ÇANKAYA UNIVERSITY

Software Requirements Specification

iProViS: Intelligent Product Vision System

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

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to present the simulation known as **iProVis: Intelligent Product Vision System**. This artificial intelligence-based vision system is designed to provide users with the ability to easily identify the cheapest products and locate the nearest availability of these products. The document provides detailed information about the project requirements, addressing identified constraints, and the proposed software functionalities. It also outlines the software technologies utilized in the project's execution, highlighting the positive impacts these technologies have had on overall system performance. Additionally, this Software Requirements Specification (SRS) defines how participants interact with the model-based computer vision system.

1.2 Scope of the Project

Today, customers have numerous shopping options, including online and physical stores, each with various shopping methods. The rapid growth of online shopping has made life easier for customers, thanks to software solutions tailored for this space. However, this convenience has also led to challenges such as "getting away from the product itself," with numerous sellers offering the same product. As a result, while e-commerce platforms offer various opportunities for sellers, the process of choosing the best product at the most affordable price has become increasingly difficult for the end user. The price differences across sellers on multiple platforms are often minimal, which complicates the decision-making process.

Despite the advancements made by developed online marketplaces, customers still struggle with finding the "most preferred product at the most affordable price." This challenge becomes more pronounced when shopping in physical stores. Our system is designed to address these difficulties. With our solution, customers can easily determine the best price and the nearest store location for a product they are considering. If the product is unavailable in a physical store, the system will recommend the most suitable online marketplace.

The key insight behind this system is that customers instinctively seek "better prices" when they find products in physical stores that seem overpriced. Our AI-powered software, equipped with advanced artificial intelligence models, will provide real-time information on the best prices and locations for these products, ensuring that end users can make purchases under the most optimal conditions.

1.3 Glossary

- **End User**: The person or group who uses a product or service at the final stage.
- **E-commerce**: The buying and selling of products or services over the internet.
- **Physical Stores**: Brick-and-mortar locations where products are sold directly to customers face-to-face.
- **Artificial Intelligence (AI)**: Computer systems and software designed to mimic human intelligence.
- **AI Model**: An algorithm trained to perform a specific task or make predictions.
- **Smartphone**: A mobile device that combines phone functions with advanced features like internet access and application use.

- **API (Application Programming Interface)** : Protocols for software interaction, enabling communication between systems.
- **Camera Interface** : Feature allowing users to capture product images for information retrieval.
- **Database** : Structured data storage for user profiles, product data, and transactions.
- **Hardware Interfaces** : Physical components the application interacts with, like cameras and sensors.

1.4 Overview of the Document

The second section of this document describes the functionalities of **iProVis: Intelligent Product Vision System**. The **Overall Description** provides an overview of the project's objectives, areas of application, and its intended purpose. It explains in detail how the project will assist customers and highlights the features that underscore its importance for the end user. One of the key sections, **Development Methodology**, further explains the project's construction process and the methods used to guide its development.

2. Overall Description

This project aims to revolutionize the retail experience by leveraging computer vision and deep learning technologies to replace traditional barcodes and QR codes for product recognition in sectors such as supermarkets, clothing, food, and electronics. The developed system addresses multiple consumer needs, including effortless access to product price information, comparative price analysis across stores, and multilingual support for accessing detailed product content information.

The product is designed as a standalone mobile application integrated with the iPRoVis system's API for advanced functionality. This integration enables not only product identification but also facilitates an intelligent recycling process by recognizing recyclable packaging types, determining whether the packaging is empty or full, and providing weight details. This functionality ensures efficient sorting and recognition by smart recycling machines, thereby reducing the time and effort required for waste disposal and recycling.

The system is developed with the dual purpose of enhancing the consumer shopping experience and contributing to sustainability goals under a green IT strategy. By promoting environmentally friendly practices and providing advanced features like multilingual support, the product caters to a diverse user base from various cultural and linguistic backgrounds.

The project also proposes an innovative framework for recycling by connecting product recognition capabilities to smart recycling systems, encouraging sustainable behaviors among users. This holistic approach not only solves immediate user needs but also aligns with long-term sustainability objectives, making it a transformative solution for both the retail and environmental sectors.

2.1.1. Development Methodology

For the development of this project, we have adopted the Agile methodology, specifically leveraging the Scrum framework, which is ideal for dynamic and iterative project needs. Scrum divides the work into smaller, manageable cycles called sprints, each lasting a fixed duration of 2-4 weeks and delivering a potentially shippable product increment. This iterative approach enables flexibility and continuous improvement throughout the development process.

In our Scrum-based workflow, tasks are organized and tracked through a Scrum Board, which consists of six distinct phases: Project Backlog, containing all planned tasks; To Do, listing prioritized tasks ready for the sprint; In Progress, showing tasks currently under development; In Review, for tasks being tested and validated; To Deploy, indicating modules ready for integration;

and Done, where completed and verified tasks are marked. These phases ensure a structured and transparent workflow.

Sprint	Phase	Activities	Expected Outcome
Sprint1	Phase 1	Planning	Conduct requirement gathering and finalize project scope. Define project backlog, breaking down tasks into epics and user stories. Prioritize tasks in the backlog based on stakeholder input and project goals. Prepare initial wireframes and prototypes for the mobile application. Establish the development environment, tools, and repository structure.
Sprint 2			Develop the detailed architecture for the system, including computer vision and API integrations. Finalize UI/UX designs for the mobile application. Plan data acquisition strategies for training the computer vision models. Conduct risk assessments and create mitigation strategies.
Sprint 3	Phase 2	Training & Testing	Acquire and preprocess data for the computer vision model. Begin training the machine learning models for product recognition. Implement basic functionality for mobile app features, such as user registration and login. Conduct initial unit testing for developed modules.
Sprint 4			Optimize and fine-tune the machine learning models to improve accuracy. Develop price comparison and multilingual product detail features. Test API integration with iPRoVis for smart recycling functionality. Perform integration testing for the mobile application.
Sprint 5			Implement error handling and security measures in the application. Conduct performance testing for computer vision recognition under varying conditions. Execute user acceptance testing (UAT) with a select group of participants. Collect feedback from UAT and prioritize improvements in the backlog.
Sprint 6	Phase 3	Go-Live	Finalize all features and resolve outstanding bugs identified during UAT. Deploy the application to a staging environment for final validation. Conduct end-to-end testing to ensure system reliability and scalability. Train administrators on using the system for product data management.
Sprint 7			Prepare deployment plans and set up production servers. Launch the application to the public. Monitor system performance post-launch, addressing any issues promptly. Collect initial user feedback and plan for post-launch improvements.

Each sprint begins with detailed planning, including story points and risk assessments for tasks. The team conducts daily stand-up meetings, limited to 15 minutes, to communicate progress, address potential blockers, and ensure alignment toward sprint goals. Scrum roles are clearly defined to streamline this process: the Product Owner defines and prioritizes project requirements, the Scrum Master ensures adherence to the Scrum framework and facilitates team efficiency, and the Development Team collaboratively works to achieve sprint objectives.
Scrum's iterative nature allows the team to adapt to evolving requirements and incorporate stakeholder feedback after each sprint. Delivering a working increment at the end of every sprint ensures early validation and continuous improvement of the product. This adaptability is particularly advantageous for the project, as it accommodates changes while maintaining high-quality outcomes.

