ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Test Plan, Test Design Specifications and Test Cases

Version 1

CENG 408

Innovative System Design and Development II

SUMMER TRAINING
INFORMATION SYSTEM

Nursena Bitirgen- 202011029
Tamer Memiş - 201911210
Furkan Yamaner - 202011211
Boran Gülbaşar - 202011033

Advisor: Gül TOKDEMİR & SEZER UĞUZ

Table of Contents

1. INTRODUCTION

1.1 Version Control

Version No	Description of Changes	Date
1.0	Initial test plan version	March 28, 2025

1.2 Overview

This document lays out our complete testing strategy for the iProvis app we're building, which helps users get detailed information and compare prices of products just by taking a photo. Our goal is to make sure every feature works smoothly, accurately, and efficiently. By following this test plan, we aim to ensure the app delivers a reliable and seamless experience to users

1.3 Scope

The test plan covers functional and non-functional aspects of the application, including user authentication, product identification using image recognition, retrieval of product details, and price comparison through web scraping. It also includes tests for performance, security, and usability. This document will provide detailed test design specifications and test cases for each major feature of the application.

1.4 Terminology

Acronym	Definition
TFLite	Lightweight ML model framework
MongoDB	Database service
UI/UX	User interface and experience
API	Application Programming Interface
LG	Login
PC	Price Comparasion
PI	Product Identification
PI.R	Product Information Retrieval
PERF/SEC	Performance/Security

2. FEATURES TO BE TESTED

This section outlines the key features of the application that will be tested to ensure functionality, performance, and user experience.

2.1 Login (LG)

The login feature allows users to create an account and sign in to access personalized features. While logging in is optional, registered users can save preferences and access additional functionalities.

2.2 Product Identification (PI)

The product identification feature enables users to take a photo of a supermarket product. The app processes the image using machine learning to recognize the product and fetch relevant details.

2.3 Price Comparison (PC)

Once a product is identified, the application retrieves its price from various online and offline stores using web scraping techniques. This feature ensures users can compare prices efficiently.

2.4 Product Information Retrieval (PI.R)

In addition to price comparison, the application provides detailed ingredient information, nutritional facts, and other relevant product details.

2.5 User Interface (UI)

The overall UI of the application will be tested for usability, responsiveness, and smooth navigation to ensure a seamless user experience.

2.6 Performance and Security (PERF/SEC)

Performance tests will assess the speed and reliability of image recognition, data retrieval, and price comparison features. Security tests will focus on user data protection and safe interactions with external sources.

3. FEATURES NOT TO BE TESTED

This section lists the features that will not be included in the current testing scope and provides reasons for their exclusion.

3.1 Third-Party Services We Rely On

We're assuming services like Firebase Auth, MongoDB or other third-party services will work as advertised. Why? We can't control whether Google's servers will crash, just like we can't test whether your phone's mobile data will work in the basement of a department store.

3.2 Real-Time Price Accuracy

We'll verify our scraping works, but we can't guarantee that Store X hasn't just changed prices 2 minutes ago due to economics in Turkey.

3.3 Phone's Camera Quality

Blurry potato-cam photos? We'll optimize our image processing, but we can't magically fix hardware limitations. If your lens is smudged with fingerprints, recognition might struggle.

4. ITEM PASS/FAIL CRITERIA

Expected Results:

1. The product should be recognized within 2 seconds.
2. Price comparisons from at least 3 different stores should be displayed.
3. Stock status ("In Stock" or "Out of Stock") must be shown.
4. Recycling information (e.g., "This packaging is recyclable") should appear.

Test Result Evaluation:

- If all expected results are met : **PASS**
- If the product is not recognized or price data is missing : **FAIL** (A defect is logged)

4.1 Exit Criteria

1. **Mandatory Criteria:**
   - 100% of critical (High Priority) tests must pass (e.g., product recognition, price comparison).
   - The system must support 100 concurrent users (performance test).
   - No API integration errors should occur.

2. **Ideal Criteria:**
   - 95% of all test cases should pass.
   - Average response time should not exceed 3 seconds.

4.1.1 Real-World Scenario Example

Test Steps:

1. A user takes a photo of a coffee package.
2. The app:
   - Recognizes the product in 1.5 seconds : PASS
   - Displays price comparisons from 5 stores : PASS
   - But recycling information is missing : FAIL (Defect logged)

Conclusion:

- Since a critical feature (recycling info) failed, the test is unsuccessful.
- The developer must fix the issue and retest.

