CrowdMarket

Technical Specification Document

Introduction

This document presents the technical details for the CrowdMarket system. The functionalities already described in the presentation document are here further refined in terms of requirements for the system-to-be, as well as high-level specification diagrams for the system architecture and activities related to each functionality. We also provide the mockups for the mobile application, which are useful to illustrate the tool providing users the interfaces for data collection, validation, and consumption.

This document is not auto-contained; the reader must refer to the accompanying *presentation document* for further details on the problem and proposed solution. Hereafter we refer to the later document as *PD*.

Proposed System

Requirements Elicitation

In alignment with the functionalities of the CrowdMarket (as described in the Functionalities section of the PD), the following requirements are elicited for the CrowdMarket System.

Missing Product Registration

R1. Users must have access to an interface where they can insert missing products
   a. Users must be able to take a picture from the product barcode
   b. Product details must be fetched from the backend application
   c. Users must be able to insert missing data regarding the following mandatory fields: *name, brand, specifier, category*
      i. Product categories are unique and pre-inserted by system administrators
      ii. Users must be able to choose from an existing brand or to insert a new brand
   d. Users must be informed of a successful product insertion

Single Product Update

R2. Users must have access to an interface where they can add or update product prices of single products
   a. The user must be able to search for the product by barcode, name, brand, or category

b. The user must be given the option to navigate to the product insertion interface if the product is not found
    i. Upon missing product insertion, the system must navigate back to the price update interface with the corresponding product pre-selected
c. If the product exists, the current price must be displayed
d. The user must be able to update the product price as seen in the supermarket
    i. The user must be able to insert the normal product price
    ii. If no normal price is given, the user must be prompted with an alert to insert one
e. The user must be able to insert the absolute discounted product price if an offer exists in that supermarket
    i. If a new discounted price is given, it must replace the previous one
    ii. If the new discounted price is higher or equal the normal price, the user must be prompted with an alert informing the inconsistency
    iii. If no discounted price is given, the previous discounted price must be discarded
f. The price update must be validated
    i. If the normal price is *k* times higher or lower than the current normal price, the product price must be marked for manual validation (see Sec. Validation in the PD)

Receipt-based Product Update

R3. Users must have access to an interface where they can add or update prices from their shopping receipt
    a. Users must be able to attach an image from the receipt
    b. Users must be able to insert the total price of the receipt
    c. Users must be able to insert the data of the receipt
    d. The user must be able to update the product price as seen in the receipt
        i. The user must be able to insert the normal product price
        ii. If no normal price is given, the user must be prompted with an alert to insert one
    e. The user must be able to insert the absolute discounted product price as seen in the receipt
        i. If a new discounted price is given, it must replace the previous one
        ii. If the new discounted price is higher or equal the normal price, the user must be prompted with an alert informing the inconsistency
        iii. If no discounted price is given, the previous discounted price must be discarded
    f. The price update must consider the receipt date
        i. Products that have been updated after the receipt date (*update-date* > *shopping-date*) must remain with the current and most updated prices
    g. Acronyms used by different supermarket units and chains to identify identical products must be mapped to a single product identifier

        i. If the acronym does not exist or has not been mapped to a specific product, the price update is performed over a new dummy product registry associated with the acronym
        ii. If the acronym exists and has been mapped to a specific product, the price update is performed over the corresponding product
    h. The receipt-based product update must be validated
        i. All data input by basic users must be manually validated by a different user (see sections User Level and Data Validation in the PD)
        ii. Data input by advanced users is subject to the *automated validation*: product prices *k* times higher or lower than the current price must be market for manual validation (see Data Validation in the PD)

Product Acronym Association

R4. Users must be able to associate orphan acronyms with a real product
    a. Users that have a favorite supermarket with orphan acronyms must be prompted with a request for associating orphan acronyms
        i. The user must be given the choice of skipping the association
    b. For each orphan acronym, the user must be able to find the corresponding product
        i. If a product is found and selected, the acronym must be associated with the product
    c. The new association should be validated
        i. If the selected product's name contains all the acronym characters, the association should be accepted as valid
        ii. Otherwise, the association should be discarded
    d. Dummy product registry must be discarded
        i. If the dummy product price is more recent, the newly associated product price must be replaced with the dummy product price
        ii. Existing association between the dummy product and the acronym must be updated with the new product

