PROJECT

**Project Title:**

**HematoVision: Advanced Blood Cell Classification Using Transfer Learning**

# Team Id :LTVIP2025TMID43272

**Team Member: Gurrala Chinnari**

Mail Id: [gurralachinnari21@gmail.com](mailto:gurralachinnari21@gmail.com)

**Submitted By:**

• Gurrala Chinnari

• Swarnandhra college of engineering and technology

• Artificial intelligence and machine learning

• Roll No: 22A21A6127

**Submitted To:**

SmartBridge

# Phase-1 : Brainstorming & Ideation

## Objective:

To develop an AI-powered blood cell classification system using transfer learning that enhances diagnostic accuracy, reduces manual analysis time, and supports healthcare applications such as automated diagnostics, remote consultations, and medical

education.

## Key Points:

### Problem Statement:

* + Microscopic analysis of blood cells plays a critical role in diagnosing conditions such as leukemia, anemia, and infections. However, manual analysis

is time-consuming and subject to human error.

* + HematoVision utilizes transfer learning to classify blood cell types from microscopic images accurately. By integrating Al into hematovision system

enhances diagnostic speed, precision, and reliability, supporting early disease detection and improves patient outcomes.

### Proposed Solution:

* + AnAI-powered application that uses transfer learning to accurately classify blood cells from microscope images in real-time.
  + Ithelps doctors by providing quick, reliable results, reducing manual effort andimproving the speed and accuracy of diagnosis.

### TargetUsers:

* + Pathologists and lab technicians who need quick and accurate blood cell analysis.
  + Hospitals and diagnostic centers aiming to automate blood smear classification.
  + Doctors offering remote consultations who require fast diagnostic support.
  + Medical students and trainees learning about blood cell identification.

### Expected Outcome:

* + A functional AI-powered application that accurately classifies blood cells from images using transfer learning.
  + Provides fast, reliable diagnostic support to improve efficiency in medical analysis and education.

# Phase-2: Requirement Analysis

## Objective:

Define the technical and functional requirements for the HematoVision application.

## Key Points:

### Technical Requirements:

#### ProgrammingLanguage:Python

* + **Python Packages:**

NumPy, Pandas, Scikit-learn, Matplotlib, SciPy,

Seaborn, TensorFlow, Flask

#### Frameworks:

Flask for web integration, TensorFlow for deep learning

* + **Pre-trained Model:** VGG16 (used for transfer learning)

#### DevelopmentTools:

Command Line (pip install)

### Functional Requirements:

* + Ability to **upload microscopic blood cell images** through the web interface.
  + Classify blood cells into types like **eosinophils, lymphocytes, monocytes, and neutrophils** using a trained model.
  + Display **classification results** along with prediction confidence.
  + Provide an easy-to-use interface for doctors, students, and lab

technicians.

### Constraints & Challenges:

* + Handling imbalanced data across different blood cell classes.
  + Managing low-quality or blurry microscope images that affect prediction

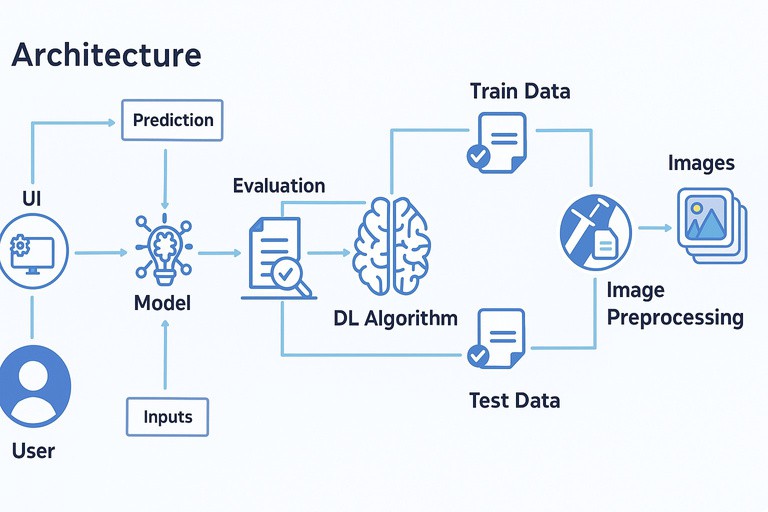
accuracy.