Phase	Backlog	To Do	In Progress	In Review	To Deploy	Done
Planning	Requirement gathering pending	Create prototypes	Set up development environment			
Planning	Finalize UI/UX designs	Plan data acquisition strategies	Finalize system architecture			
Training & Testing	Define data preprocessing steps	Acquire and preprocess data	Train machine learning model			
Training & Testing	Implement error handling	Test API integration	Develop price comparison feature			
Go-Live	Resolve final UAT feedback	End-to-end testing	Deploy to staging environment			
Go-Live	Plan for post-launch improvements	Training & Testing	Launch application			

This project aims to transform the retail industry by replacing traditional barcode and QR code scanning with computer vision technology for product recognition. Through its mobile application, users can access product price information, compare prices across stores, and obtain multilingual product details. Additionally, the system integrates with smart recycling through the iPRoVis API, supporting sustainability by enabling automated sorting of recyclable packaging.
By incorporating Agile principles and leveraging Scrum, this project ensures a structured yet flexible approach. The methodology facilitates timely, high-quality delivery while addressing innovative and sustainable goals, ultimately enhancing consumer experiences and contributing to environmental responsibility.

2.2 User Characteristics

This section defines the distinct user groups who will interact with the system. Users are classified based on their technical knowledge, expectations, and roles, providing clarity on how the system is designed to meet their needs.

2.2.1 Participants

Participants represent the general consumer audience who use the mobile application to access product information. Their interaction with the system includes learning about product prices, comparing prices across stores, and accessing detailed product content information.
2.2.1.1. General Consumer Representation: Participants are typical consumers who rely on the system to make informed purchasing decisions by comparing prices and analyzing product details through the mobile application.
2.2.1.2. Ease of Use: Participants require no prior technical knowledge to use the system. A user-friendly interface ensures they can navigate and complete tasks effortlessly.
2.2.1.3. Multilingual Support: Participants can benefit from multilingual features that enable them to view product information in multiple languages, ensuring accessibility across diverse regions.
2.2.1.4. Recycling Integration: The system supports participants in streamlining recycling processes by integrating with smart recycling systems for products designed with sustainable practices.

2.2.2 Admin

Admins are the authorized personnel responsible for managing and overseeing the system's functionality. Their primary roles include updating product data, maintaining system integrations, and ensuring overall performance.
2.2.2.1. System Oversight: Admins monitor and manage the system's operations, ensuring smooth functionality and addressing user feedback effectively.
2.2.2.2. Data Management: Admins are responsible for updating product information, including pricing and descriptions, and addressing user-submitted feedback.
2.2.2.3. Technical Competence: Admins require basic computer skills to operate the system and perform their duties efficiently.
2.2.2.4. System Integration Management: Admins oversee the integration of the iPRoVis API and smart recycling systems, ensuring seamless functionality and accurate data flow.
2.2.2.5. Multilingual Capabilities: Admins must ensure the accuracy and consistency of content across multiple languages and may require familiarity with different languages to manage multilingual features effectively.

3. REQUIREMENTS SPECIFICATION

3.1 External Interface Requirements

3.1.1 User Interfaces

- **Mobile Application Interface**
  - The design should be user-friendly, enabling smooth navigation.
  - A camera interface should be integrated for product image capture.
  - Display detailed product information, including price, stock status, and multilingual content.
  - Include a price comparison feature that shows a list of stores and their prices.
  - Provide a user profile management section to save preferences and viewing history.

3.1.2 Hardware Interfaces

- **Camera**
  - Integration with the device's camera for image capture.
- **Sensors**
  - Utilize ambient light sensors to optimize image capture conditions.
- **Internet Connectivity**
  - Support for Wi-Fi or mobile data connections to facilitate server communication.

3.1.3 Software Interfaces

- **Operating Systems**
  - The app should be compatible with Android platforms.
- **APIs**
  - Integration with third-party APIs for price comparison and retrieving product information.
- **Database**
  - The application should connect to a backend database to store user profiles, product data, and transaction history.

3.1.4 Communications Interfaces

- **Network Protocols**
  - HTTPS should be used for secure communication with the iProViS server.
- **Data Formats**
  - JSON or CSV formats should be employed for data exchange between the mobile app and the server.

3.2 Functional Requirements

3.2.1 Login/Register Feature

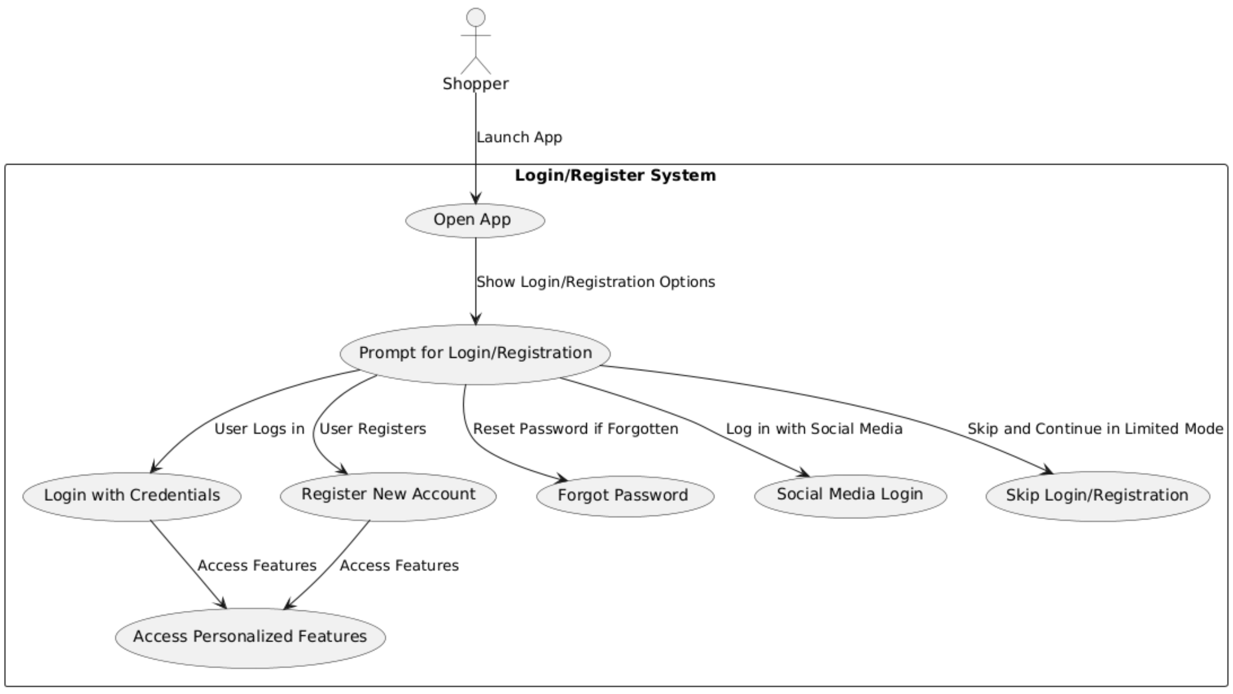
- **Primary Actor:** Shopper/Customer
- **Goal:** Allow users to log into their account or create a new one for personalized features.
- **Preconditions:** The app is installed, and the user is either logging in for the first time or after a session has ended.
- **Postconditions:** The user gains access to personalized features, including saved searches and activity history.