Registered Product Validation

R5. Newly registered products must be validated
    a. Users that search for market products that have been registered and not yet validated must be asked for the correctness of the product details
        i. The user must be given the choice of skipping the validation
    b. The user reply should be accepted as valid
        i. A positive reply should mark the product as valid
        ii. A negative reply should navigate to the product registration interface, from which the user may correct the product details; the product should be validated by another user before been marked as valid

Single Product Update Validation

R6. Products market for manual validation must be validated
   a. Users that search for market products must be asked for the correctness of the price
      i. The user must be within the supermarket area
      ii. The user must be given the choice of skipping the validation
   b. The user reply should be accepted as valid
      i. A positive reply should unmark the product price
      ii. A negative reply should remove the product price
   c. The positive or negative user reply should be rewarded with 1CMP (see Sec. User Levels in the PD)

Receipt-based Product Update Validation

R7. Data parsed from receipts must be validated against the provided receipt picture
   a. Users must be asked for validating other users' receipt when they open the mobile application
      i. The user must be given the choice of skipping the validation
      ii. The user decision must be remembered until the application is restarted
   b. The user must confirm or deny the validity of the receipt date
      i. A positive reply should allow the validation to proceed
      ii. A negative reply should finish the validation process and remove all receipt data from the system
   c. For each parsed product, the user must confirm or deny the validity of the price
      i. A positive reply should confirm the product price in the system
      ii. A negative reply should remove the product price
   d. Each positive or negative user reply should be rewarded with 1CMP (see Sec. User Levels in the PD)

System Architecture

Considering the proposed solution based on a stack of distributed software technologies, we propose a high system architecture which includes the main components: the *mobile application* and the *backend application*. The proposed architecture is depicted in Fig. 1.

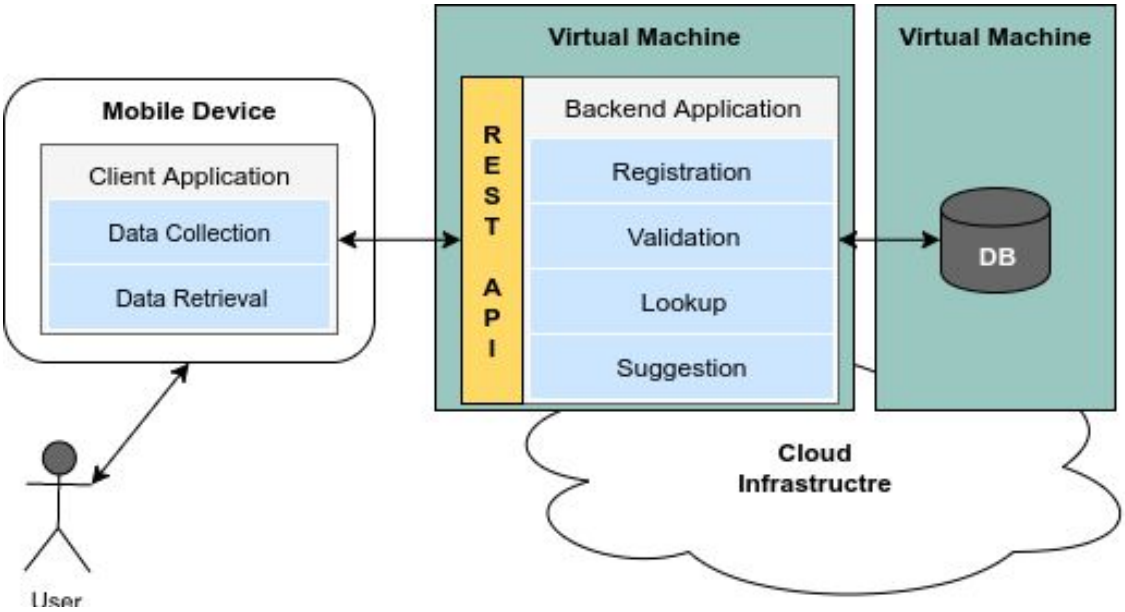

Fig. 1: High level architecture of the proposed software solution composed of a mobile (left) and a backend application (right)

The proposed system relies on the well established paradigm of cloud computing in which the computational resources required for running a backend application are provided as a service, i.e., their management is delegated to a 3th party provider. Furthermore, two popular mobile platforms - namely, iOS and Android - have been selected as targets to host the client application interfacing with the end users.

