**Solution Requirements**

In any machine learning-based image classification project — especially one like *Smart Sorting: Detecting Rotten Fruits with Transfer Learning* — defining solution requirements is essential. These requirements ensure that the final system delivers accurate, fast, and user-friendly predictions while remaining scalable and maintainable in real-world environments like farms, markets, or warehouses.

**Functional Requirements**

These specify the core features and behaviors the system must support to accomplish its objectives of identifying fresh vs. rotten fruits using a trained model and providing users with easy access to results.

| **FR No.** | **Functional Requirement (Epic)** | **Sub Requirement (Story / Sub-Task)** |
| --- | --- | --- |
| FR-1 | Image Upload and Handling | - Allow users to upload or capture images through a web interface.- Validate input file types (e.g., JPG, PNG). |
| FR-2 | Image Preprocessing | - Resize images to match input shape of model (e.g., 224x224).- Normalize pixel values. |
| FR-3 | Transfer Learning Model Integration | - Load and run predictions using a pre-trained CNN model.- Output label (Fresh/Rotten) with confidence score. |
| FR-4 | Result Visualization | - Display results with color-coded feedback (e.g., green for fresh, red for rotten).- Show prediction percentage. |
| FR-5 | Dataset Management | - Organize training data into labeled folders (Fresh, Rotten).- Allow model retraining with updated data. |
| FR-6 | User Interface | - Design a clean, responsive UI for farmers/vendors.- Show image preview and prediction result in real time. |
| FR-7 | Performance Logging *(optional)* | - Log user inputs and model outputs for future analysis.- Store logs in a separate prediction folder or CSV file. |
| FR-8 | Admin Model Update *(optional)* | - Enable admins to upload new training data and retrain model if needed. |
| FR-9 | Error Handling | - Handle cases where image is blurry, corrupted, or non-fruit.- Provide user-friendly error messages. |
| FR-10 | Mobile Compatibility *(optional)* | - Ensure web UI works well on mobile devices for use in agricultural fields. |

These functional requirements guide the complete machine learning pipeline—from input image handling to real-time output—ensuring a fast, intuitive, and meaningful experience for users.

**Non-Functional Requirements**

These define system quality attributes like performance, security, scalability, and user experience expectations.

| **NFR No.** | **Non-Functional Requirement** | **Description** |
| --- | --- | --- |
| NFR-1 | Usability | The prediction interface must be intuitive, allowing non-technical users (e.g., farmers or vendors) to operate easily. |
| NFR-2 | Accuracy | The model must achieve at least 90% validation accuracy before deployment to ensure practical reliability. |
| NFR-3 | Performance | Predictions should be generated in less than 2 seconds after image upload. |
| NFR-4 | Security | Input data and model files must be protected from unauthorized access (even if Flask is not applied). |
| NFR-5 | Maintainability | New classes of fruits (e.g., Mango, Banana) should be easy to add to the dataset and retrain model as needed. |
| NFR-6 | Portability | The system should run smoothly on local machines or cloud servers (if future hosting is considered). |
| NFR-7 | Scalability | The model and interface should support multiple fruit types with minor modifications. |
| NFR-8 | Fault Tolerance | In case of invalid input or model failure, the system should return an error without crashing the web app. |

These non-functional requirements ensure that the system is robust, trustworthy, and ready for practical deployment in field conditions where time and usability are crucial.