

Solution Requirements: Smart Sorting

Date	19 January 2026
Team ID	LTVIP2026TMIDS83701
Project Name	Smart Sorting: Transfer learning for identifying Rotten Fruits and Vegetables
Maximum Marks	2 Marks

Introduction

T This document outlines the solution requirements for the **Smart Sorting Transfer Learning System**, an AI-powered platform for identifying fresh and rotten fruits and vegetables. It defines the functional and non-functional requirements necessary for the successful development and deployment of the system. The system leverages transfer learning and deep learning techniques to automate quality inspection, reduce food waste, and improve efficiency in agricultural supply chains, warehouses, and retail markets.

2. Functional Requirements

Functional requirements define what the system must do.

2.1 User Management

- **FR1.1 User Registration:** The system shall allow new users (e.g., quality inspectors, warehouse managers, retailers) to register using email and password.
 - **FR1.2 Social Media Registration:** The system shall allow users to register using Google or other supported authentication providers.
 - **FR1.3 User Login:** The system shall allow registered users to log in securely.
 - **FR1.4 Password Recovery:** The system shall provide a password recovery mechanism.
-

2.2 Image Upload and Management

FR2.1 Image Upload: The system shall allow users to upload fruit and vegetable images in supported formats (e.g., JPEG, PNG).

FR2.2 Image Validation: The system shall validate uploaded images based on file size, resolution, and format requirements.

FR2.3 Secure Storage: The system shall temporarily store uploaded images securely for processing.

FR2.4 Image Deletion: The system shall automatically delete images after processing and result delivery to ensure data privacy.

2.3 Freshness Classification (Transfer Learning-Based)

FR3.1 Image Preprocessing: The system shall preprocess images (e.g., resizing, normalization, augmentation if required) before feeding them into the model.

FR3.2 Fresh/Rotten Classification: The system shall classify fruits and vegetables into predefined categories such as:

Fresh

Rotten

(Optional future: specific spoilage types or multi-class produce categories)

FR3.3 Classification Confidence: The system shall provide a confidence score for each prediction.

FR3.4 Model Integration: The system shall integrate a pre-trained deep learning model (e.g., ResNet, MobileNet, EfficientNet) fine-tuned using transfer learning.

2.4 Result Presentation

FR4.1 Result Display: The system shall display classification results including freshness status and confidence score.

FR4.2 Visual Feedback: The system may highlight affected regions or display annotations indicating spoilage areas.

FR4.3 Historical Records (Future Scope): The system may allow users to view past inspection results and analytics reports.

3. Non-Functional Requirements

Non-functional requirements define how the system performs.

3.1 Performance

NFR1.1 Response Time: The system shall classify an image and display results within 3–8 seconds, depending on image size and server load.

NFR1.2 Scalability: The system shall support increasing numbers of concurrent users and image uploads.

NFR1.3 Throughput: The system shall process at least X images per minute (based on operational requirements).

3.2 Security

NFR2.1 Data Privacy: The system shall ensure the privacy of user data and uploaded images.

NFR2.2 Authentication: The system shall implement secure authentication mechanisms.

NFR2.3 Authorization: Users shall only access their own data.

NFR2.4 Data Encryption: All data transmission shall use secure protocols (e.g., HTTPS).

3.3 Usability

NFR3.1 User-Friendly Interface: The system shall provide an intuitive interface for image upload and result viewing.

NFR3.2 Accessibility: The system shall follow accessibility standards where applicable.

NFR3.3 Error Handling: The system shall provide clear and informative error messages.

3.4 Reliability

NFR4.1 Uptime: The system shall maintain 99.9% uptime.

NFR4.2 Data Integrity: The system shall ensure accuracy and integrity of predictions and stored data.

NFR4.3 Fault Tolerance: The system shall gracefully handle failures and unexpected errors.

3.5 Maintainability

NFR5.1 Modularity: The architecture shall be modular to allow easy updates of the ML model or frontend components.

NFR5.2 Code Quality: The codebase shall follow best coding practices and documentation standards.

NFR5.3 Testability: System components shall be testable for validation and regression testing.

3.6 Portability

NFR6.1 Platform Independence: The web application shall be accessible across multiple operating systems and browsers.

NFR6.2 Deployment Flexibility: The system shall support cloud and on-premise deployment options.

4. Technical Requirements

TR1.1 Programming Language: Python (Backend & Machine Learning).

TR1.2 Web Framework: Flask or Django.

TR1.3 Machine Learning Framework: TensorFlow / Keras / PyTorch (Transfer Learning-based CNN model).

TR1.4 Frontend Technologies: HTML, CSS, JavaScript (React optional).

TR1.5 Database (Future Scope): SQLite / PostgreSQL / MongoDB for user and inspection data.

TR1.6 Deployment Environment: Cloud platform (AWS, Google Cloud, Azure) for scalability and real-time processing.