As depicted in Fig. 1, the mobile/client application interacts with a RESTful API provided by the backend application. The API includes all the services for receiving data collected by users and to retrieve data to be consumed by users. More precisely, the backend application includes both the business logic responsible for the processing of incoming and outgoing data, as well as the persistence layer responsible for the data storage. The mobile/client application, in turn, is responsible for enabling users to interface with the system both by sending data (missing product registration, price updates, automated price validation, etc) and consuming data (product search, supermarket suggestion, etc).

Entity Relationship Diagram

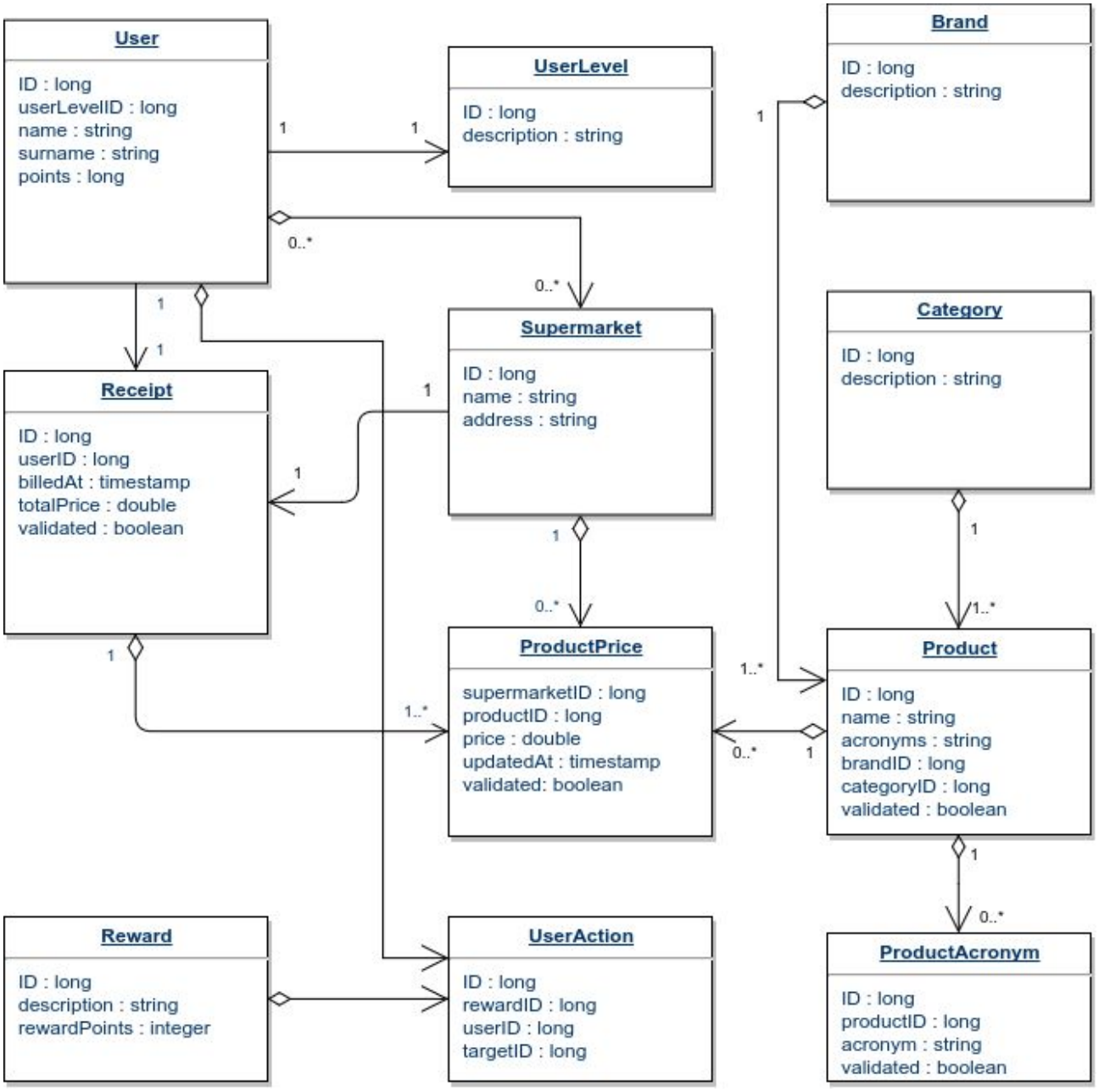

Fig. 2: Activity flow for the single product update feature

Flow Diagrams

Single Product Update

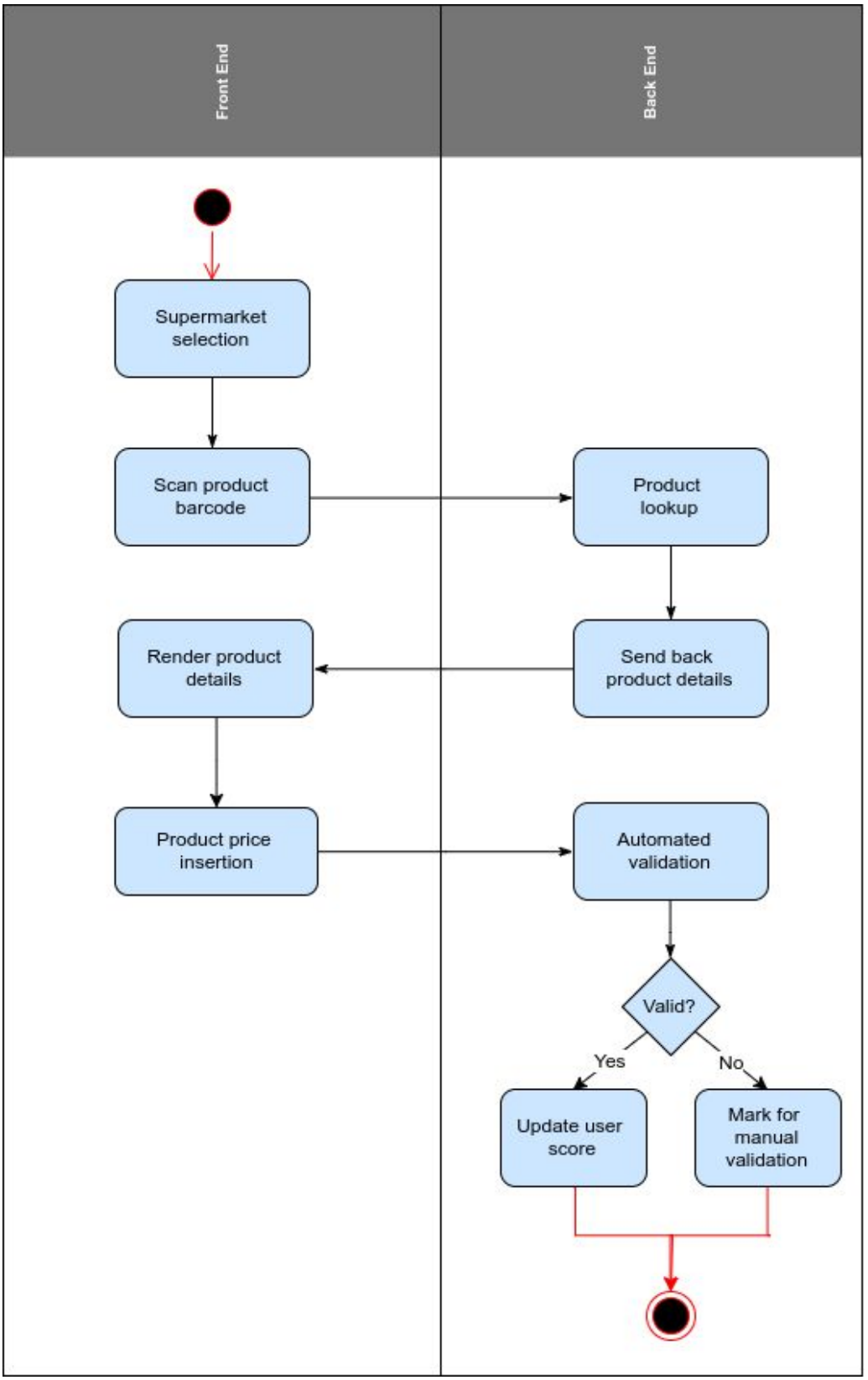