Main Flow:

1. Open the app: The user launches the app for the first time or after logging out.
2. Login/Register prompt: The app offers options to log in or create a new account (e.g., “Log In” or “Create Account”).
3. Login: If the user has an existing account, they enter their credentials (e.g., email or username and password). The app verifies these and grants access to personalized features.
4. Registration (for new users): When selecting “Create Account,” the user provides necessary details (email, password, name, shipping address, etc.). Once submitted, the account is created, and the user is logged in.
5. Account settings: Users can modify preferences like language or notification settings.
6. Access personalized features: After logging in, users can scan products, view saved preferences, and receive targeted offers.

Alternative Flow:

- Forgot Password: If the user forgets their password, a reset link is sent via email.
- Social Media Login: Users can log in via social media accounts (e.g., Google, Facebook, or Apple ID).
- Skip Login: Users can skip the login and use the app in limited mode, with an option to log in for a personalized experience.



3.2.2 Capture Product Photo for Information Retrieval

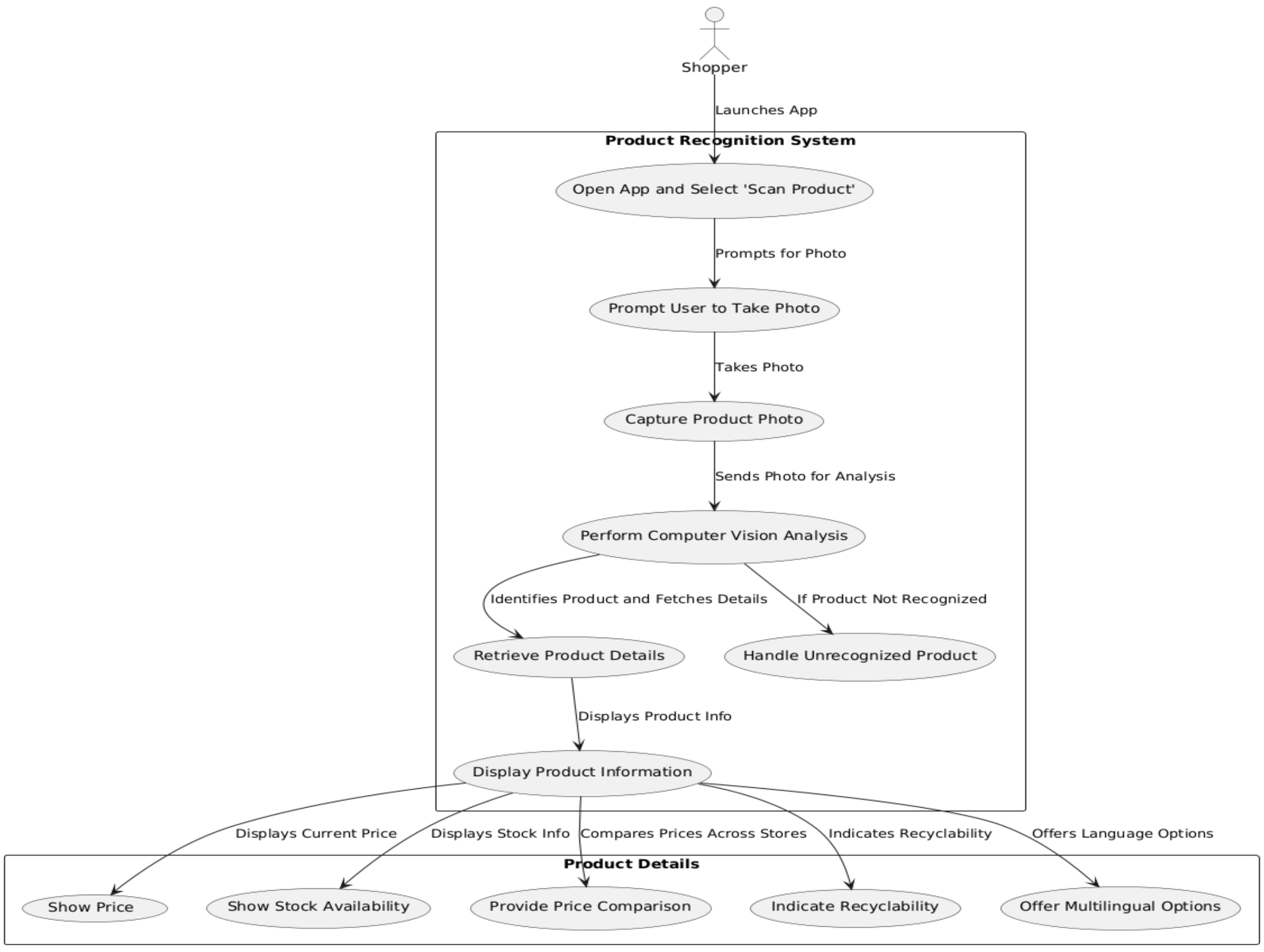
- **Primary Actor:** Shopper/Customer
- **Goal:** Capture a product photo to retrieve detailed information.
- **Preconditions:** The app is installed, and the product is visible to the camera.
- **Postconditions:** The user receives detailed information, including price, stock status, price comparisons, recyclability, and multilingual options.

Main Flow:

1. The user opens the app and selects "Scan Product" or "Take Photo."
2. The app prompts the user to take a clear photo of the product.
3. The user captures the photo.
4. The app uses computer vision to identify the product.
5. Product details are displayed:
   - Price: The product's current price.
   - Stock Availability: Whether the product is in stock or not.
   - Price Comparison: Comparisons across stores (online or offline).
   - Recyclability: Information about whether the product or packaging is recyclable.
   - Multilingual Support: Option to view details in different languages.

Alternative Flow:

- If the product cannot be recognized, the app notifies the user and offers manual entry or retry options.



