**Project Design Phase**

**Solution Architecture**

|  |  |
| --- | --- |
| Date | 15 February 2025 |
| Team ID | LTVIP2025TMID33915 |
| Project Name | Hemotovision |
| Maximum Marks | 4 Marks |

**🏗️ Solution Architecture – HematoVision**

**🎯 Objective of the Solution Architecture**

The goal of HematoVision’s architecture is to bridge the gap between the **healthcare diagnostic problem** (manual blood smear analysis) and a reliable, accessible **technology solution** using deep learning. It defines the structure, behavior, and flow of information from image input to final prediction, ensuring scalability, simplicity, and usability in clinical and educational settings.

**🧱 Key Components**

1. **Frontend (User Interface)**
   * Built using HTML/CSS/Bootstrap
   * Allows image upload and displays classification results
   * Responsive design (mobile and desktop support)
2. **Backend (Web Server + Business Logic)**
   * Developed using Flask (Python-based microframework)
   * Manages routing, file handling, and model inference calls
3. **Model (AI Engine)**
   * Pre-trained ResNet50 CNN fine-tuned for 4-class blood cell classification
   * Implemented using TensorFlow/Keras
   * Loaded as a .h5 model and invoked via Flask APIs
4. **Data Flow**
   * User uploads an image via browser
   * Flask routes the image to the prediction engine
   * The model processes the image and returns class label + confidence score
   * Result is displayed on-screen with explanation
5. **Storage**
   * For basic setup: no persistent storage (session-based)
   * For advanced setups: optional CSV logging of predictions or integration with SQLite/cloud database
6. **Deployment Environment**
   * Can run locally or be deployed on cloud (AWS EC2, Google Cloud, Heroku, etc.)
   * Lightweight enough for offline use in rural settings

A diagram of a software algorithm

AI-generated content may be incorrect.

**🔧 Features Defined by Architecture**

* Image preprocessing
* Model loading and optimization
* API-based prediction
* UI result display
* Error handling and validation
* Multi-platform support

**🚀 Benefits of the Architecture**

* **Modular** and easy to expand
* **Lightweight**: Runs even on basic systems
* **Flexible**: Can scale up to cloud, mobile apps, or offline desktop tools
* **Secure**: Local processing ensures patient data privacy
* **Customizable**: Easily extendable to other diagnostic image types

**🔗 Reference**

* Inspired by:  
  AWS AI Architecture Example – [Voice Applications in Clinical Research](https://aws.amazon.com/blogs/industries/voice-applications-in-clinical-research-powered-by-ai-on-aws-part-1-architecture-and-design-considerations/)