**User Acceptance Testing (UAT) Template**

|  |  |
| --- | --- |
| Date | 28 June 2025 |
| Team ID | LTVIP2025TMID35377 |
| Project Name | Smart Sorting Transfer Learning for Identifying Rotten Fruits and Vegetables |
| Maximum Marks |  |

**Project Overview:**

Project Name: Smart Sorting Transfer Learning for Identifying Rotten Fruits and Vegetables

Project Description: The project aims to develop an **intelligent image classification system** that can **automatically detect and sort rotten fruits and vegetables** using **Transfer Learning**, a powerful technique in deep learning that reuses a pre-trained model to solve a new problem with less data and training time.

Project Version: v1.1,v2.0,v3.0,v4

Testing Period: [10/06/2025] to [25/6/2025]

**Testing Scope:**

[List of Features and Functionalities to be Tested]

**🔹 1. Image Upload Functionality**

* Uploading different formats: JPG, PNG, JPEG
* Size validation (e.g., max 5MB)
* Error handling for unsupported or corrupted images

**🔹 2. Image Preprocessing**

* Image resizing to model’s input shape (e.g., 224×224)
* Color normalization and scaling
* Check for grayscale or noisy inputs

**🔹 3. Model Prediction (Core Functionality)**

* Correct classification into:
  + Fresh
  + Rotten
* Accuracy and confidence level display
* Handling poor quality images gracefully

**🔹 4. User Interface**

* Clean and simple UI layout
* Real-time loading indicators (optional)
* Clear result display after prediction
* Reset/clear button to upload new image

**🔹 5. Backend API Integration**

* API receiving image properly
* JSON response format from prediction
* Error handling if model/API fails

**🔹 6. Performance Testing**

* Response time under 3 seconds
* Ability to handle multiple requests (concurrent API calls)
* Consistent accuracy under load

**🔹 7. Functional Filters (if added)**

* Filtering results (e.g., only rotten images)
* Sort by confidence levels (optional)

**🔹 8. Fine-Tuning Validation (if applied)**

* Accuracy improvement after fine-tuning
* Overfitting/underfitting check
* Performance before vs after tuning

**🔹 9. Dashboard (Power BI or Excel, if used)**

* Data filters (e.g., date, freshness status)
* Correct graphs/charts: pie, bar, line
* DAX expressions (if any) working as intended

**🔹 10. Report Generation (Optional Feature)**

* Export result to PDF/CSV
* Include prediction time, confidence, and label

[List of User Stories or Requirements to be Tested]

**Project Title:** *Smart Sorting Transfer Learning for Identifying Rotten Fruits and Vegetables*  
**Version:** v1.0

**🧑‍🌾 User Role: General User (e.g., farmer, vendor, quality checker)**

**🔹 US-01: Upload Fruit/Vegetable Image**

**As a user**, I want to upload an image of a fruit or vegetable,  
**so that** the system can analyze and classify it.

* ✅ **Requirement to be Tested:** File format, size, upload success/failure

**🔹 US-02: Predict Fresh or Rotten Status**

**As a user**, I want the system to predict whether the item is fresh or rotten,  
**so that** I can sort or reject the bad ones.

* ✅ **Requirement to be Tested:** Accurate classification with clear result label and confidence

**🔹 US-03: See Prediction Confidence**

**As a user**, I want to see how confident the model is in its prediction,  
**so that** I can trust or double-check unclear results.

* ✅ **Requirement to be Tested:** Confidence percentage display (e.g., 96.8%)

**🔹 US-04: Use a Simple and Clean Interface**

**As a user**, I want an easy-to-use interface,  
**so that** I can quickly upload and view results without technical knowledge.

* ✅ **Requirement to be Tested:** UI layout, button visibility, responsiveness

**🔹 US-05: Re-upload or Check Another Image**

**As a user**, I want to upload another image after checking one,  
**so that** I can test multiple items without refreshing the page.

* ✅ **Requirement to be Tested:** Reset/Upload New Image button functionality

**🔹 US-06: Get Fast Results**

**As a user**, I want the system to process images quickly,  
**so that** I don't waste time waiting.

* ✅ **Requirement to be Tested:** Response time < 3 seconds

**🔹 US-07: See a Summary Dashboard (Optional/Bonus)**

**As a user**, I want to see a dashboard summarizing the results of multiple items,  
**so that** I can get an overview of the freshness status.

