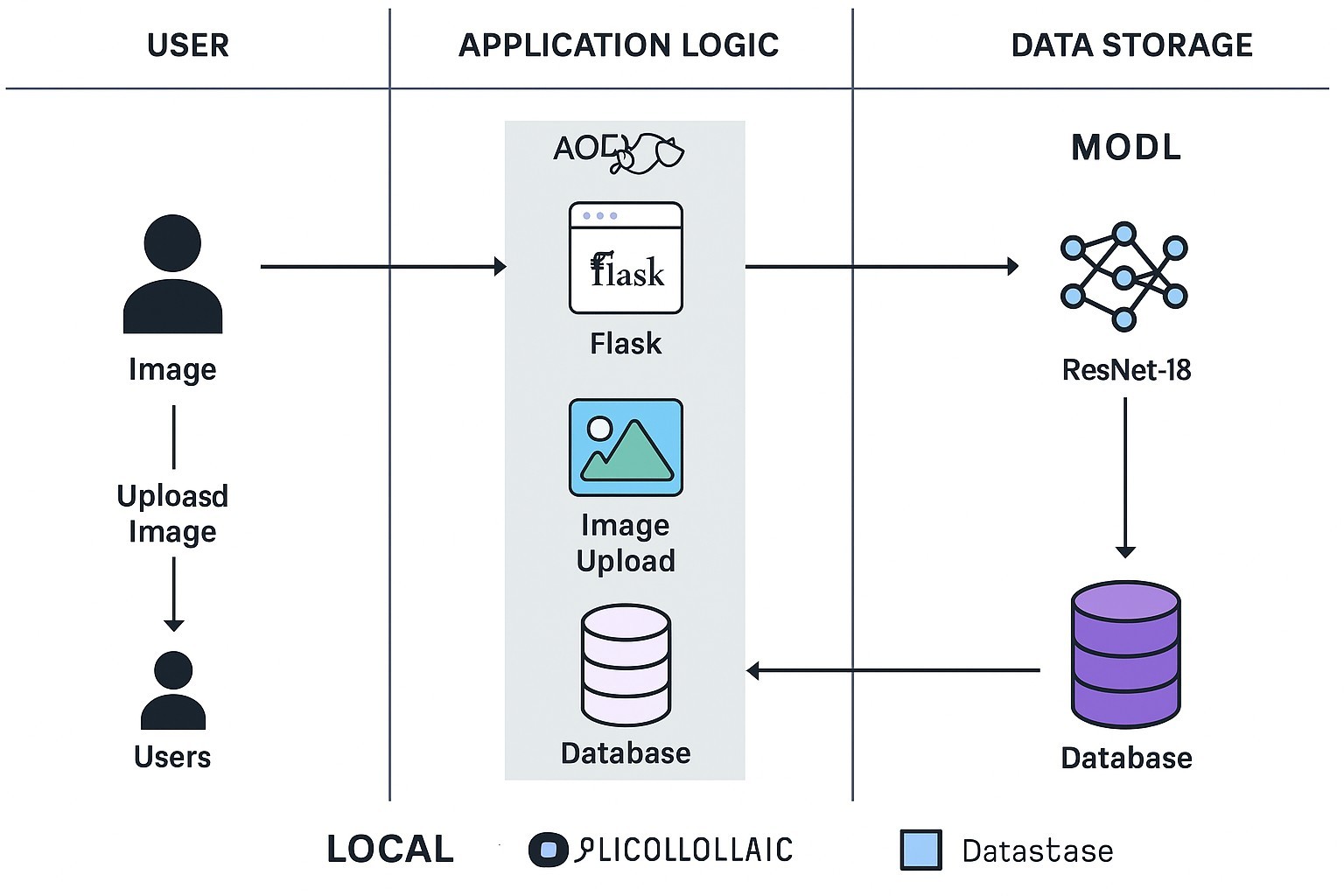
**Project Design Phase-II Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 26 June 2025 |
| Team ID | LTVIP2025TMID39268 |
| Project Name | Hematovision – Advanced Blood Cell  Classification using Transfer Learning |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2

**/**

MobileNetV2

**Table-1 : Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | Web interface where users upload Blood Cell images | HTML, CSS, JavaScript |
| 2. | Application Logic-1 | Backend logic for handling image upload and routing | Python(Flask Framework) |
| 3. | Application Logic-2 | N/A (Not using speech-to-text) | - |
| 4. | Application Logic-3 | N/A (Not using speech-to-text) | - |
| 5. | Database | Store image metadata and prediction results | SQLite (local), can scale to PostgreSQL |
| 6. | Cloud Database | Not used in local setup (future scope: Firebase /  AWS DynamoDB) | - |
| 7. | File Storage | Store uploaded images locally | Local filesystem (uploads/) |
| 8. | External API-1 | Optional integration for weather-related poultry insights | OpenWeatherMap API (Future Scope) |
| 9. | External API-2 | Not applicable | - |
| 10. | Machine Learning Model | Image classification using pretrained MobileNetV2 model | PyTorch + ResNet-18 |
| 11. | Infrastructure (Server / Cloud) | Application hosted and tested locally using Flask server | Localhost (127.0.0.1:5000). |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | Backend and model based on open-source tools | Flask, PyTorch, NumPy, OpenCV |
| 2. | Security Implementations | Validating file uploads, restricting extensions, local session handling | File type validation, Flask-CORS  (optional) |
| 3. | Scalable Architecture | Modular logic allows upgrade to microservices and cloud deployment | 3-tier architecture (UI - Logic - Storage) |
| 4. | Availability | Local setup for now; can be made highly available using containerization | Docker (future), Flask |
| 5. | Performance | Lightweight model (ResNet-18), fast image inference (<3s per request) | PyTorch with GPU/CPU acceleration |