* + Optimizing model size and performance for faster processing and easy deployment.
  + Ensuring the web interface is responsive and user-friendly on all devices.

# Phase-3: Project Design

## Objective:

Develop the architecture and user flow of the application.



## Key Points:

### System Architecture:

* + The user uploads a blood cell image through the web interface (UI).
  + The image is sent as input to the trained deep learning model.
  + The model uses a pre-trained CNN (like VGG16) to process the image

and predict the blood cell type.

* + The prediction is evaluated and displayed back on the UI with confidence levels.
  + Initially, the model is built using preprocessed images from the dataset, split into training and testing sets.
  + The training data is used to fine-tune the DL algorithm using transfer learning, while test data is used for evaluation.

### User Flow:

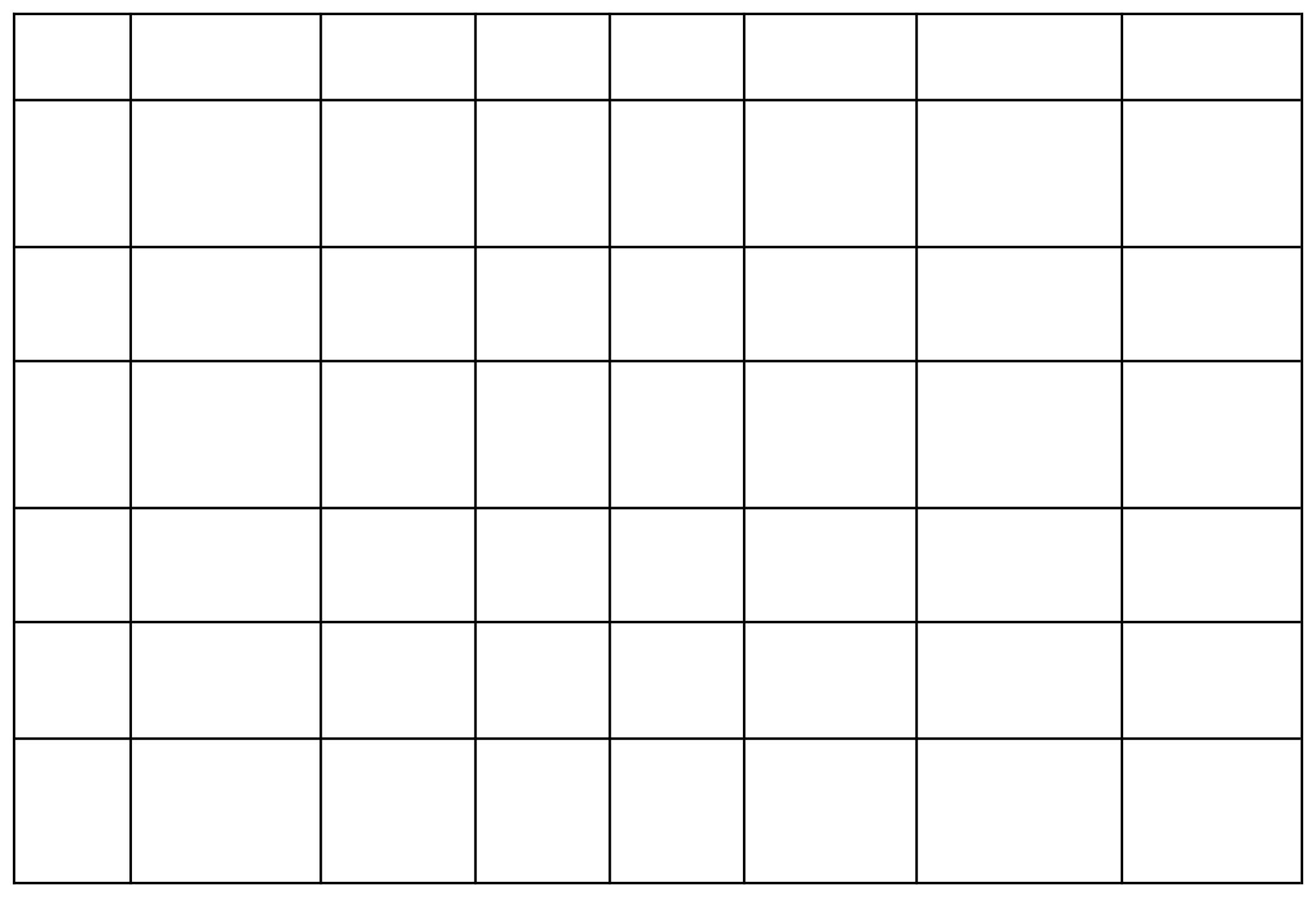
* + Step 1: User opens the web application. ● Step 2: Uploads a microscope image of a blood smear. ● Step 3: The model processes the image and classifies the cell type. ● Step 4: The predicted cell type and confidence score are displayed. ● Step 5:User can repeat with other images or use the results for diagnostic

insight.

