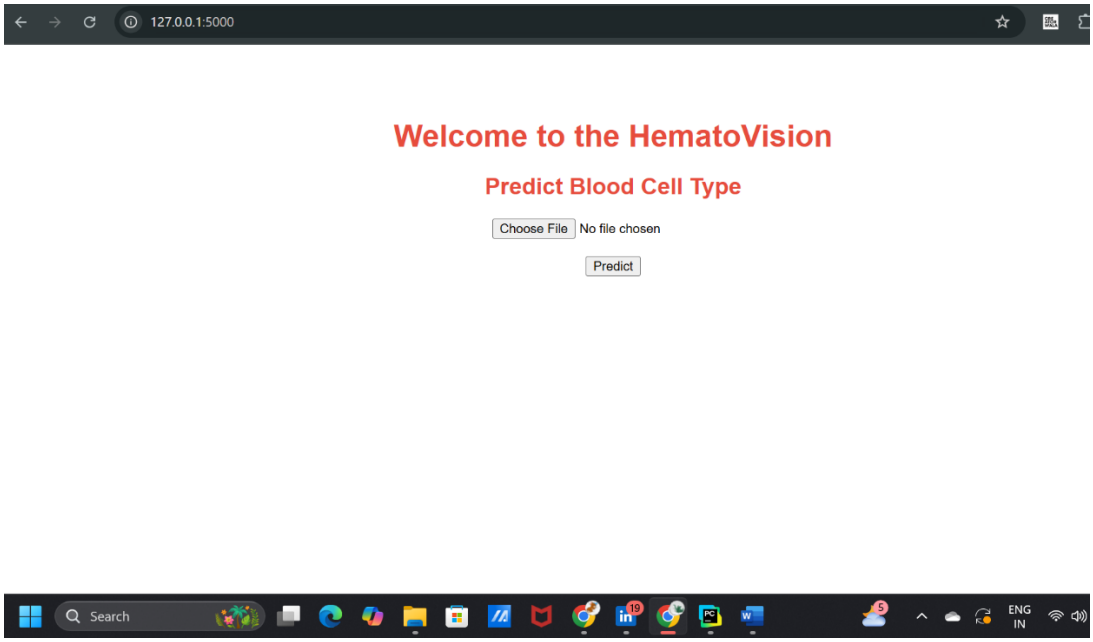
4. Calculation Fields Used



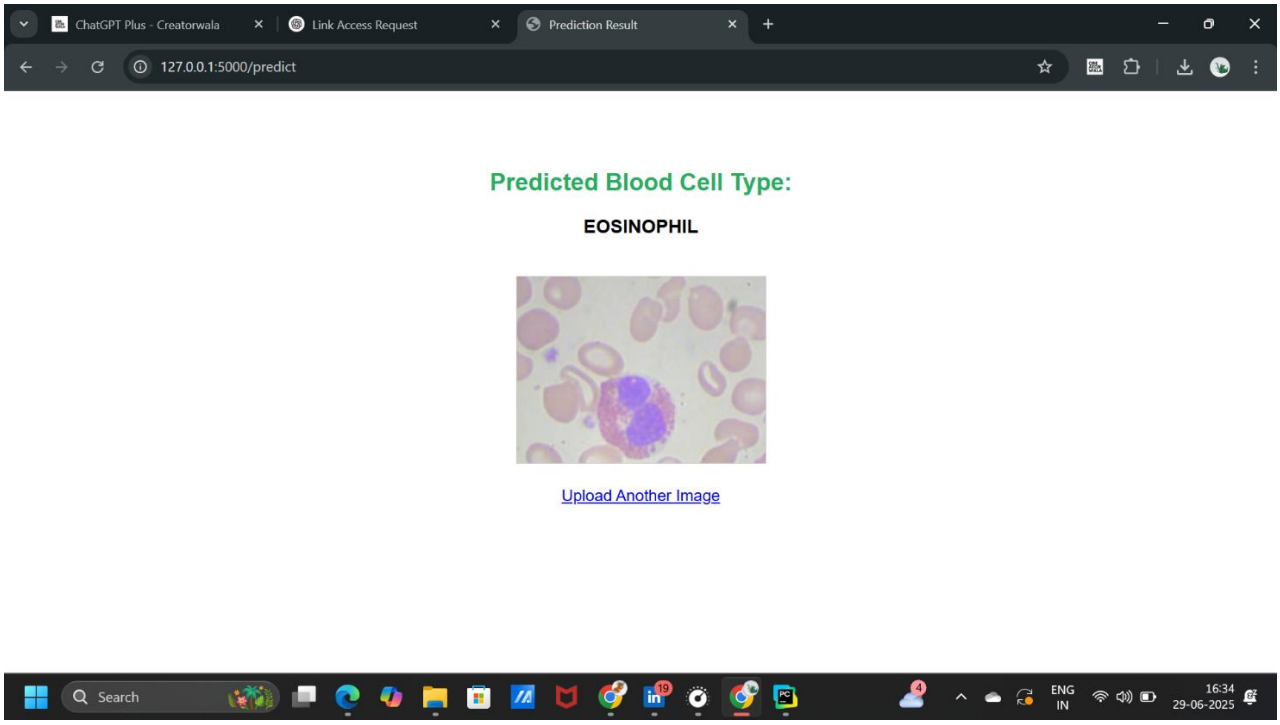
5. Model Evaluation



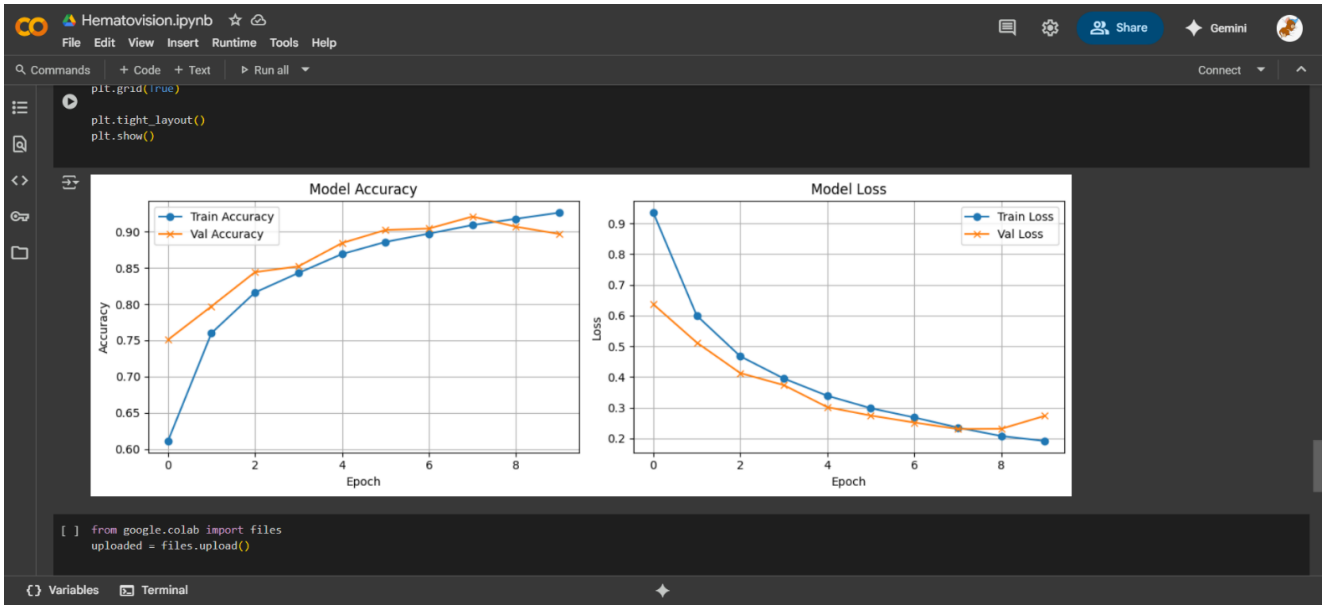
6. Interface



7. Dashboard Design and output



7. Visualizations



8. ADVANTAGES & DISADVANTAGES

✅ Advantages:

- Lightweight and Efficient: MobileNetV2 is optimized for performance on devices with limited computational power, making it ideal for rapid medical predictions.
- High Accuracy with Fewer Parameters: Delivers good classification performance without requiring heavy hardware.
- Transfer Learning Ready: The model can be quickly adapted with limited data and training time.
- Easy Integration: Compatible with Flask and web deployment for real-time predictions.

⚠️ Disadvantages:

- Limited Complexity Handling: MobileNetV2 may underperform compared to heavier models when handling high-resolution or highly complex images.
- Not Specialized for Medical Imaging: It is a general-purpose image classifier; may not capture domain-specific features unless further fine-tuned.
- Requires Careful Preprocessing: Model performance is sensitive to how images are resized, normalized, or augmented.

9. CONCLUSION

The HematoVision project successfully demonstrates the application of deep learning and transfer learning in the medical domain, specifically in automating blood cell classification. By using MobileNetV2, the system can accurately categorize blood cells into types such as Neutrophils, Eosinophils, Monocytes, and Lymphocytes. The integration of this model into a Flask-based web interface allows for seamless interaction with healthcare personnel, enabling fast and accessible diagnostic support. Overall, the project highlights how AI can assist medical professionals by reducing workload, improving accuracy, and accelerating diagnostics**.**

10. FUTURE SCOPE

- **Support for More Cell Types:** Future enhancements may include the ability to classify additional blood cells like Basophils, abnormal cells (e.g., blasts), or red blood cells.
- **Real-World Deployment:** The system can be integrated into hospital lab workflows with microscope-camera setups for automatic analysis.
- **Mobile & Cloud Deployment:** Hosting the system on cloud platforms or mobile devices to support diagnostics in rural or under-resourced settings.
- **Explainability:** Adding explainable AI (XAI) features to show why a prediction was made — important in healthcare applications.
- **Continual Learning:** Enabling the model to update itself with new data over time, improving accuracy and adaptability.

11. APPENDIX

Github repository: https://github.com/Dvsdeepak96/Hematovision