This document outlines the solution requirements for the HematoVision project, an AI-powered system for advanced blood cell classification. It details the functional and nonfunctional requirements necessary for the successful development and deployment of the system, drawing upon insights from the project overview, model analysis, and user stories.

Functional Requirements

Functional requirements define what the system must do. For HematoVision, these include:

User Management

FR1.1 User Registration: The system shall allow new users to register an account using their email and password.

FR1.2 Social Media Registration: The system shall allow users to register using their Facebook or Gmail accounts.

FR1.3 User Login: The system shall allow registered users to log in using their credentials.

FR1.4 Password Recovery: The system shall provide a mechanism for users to recover forgotten passwords.

Image Upload and Management

FR2.1 Image Upload: The system shall allow users to upload blood cell images in supported formats (e.g., JPEG, PNG).

FR2.2 Image Validation: The system shall validate uploaded images to ensure they meet specified criteria (e.g., file size, format, resolution).

FR2.3 Secure Storage: The system shall securely store uploaded images temporarily for processing.

FR2.4 Image Deletion: The system shall automatically delete uploaded images after processing and result delivery.

Blood Cell Classification

FR3.1 Image Preprocessing: The system shall preprocess uploaded images (e.g., resizing, normalization) before feeding them to the classification model.

FR3.2 Cell Type Classification: The system shall classify blood cell images into predefined categories (e.g., Eosinophil, Lymphocyte, Monocyte, Neutrophil).

FR3.3 Classification Confidence: The system shall provide a confidence score for each classification result.

FR3.4 Model Integration: The system shall integrate a pre-trained machine learning model for blood cell classification.

Result Presentation

FR4.1 Result Display: The system shall display the classification results to the user, including the predicted cell type and confidence score.

FR4.2 Visual Feedback: The system shall provide visual feedback on the classified image, potentially highlighting the classified areas or displaying bounding boxes.

FR4.3 Historical Results (Future): The system may provide a feature to view past classification results.

Non-Functional Requirements

Non-functional requirements specify how the system performs a certain function. For HematoVision, these include:

Performance

NFR1.1 Response Time: The system shall classify an image and display results within 510 seconds, depending on image size and server load.

NFR1.2 Scalability: The system shall be scalable to handle an increasing number of concurrent users and image uploads.

NFR1.3 Throughput: The system shall be capable of processing at least X images per minute (where X is a defined metric based on expected usage).

Security

NFR2.1 Data Privacy: The system shall ensure the privacy and confidentiality of user data and uploaded images.

NFR2.2 Authentication: The system shall implement secure user authentication mechanisms.

NFR2.3 Authorization: The system shall ensure that users can only access their own data and functionalities.

NFR2.4 Data Encryption: All data transmitted between the client and server shall be encrypted using industry-standard protocols (e.g., HTTPS).

Usability

NFR3.1 User-Friendly Interface: The system shall provide an intuitive and easy-to-use interface for image upload and result viewing.

NFR3.2 Accessibility: The system shall adhere to accessibility guidelines to ensure usability for users with disabilities.

NFR3.3 Error Handling: The system shall provide clear and informative error messages to users.

Reliability

NFR4.1 Uptime: The system shall maintain an uptime of 99.9%.

NFR4.2 Data Integrity: The system shall ensure the integrity of all processed data and classification results.

NFR4.3 Fault Tolerance: The system shall be designed to gracefully handle failures and recover from unexpected errors.

Maintainability

NFR5.1 Modularity: The system architecture shall be modular to facilitate easy updates and maintenance of individual components.

NFR5.2 Code Quality: The codebase shall adhere to high coding standards, be well documented, and easily understandable.

NFR5.3 Testability: The system components shall be testable to ensure proper functionality and facilitate regression testing.

Portability

NFR6.1 Platform Independence: The web application shall be accessible from various operating systems and web browsers.

NFR6.2 Technology Stack: The system shall be built using widely supported and opensource technologies to ensure future compatibility and ease of deployment.

Technical Requirements

Technical requirements specify the technology stack and infrastructure needed.

TR1.1 Programming Language: Python (for backend and machine learning).

TR1.2 Web Framework: Flask.

TR1.3 Machine Learning Framework: TensorFlow/Keras.

TR1.4 Frontend Technologies: HTML, CSS, JavaScript.

TR1.5 Database (Optional for future): SQLite or PostgreSQL for user accounts and historical data.

TR1.6 Deployment Environment: Cloud-based platform (e.g., AWS, Google Cloud, Azure) for scalability and accessibility.

Conclusion

This Solution Requirements document provides a comprehensive overview of the necessary functional, non-functional, and technical requirements for the Smart Sorting project. Adhering to these requirements will ensure the development of a robust, efficient, and userfriendly blood cell classification system that meets the needs of pathologists and healthcare professionals.