3.2.3 Price and Stock Availability Display

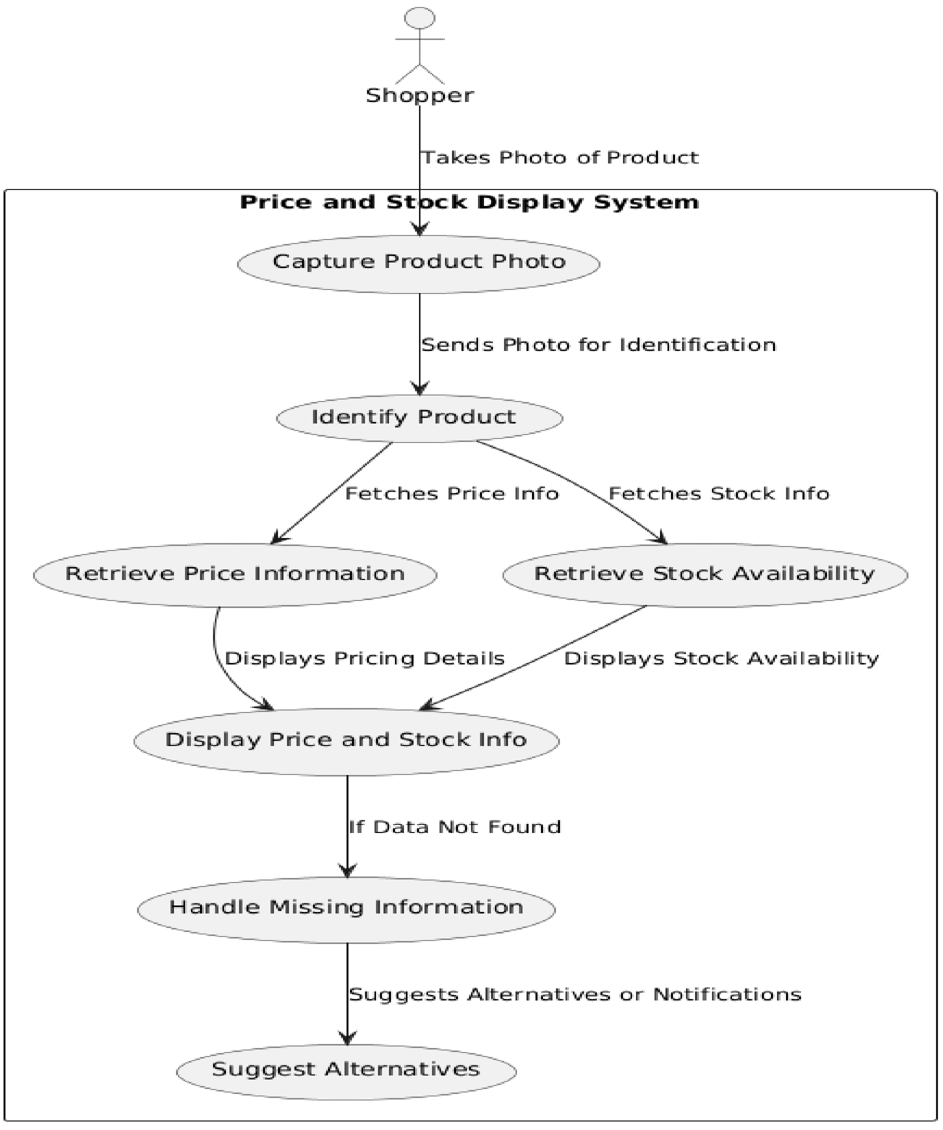
- **Primary Actor:** Shopper/Customer
- **Goal:** View price and stock status for a product.
- **Preconditions:** The user has taken a product photo, and the app has recognized it.
- **Postconditions:** The user views current pricing and stock information.

Main Flow:

1. After capturing the product photo, the app retrieves and displays the price.
2. It shows stock availability at nearby stores or online retailers.
3. The user can click on a store or retailer to view more details.

Alternative Flow:

- If the price or stock is unavailable, the app notifies the user and suggests alternatives or offers to notify them when the product is back in stock.


Shopper
Takes Photo of Product
Price and Stock Display System
Capture Product Photo
Sends Photo for Identification
Identify Product
Fetches Price Info
Fetches Stock Info
Retrieve Price Information
Retrieve Stock Availability
Displays Pricing Details
Displays Stock Availability
Display Price and Stock Info
If Data Not Found
Handle Missing Information
Suggests Alternatives or Notifications
Suggest Alternatives

3.2.4 Price Comparison Feature

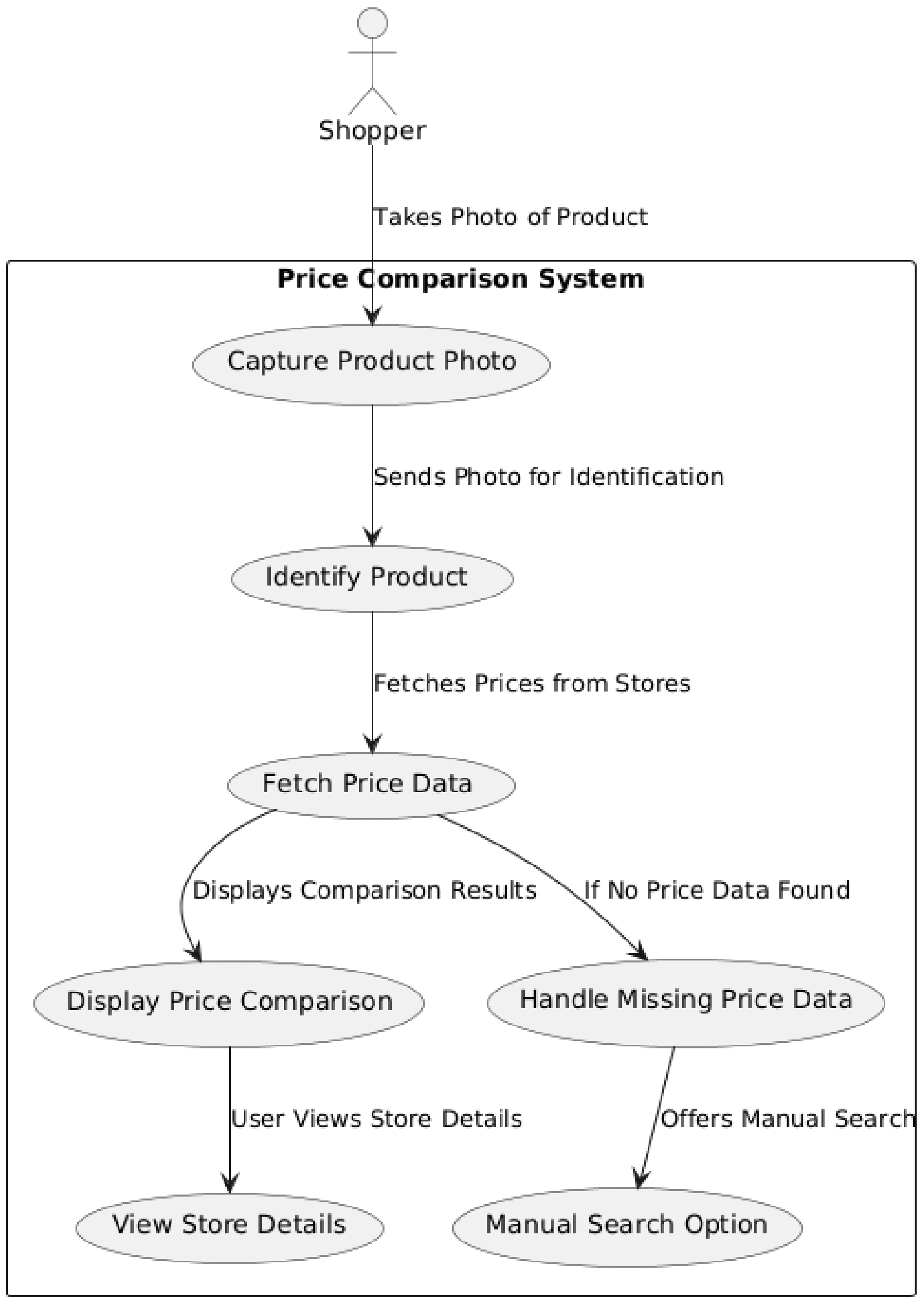
- **Primary Actor:** Shopper/Customer
- **Goal:** Compare prices for the product across different retailers.
- **Preconditions:** The user has taken a product photo, and the app has identified it.
- **Postconditions:** The user sees a comparison of prices from various stores.

