**Project Design Phase-II**

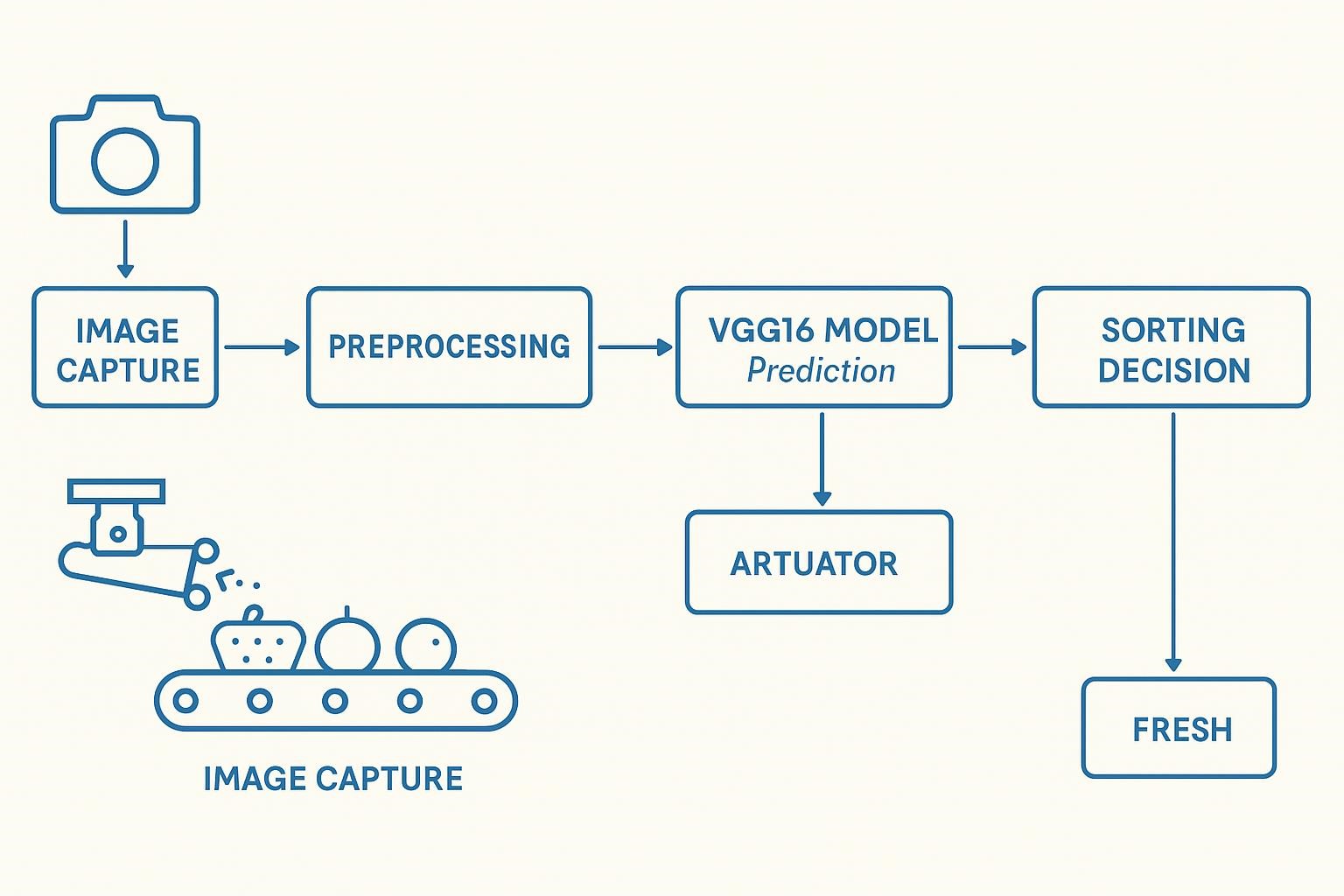
**Technology Stack (Architecture & Stack)**

|  |  |
| --- | --- |
| Date | 26 June 2025 |
| Team ID | LTVIP2025TMID41465 |
| Project Name | Smart Sorting:Transfer Learning for Identifying rotten fruits and vegetables |
| Maximum Marks | 4 Marks |

**Technical Architecture:**

A camera-equipped conveyor belt captures images of fruits/vegetables. These images are sent to an AI module using a **VGG16**-based model (transfer learning). The model identifies rotten produce. Based on predictions, sorting actuators separate fresh from rotten produce. The application is hosted locally or optionally on cloud (e.g., Google Colab or AWS).

**EXAMPLE:Smart Sorting:Transfer Learning for Identifying rotten fruits and vegetables**



**Table1:Components & Technologies:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Component** | **Description** | **Technology** |
| 1. | User Interface | Interface to upload/test images, optional dashboard | Streamlit (Python), HTML/CSS (for  Custom styling) |
| 2. | Application Logic-1 | Image acquisition from camera/conveyor input | OpenCV, Python |
| 3. | Application Logic-2 | Preprocessing and augmentation of images | Pillow (PIL), TensorFlow Keras  preprocessing |
| 4. | Application Logic-3 | Transfer learning model training & prediction | MobileNetV2 with TensorFlow/Keras |
| 5. | Database | Local dataset of fruit/vegetable images | Local filesystem (Structured directories) |
| 6. | Cloud Database | Optional dataset backup/storage | Google Drive, AWS S3 |
| 7. | File Storage | Stores input/output images and model files locally | Local filesystem |
| 8. | External API-1 | Optional integration with IoT sensors | REST API (optional, if sensors included) |
| 9. | External API-2 | Optional cloud ML API (if used for performance) | Google Cloud Vision API (optional) |
| 10. | Machine Learning Model | Detect rotten fruits using image classification | Transfer Learning using MobileNetV2  (TensorFlow/Keras) |
| 11. | Infrastructure | Transfer Learning using VGG16 (Keras, TensorFlow) | Local Machine, Render.com, AWS EC2,  Streamlit Cloud |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | Frameworks for ML and image processing | TensorFlow, Keras, Streamlit, pillow |
| 2. | Security Implementations | Not sensitive; limited access locally or via secured cloud | Limited local/cloud access |
| 3. | Scalable Architecture | Modular design; can be expanded with REST APIs & cloud | Cloud databases |
| 4. | Availability | High if hosted on cloud; manual mode otherwise | AWS, Render, Streamlit Cloud for uptime |
| 5. | Performance | Optimized using pretrained VGG16, image augmentation | TensorFlow, Keras |