# Phase-4: Project Planning

## Objective:

Breakdown development tasks for efficient completion.



**Sprint Task**

**Priority**

**Duration Deadline Assigned to Dependencies**

Sprint 1

Environment

Setup & Package Installation

Dataset

Collection & Preprocessing

🔴 High

3hours

Day 1

Member 1

Anaconda ,

Python

**Expected**

**outcome** Project environment ready

Sprint 1

Dataset access

🔴 High

4hours

Day 1

Member 2

Sprint 2

Model

Building using Transfer Learning

Flask Web

App Integration

Day 2

Preprocessed

Clean,prepar

ed image dataset

Trained

Sprint 2

Sprint 3

Testing &

Debugging

Medium

Sprint 3

Final

Presentation & Deployment

🟢 Low

1hour

End of

Day 2

Entire Team Working

application

Project

deployed and

demo-ready

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 🔴 High | 5hours |  | Member 3 | data, TensorFlow | classification model |
|  |  | Day 2 | Member | Trained | Working web |
| 🟡 Medium | 3hours |  | 1 & 4 | Model,Flask | interface |
|  |  |  |  | installed |  |
| 🟡 | 2 hours | Day 2 | Member 2 & 3 | Complete  System | Bug-free and responsive |
|  |  |  |  |  | system |

## Sprint Planning with Priorities

### Sprint 1 - Setup & Preparation (Day 1)

🔴

#### High Priority

* Set up the environment using Anaconda Navigator.
* Install all required Python packages (TensorFlow, Flask, etc.).
* Collect and preprocess the blood cell image dataset.

### Sprint 2 – Model Development & Integration (Day 2)

🔴

#### High Priority

* Build and train the blood cell classification model using transfer learning (e.g., VGG16).
* Integrate the trained model with the Flask web application.

### Sprint 3 – Testing, Deployment & Submission (Day 2)

🟡

#### Medium Priority

* Test the app functionality, fix bugs, and improve UI responsiveness.

#### 🟢 Low Priority

* Finalize deployment and prepare presentation/demo materials.

# Phase-5: Project Development

## Objective:

Implement the core features of the HematoVision application using transfer learning for blood cell classification.

## Key Points:

### Technology Stack Used:

* + **Frontend:** HTML (via Flask templates)
  + **Backend:** Flask Framework
  + **Deep Learning:** TensorFlow with pre-trained VGG16 model
  + **Programming Language:** Python

### Development Process:

* + Built and trained a blood **cell classification model using transfer learning** (VGG16).
  + Preprocessed a dataset of **12,000 labeled blood cell** images.
  + Integrated the trained model into a **Flask web application**.
  + Developed a user interface to upload images and display classification

**results with prediction confidence**.

### Challenges & Fixes:

* + **Challenge:** Model overfitting on certain blood cell types.

**Fix:** Used data **augmentation and dropout** regularization.

* + **Challenge:** Large model size caused slow predictions.