Main Flow:

1. After the product is recognized, the app retrieves price data from multiple retailers.
2. A comparison is displayed, showing the lowest and highest prices.
3. The user can select a retailer to view additional details.

Alternative Flow:

- If no comparison is available, the app notifies the user and offers an option for manual search.



3.2.5 Recyclability Information Display

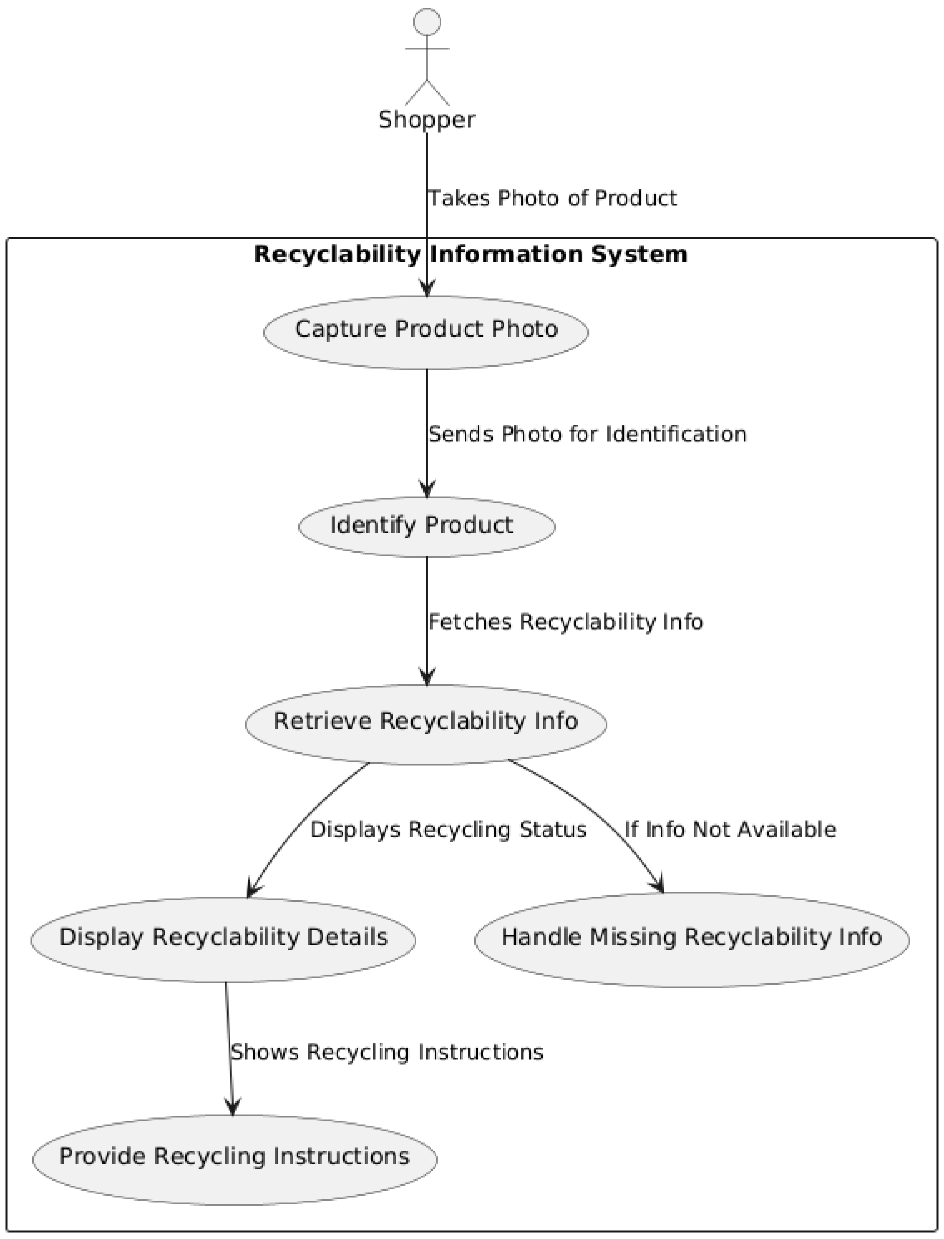
- **Primary Actor:** Shopper/Customer
- **Goal:** Display recyclability information about the product.
- **Preconditions:** The user has taken a product photo, and the app has recognized it.
- **Postconditions:** The user learns if the product or packaging is recyclable.

Main Flow:

1. After product recognition, the app displays recyclability information.
2. An icon or label indicates whether the product is recyclable (e.g., "Recyclable," "Not Recyclable," or "Check local guidelines").
3. If recyclable, the app provides additional recycling instructions.

Alternative Flow:

- If recyclability data is unavailable, the app prompts the user to check the packaging or contact the manufacturer for further information.



Shopper
Takes Photo of Product
Recyclability Information System
Capture Product Photo
Sends Photo for Identification
Identify Product
Fetches Recyclability Info
Retrieve Recyclability Info
Displays Recycling Status
If Info Not Available
Display Recyclability Details
Handle Missing Recyclability Info
Shows Recycling Instructions
Provide Recycling Instructions