Test Scenario	Status	Description
Product Recognition	PASS	Recognized in 1.5 seconds.
Price Comparison	PASS	5 stores listed.
Recycling Information	FAIL	Missing data. (Defect #123)

Decision:

- Due to a High Priority defect, the release is not approved.

4.1.2 Performance Test Criteria

- **Load Test:** With 100 users scanning products simultaneously, server response time should be <1 second**.**

- **Battery Test:** After 1 hour of use, battery consumption should be <10%**.**

5. REFERENCES

[1] iProViS-Intelligent-Product-Vision-System SRS Documentation 3.2 - Functional Requirements

[2] iProViS-Intelligent-Product-Vision-System SRS Documentation 3.2.2 Capture Product Photo for Information Retrieval Use Case

[3] iProViS-Intelligent-Product-Vision-System SRS Documentation 3.2.3 Price and Stock Availability Display Use Case

[4] iProViS-Intelligent-Product-Vision-System SRS Documentation 3.2.5 Recyclability Information Display Use Case

6. TEST DESIGN SPECIFICATIONS

This section defines the test approach for each major feature of the iProvis application, including subfeatures to be tested and high-level test cases. The design ensures comprehensive validation of functional and non-functional requirements.

6.1 Login (LG)

6.1.1 Subfeatures to be tested

6.1.1.1 User Authentication (LG.AUTH)

- Valid credentials verification via Firebase Auth.
- Session persistence after app restart.

6.1.1.2 Input Validation (LG.VAL)

- Email format verification (RFC 5322 compliance).
- Password strength enforcement (minimum 8 chars, special characters).

6.1.1.3 Error Handling (LG.ERR)

- Graceful handling of incorrect credentials.
- Network failure recovery (offline mode support).

6.1.1.4 Guest Mode (LG.GST)

- Unauthenticated access with restricted functionality.

6.1.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
LG.TC01	*LG.01, LG.02*	*H*	*Successful login with valid Firebase credentials redirects to home screen within ≤2s.*
LG.TC02	LG.01, LG.VAL	M	Login attempt with invalid email format triggers input validation error.
LG.TC03	LG.01, LG.ERR	H	Login fails with correct email + wrong password; verify error message.
LG.TC04	LG.01, LG.ERR	M	Network interruption during login shows offline error message
LG.TC05	LG.GST	L	Guest mode access grants limited functionality without database writes.

6.2 Add User (AU)

6.2.1 Subfeatures to be tested

6.2.1.1 Admin (AU.AD)

Validates administrative privileges for user creation, including role assignment (admin/guest) and database integrity checks. Tests cover duplicate prevention and permission validation.

6.2.1.2 Guest (AU.GT)

Ensures guest users can be added with restricted permissions, excluding access to admin-only functionalities like role modification or system settings.

6.2.2 Test Cases

Here list all the related test cases for this feature

TC ID	Requirements	Priority	Scenario Description
AU.AD.01	3.2	H	Add an admin user who is not already in db
AU.AD.02	3.3	H	Add an admin user who already exist
AU.AD.03	3.4	M	Validate admin role assignment in MongoDB
AU.GT.01	3.5	L	Add guest user with default permissions.

7. Detailed Test Cases

7.1 AU.AD.01

TC_ID	AU.UR.01
Purpose	Verify new user can register with valid credentials.
Requirements	FR-UR-01 (User Registration)
Priority	High.
Estimated Time Needed	3 Minutes
Dependency	Database connection available
Setup	Ensure test email not register in system.
Procedure	[A01] Navigate to "Sign Up" page.
	[A02] Enter test email.
	[A03] Click "Register".
	[V01] Verify success toast: "Account created successfully!"
	[V02] Check database (Firebase/MongoDB) for new user record.

7.2 LG.AD.02

TC_ID	AU.UR.02
Purpose	Prevent registration with existing email
Requirements	FR-UR-02
Priority	High.
Estimated Time Needed	2 Minutes
Dependency	Existing user in database
Setup	Ensure test email is in system.
Procedure	[A01] Attempt to add duplicate email.
	[V01] Verify error: "User already exists."