**Fix:** Applied model optimization and saved in**.h5** format for faster loading.

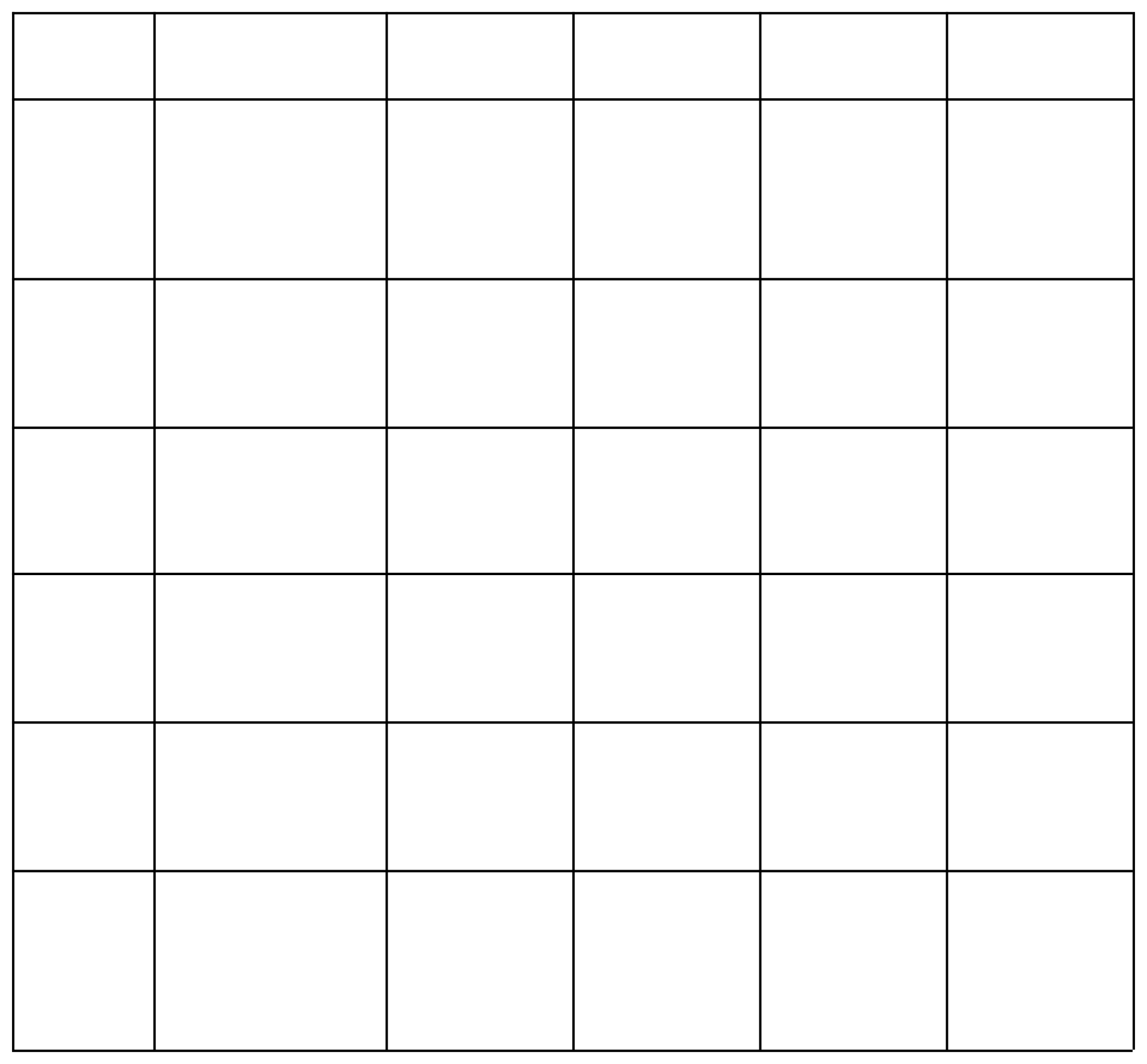
* + **Challenge:** Image quality variation.

**Fix:** Added preprocessing steps like resizing and normalization before prediction.

# Phase-6: Functional & Performance

## Objective:

Ensure that the HematoVision application performs accurately, reliably, and consistently across various test cases and environments.



**Test Category**

**Case ID**

TC-001

**Test**

**Scenario** Upload image of eosinophils

**Expected**

**Outcome** Correct cell type identified with confidence score

**Status**

**Tester**

Functional

Testing

✅ Passed

Tester 1

TC-002

Functional

Testing

Upload mixed Multi-class

✅ Passed

Tester 2

WBC/RBC

image

classification

displayed correctly

TC-003

Performance

Testing

Check model Results

**⚠**Needs

Optimization

Tester 3

response

time

TC-004

Bug Fix

Validation

Image with

poor lighting

displayed

under 2 secods

System still

makes reasonable prediction

Layout

adjusts properly on mobile

App loads

and predicts successfully online

✅ Fixed

Developer

TC-005

UI

Responsiveness

Test on

mobile browser

❌ Failed

Tester 2

TC-006

Deployment

Testing

Hosted on

local server and accessed remotely

🚀 Deployed

DevOps