3.2.6 Multilingual Support for Product Information

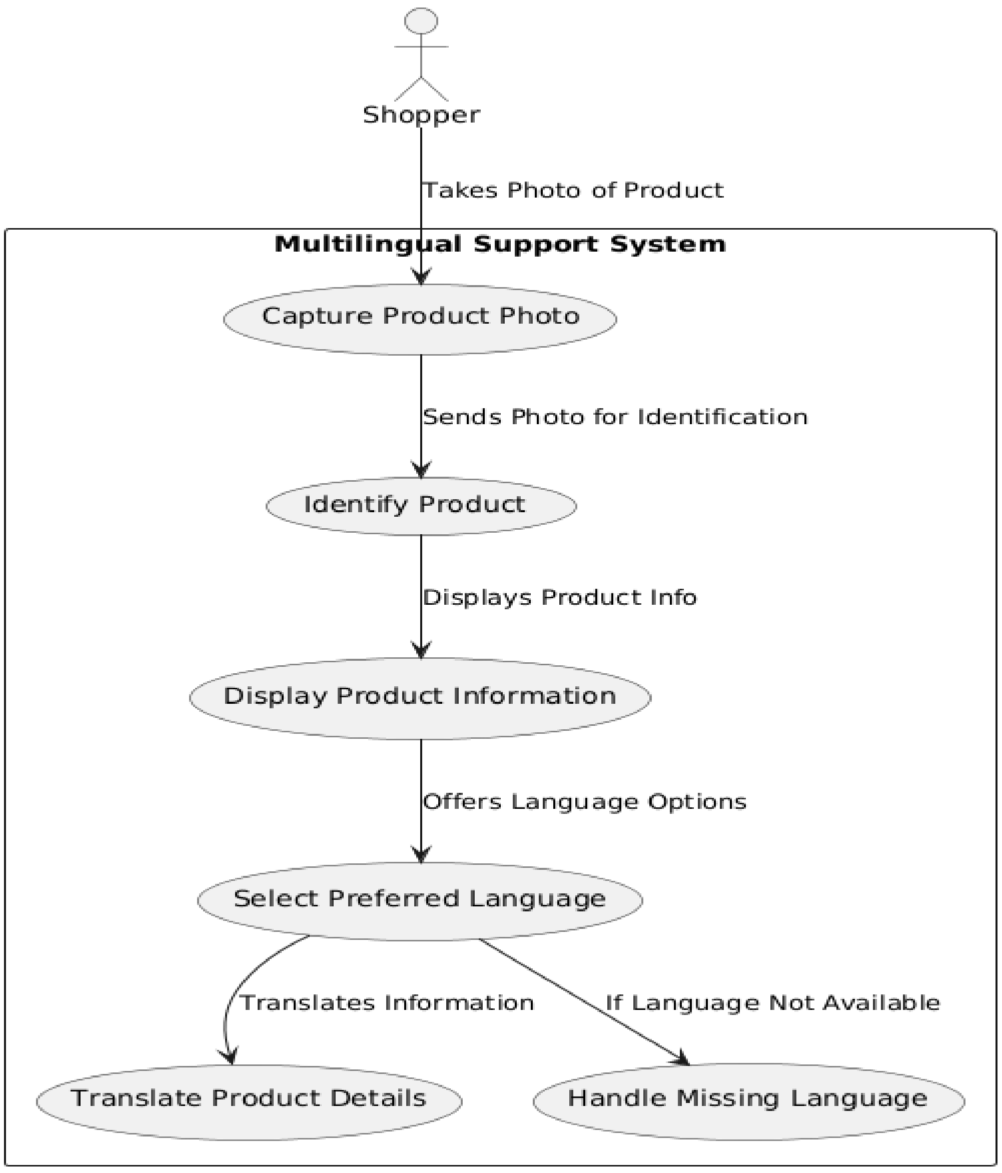
- **Primary Actor:** Shopper/Customer
- **Goal:** View product information in the user's preferred language.
- **Preconditions:** The user has taken a product photo, and the app has identified it.
- **Postconditions:** The product details are displayed in the selected language.

Main Flow:

1. Once product details (e.g., price, stock, recyclability) are shown, the app offers an option to switch to another language.
2. The user selects a preferred language (e.g., English, Spanish, French).
3. The app translates all details into the chosen language.

Alternative Flow:

- If the selected language is unavailable, the app informs the user or provides a translation tool.


Shopper
Takes Photo of Product
Multilingual Support System
Capture Product Photo
Sends Photo for Identification
Identify Product
Displays Product Info
Display Product Information
Offers Language Options
Select Preferred Language
Translates Information
If Language Not Available
Translate Product Details
Handle Missing Language

3.2.7 Feedback and Additional Product Information

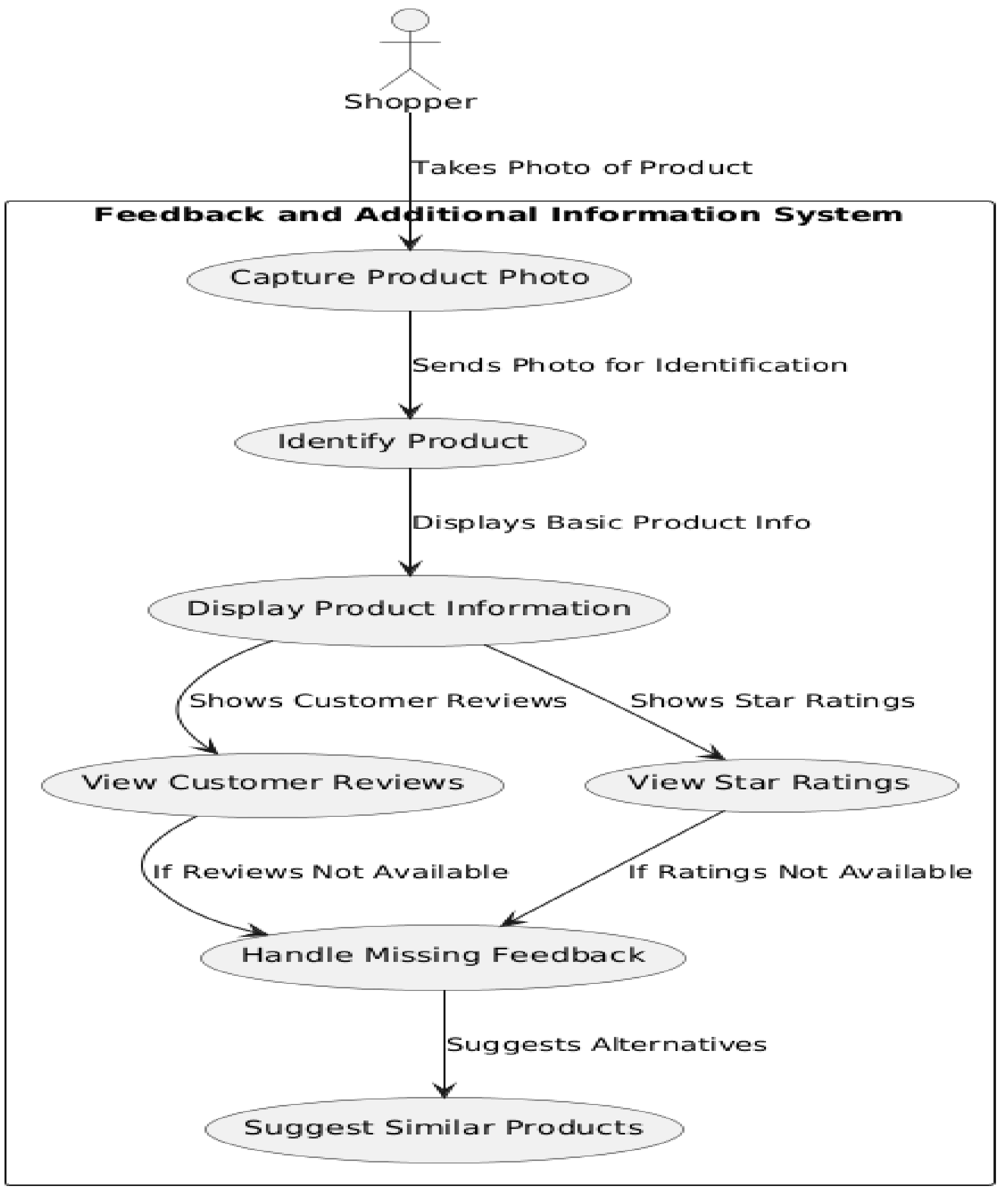
- **Primary Actor:** Shopper/Customer
- **Goal:** Access additional information, such as reviews and ratings.
- **Preconditions:** The user has taken a product photo, and the app has recognized it.
- **Postconditions:** The user gains insights into the product via reviews and ratings.