* ✅ **Requirement to be Tested:** Number of rotten vs fresh items, graphs, filters

**🔹 US-08: Mobile and Desktop Compatibility**

**As a user**, I want the platform to work on both mobile and desktop devices,  
**so that** I can use it from anywhere.

* ✅ **Requirement to be Tested:** Layout responsiveness across screen sizes

**🔹 US-09: Upload via Camera (Advanced/Future)**

**As a user**, I want to click an image using a webcam or mobile camera,  
**so that** I can test items live.

* ✅ **Requirement to be Tested:** Camera permission, live image capture (if implemented)

**Testing Environment:**

URL/Location: [Web URL or Application Location]

http://127.0.0.1:8000/

Credentials (if required): [Username/Password]

Jadi Lathish,J lathish@7

**Test Cases:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Test Steps** | **Expected Result** | **Actual Result** | **Pass/Fail** |
| TC-001 | |  | | --- | | Image  Upload &  Prediction  test |  |  | | --- | |  | | 1. Open app 2. Upload image of rotten apple 3. Click "Predict" | |  | | --- | | Output =  "Rotten" |  |  | | --- | |  | | |  | | --- | | Output =  "Rotten" |  |  | | --- | |  | | Pass |
| |  | | --- | | TC-002 |  |  | | --- | |  | | |  | | --- | | Fresh  Image  Prediction |  |  | | --- | |  | | 1. Open app 2. Upload image of fresh banana 3. Click "Predict" | |  | | --- | | Output =  "Fresh" |  |  | | --- | |  | | |  | | --- | | Output =  "Fresh" |  |  | | --- | |  | | Pass |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **TC-003** |  |  | | --- | |  | | |  | | --- | | **Invalid File Upload** |  |  | | --- | |  | | **1. Open app 2. Upload PDF file instead of image 3. Click "Predict"** | |  | | --- | | **Show error message** |  |  | | --- | |  | | |  | | --- | | Error message shown |  |  | | --- | |  | | |  | | --- | | **Pass** |  |  | | --- | |  | |
| |  | | --- | | **TC-004** |  |  | | --- | |  | | |  | | --- | | **Missing Input** |  |  | | --- | |  | | **1. Open app 2. Do not upload any image 3. Click "Predict"** | |  | | --- | | **Show validation error** |  |  | | --- | |  | | |  | | --- | | **Error popup shown** |  |  | | --- | |  | | **Pass** |
| |  | | --- | | **TC-005** |  |  | | --- | |  | | |  | | --- | | **Multiple Requests Handling** |  |  | | --- | |  | | **1. Upload 5 images one by one quickly 2. Check app responsiveness** | |  | | --- | | **App handles smoothly** |  |  | | --- | |  | | |  | | --- | | **All responded well** |  |  | | --- | |  | | **Pass** |

**Bug Tracking:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Bug ID** | **Bug Description** | **Steps to reproduce** | | **Severity** | **Status** | **Additional feedback** |
| BG-001 | Incorrect result for partially rotten fruit | 1. Upload image of slightly spoiled banana 2. Click "Predict" 3. Check output | | Medium | Open | Model gives “Fresh” though part is discolored |
| |  | | --- | | BG-002 |  |  | | --- | |  | | | |  | | --- | | UI  misalignment  on mobile  view |  |  | | --- | |  | | | 1. Open site on mobile 2. Upload image 3. Check buttons and image layout | Low | In progress | Adjust margins for better mobile responsiveness |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | BG-003 |  |  | | --- | |  | | |  | | --- | | File size >5MB causes crash |  |  | | --- | |  | | 1. Upload image >5MB 2. Click "Predict" | High | Closed | Fixed with size restriction + compression logic |

**Sign-off:**

Tester Name: [Jadi.Lathish,B pavan,s Sai,harsha]

Date: [29/06/2025]

Signature: [j lathish,pavan,sai,harsha]

**Notes:**

* Ensure that all test cases cover both positive and negative scenarios.
* Encourage testers to provide detailed feedback, including any suggestions for improvement.
* Bug tracking should include details such as severity, status, and steps to reproduce.
* Obtain sign-off from both the project manager and product owner before proceeding with deployment.