Main Flow:

1. After displaying product details, the app shows options to view customer reviews and ratings.
2. The app may display star ratings, verified reviews, and product demonstration videos.
3. The user can scroll through reviews or click on ratings for more details.

Alternative Flow:

- If reviews or ratings are unavailable, the app notifies the user and suggests similar products.



3.2.8 Notification of New Deals or Discounts

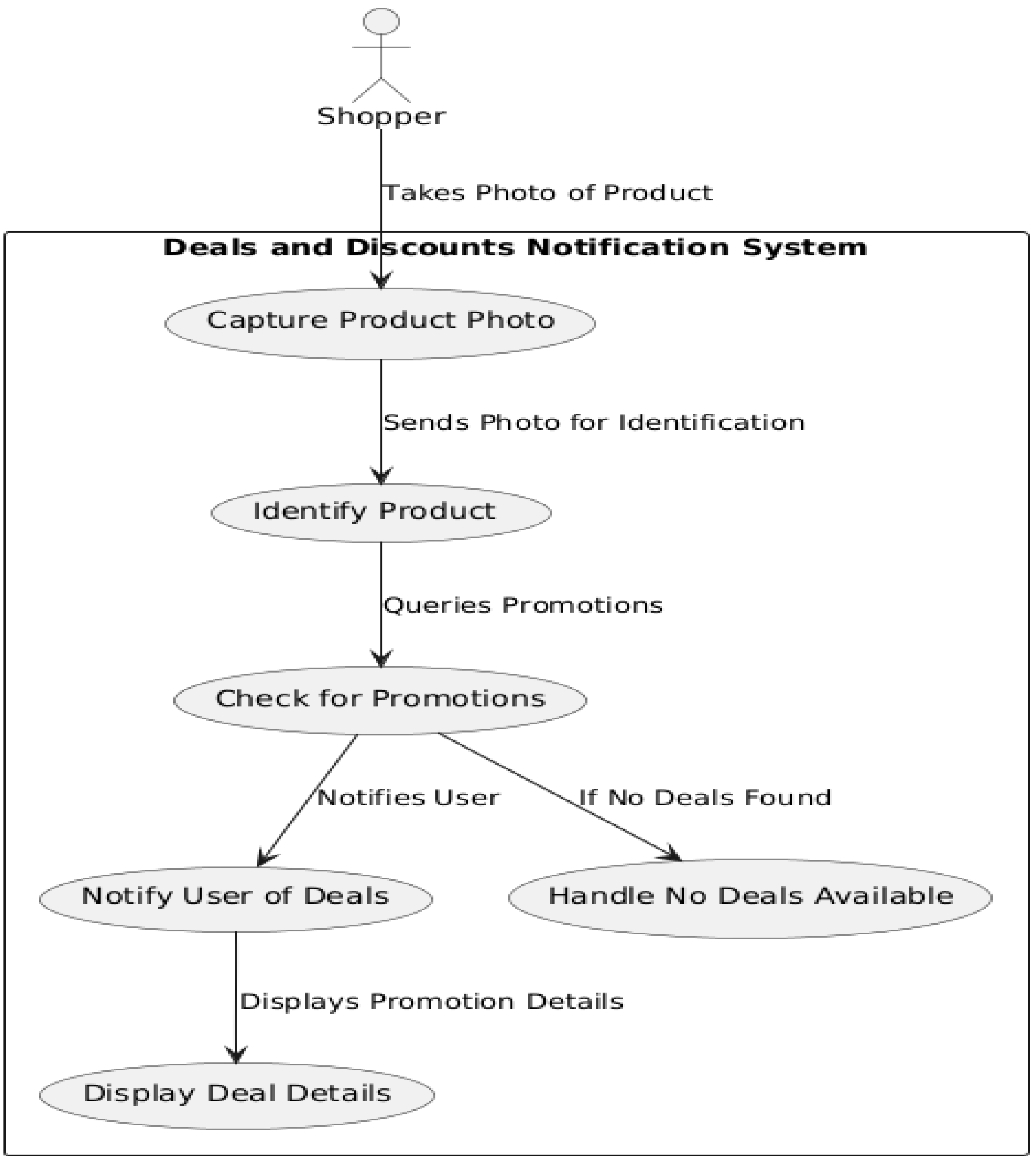
- **Primary Actor:** Shopper/Customer
- **Goal:** Receive notifications about available deals or discounts.
- **Preconditions:** The user has taken a product photo, and the app has recognized it.
- **Postconditions:** The user is notified of any available deals.

Main Flow:

1. The app checks for any ongoing promotions on the product.
2. It displays any available deals (e.g., discounts, coupon codes, or bundle offers).
3. The user can click on the offer for more details or apply it to their purchase.

Alternative Flow:

- If no discounts are available, the app informs the user that there are no current offers.



3.2.9 Handling Unrecognized Products

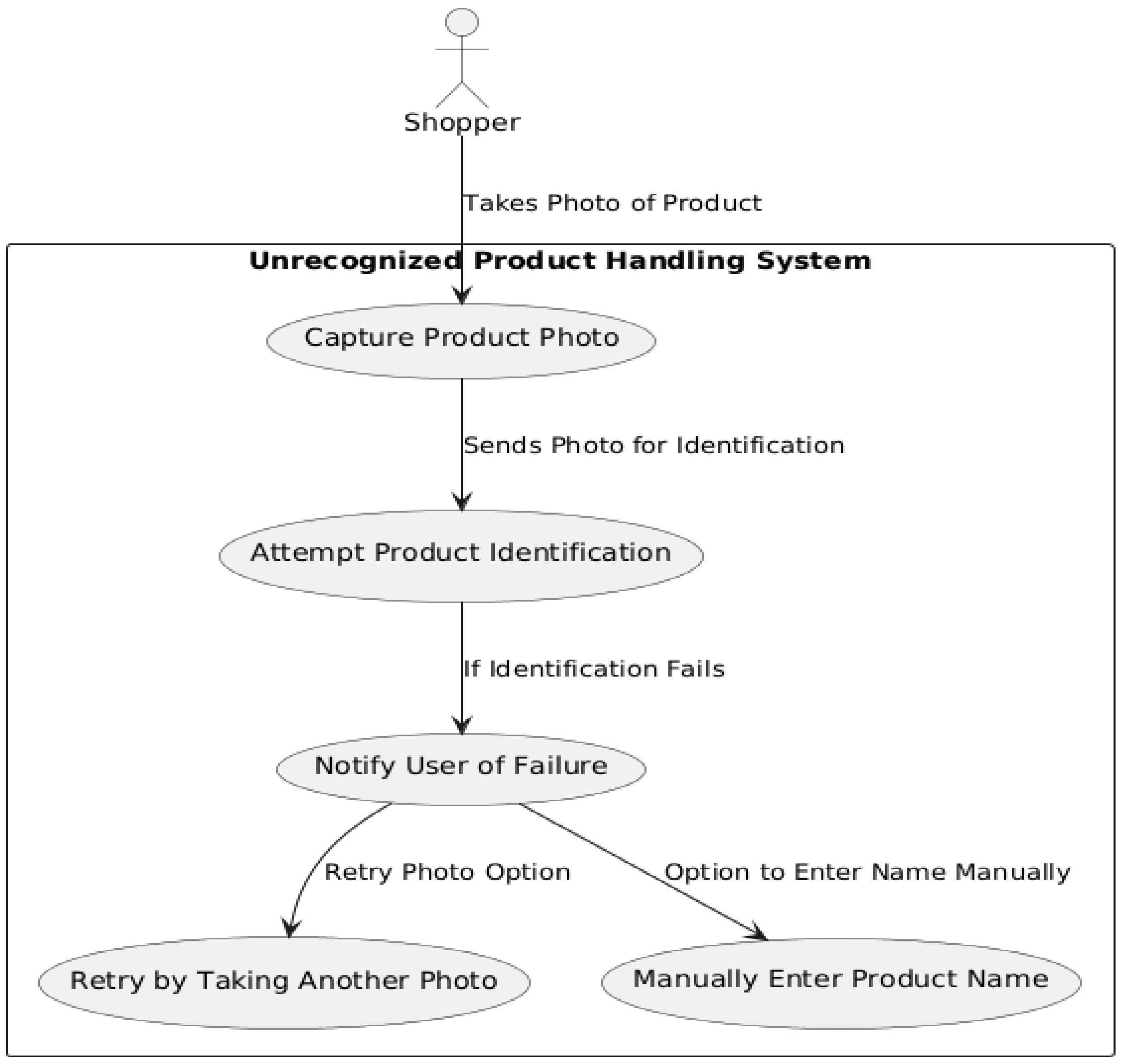
- **Primary Actor:** Shopper/Customer
- **Goal:** Assist users when the app fails to recognize a product.
- **Preconditions:** The user has taken a product photo, but the app can't identify it.
- **Postconditions:** The user is given an alternative method to find the product information.

Main Flow:

1. If the app fails to recognize the product, it notifies the user.
2. The user can manually enter the product name or retake the photo.
3. Other search methods (e.g., barcode scanning or uploading an image) are provided.

Alternative Flow:

- If manual search options fail, the app offers a "Help" feature.


Shopper
Takes Photo of Product
Unrecognized Product Handling System
Capture Product Photo
Sends Photo for Identification
Attempt Product Identification
If Identification Fails
Notify User of Failure
Retry Photo Option
Option to Enter Name Manually
Retry by Taking Another Photo
Manually Enter Product Name

3.3 Performance Requirements

The application must deliver an efficient, reliable, and responsive user experience, even under heavy usage. Here are the key performance considerations:

3.3.1 Response Time

- **Product Recognition:** The app should recognize a product within 2 seconds, assuming optimal lighting and focus conditions.
- **Price and Stock Retrieval:** Product price and stock data should appear within 3 seconds of recognition.
- **Price Comparison:** The comparison data should load within 5 seconds of retrieving product details.

3.3.2 System Throughput

- **Concurrent Users:** The app must support at least 100 concurrent users without significant performance issues.
- **Data Handling:** The backend should handle up to 10,000 transactions per hour without noticeable degradation.

3.3.3 Availability and Reliability

- **Service Availability:** The app must be available 99.9% of the time, with downtime only for planned maintenance.
- **Error Handling:** The system should display appropriate error messages and automatically retry failed processes when needed.

3.3.4 Scalability

- **User Growth:** The backend should scale horizontally to accommodate growing user numbers and app traffic.
- **Database Scaling:** The database must scale efficiently as the number of products, transactions, and user profiles increases.

3.3.5 Latency

- **Network Latency:** Average network latency should remain under 1 second for most operations.
- **Image Processing Latency:** Product recognition should occur within 2 seconds after image capture.

3.3.6 Resource Utilization

- **Memory Usage:** The app should not exceed 200MB of RAM during normal operations.
- **Battery Consumption:** The app should consume no more than 10% battery per hour during regular use.
- **Data Consumption:** The app should use no more than 50MB of data per session for product searches and comparisons.

3.3.7 Load Balancing and Redundancy

- **Server Load Balancing:** Load balancing mechanisms must ensure even traffic distribution.
- **Failover Mechanism:** Critical services should have a failover mechanism to maintain high availability.

3.3.8 Caching

- **Frequent Data Caching:** Frequently requested data should be cached to improve performance. The cache should refresh at regular intervals to ensure up-to-date information.

These performance specifications are designed to ensure the app delivers a fast, reliable, and efficient experience under varying conditions, while being scalable, resilient, and responsive to user demands.

3.4 Software System Attributes

3.4.1 Portability

The app should be usable across various devices with different screen sizes and resolutions.

3.4.2 Performance

- Product recognition should happen within 2 seconds under optimal conditions.
- The app should support 100 concurrent users without degradation.

3.4.3 Usability

The app should have an intuitive interface with minimal learning required. Accessibility features must also be provided.

3.4.4 Adaptability

The app should adapt to different lighting conditions to ensure optimal image recognition.

3.4.5 Scalability

The backend should be able to scale as the user base and product data grow.

3.5 Safety Requirements

The app must comply with data protection regulations (e.g., GDPR) and ensure user data privacy. Mechanisms should be in place to prevent unauthorized access to sensitive information.

4. REFERENCES

[1] Li, Q., Peng, X., Cao, L., Du, W., Xing, H., Qiao, Y., & Peng, Q. (2020). Product image recognition with guidance learning and noisy supervision. Computer Vision and Image Understanding, 196, 102963.
[2] Bai, Y., Chen, Y., Yu, W., Wang, L., & Zhang, W. (2020). Products-10k: A large-scale product recognition dataset. arXiv preprint arXiv:2008.10545.
[3] Chen, S., Liu, D., Pu, Y., & Zhong, Y. (2022). Advances in deep learning-based image recognition of product packaging. Image and Vision Computing, 128, 104571.