Fig. 3: Activity flow for the single product update feature

Receipt-based Product Update

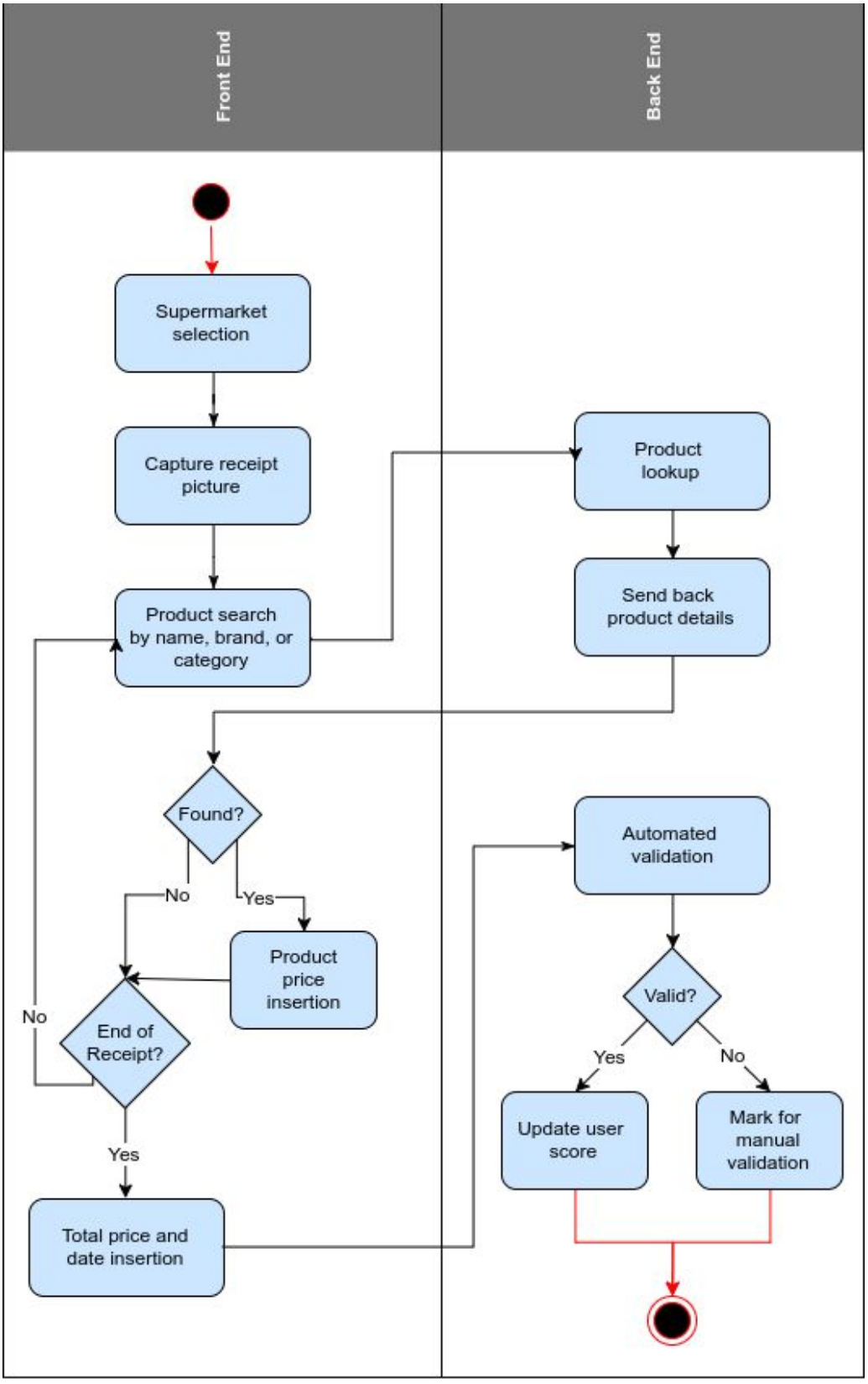

Fig. 4: Activity flow for the receipt-based product update feature

Product Search

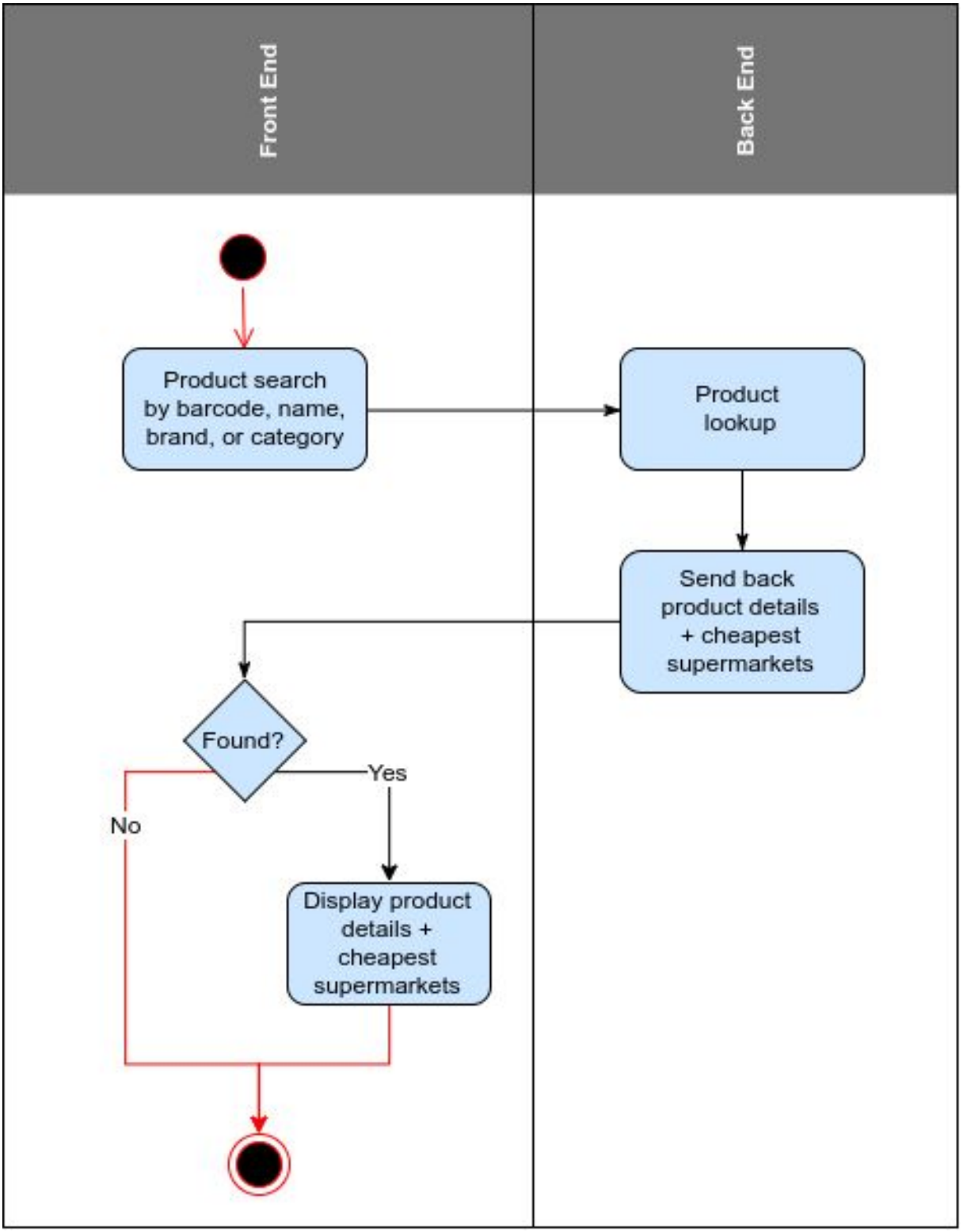

Fig. 5: Activity flow for the product search feature

Supermarket Suggestion

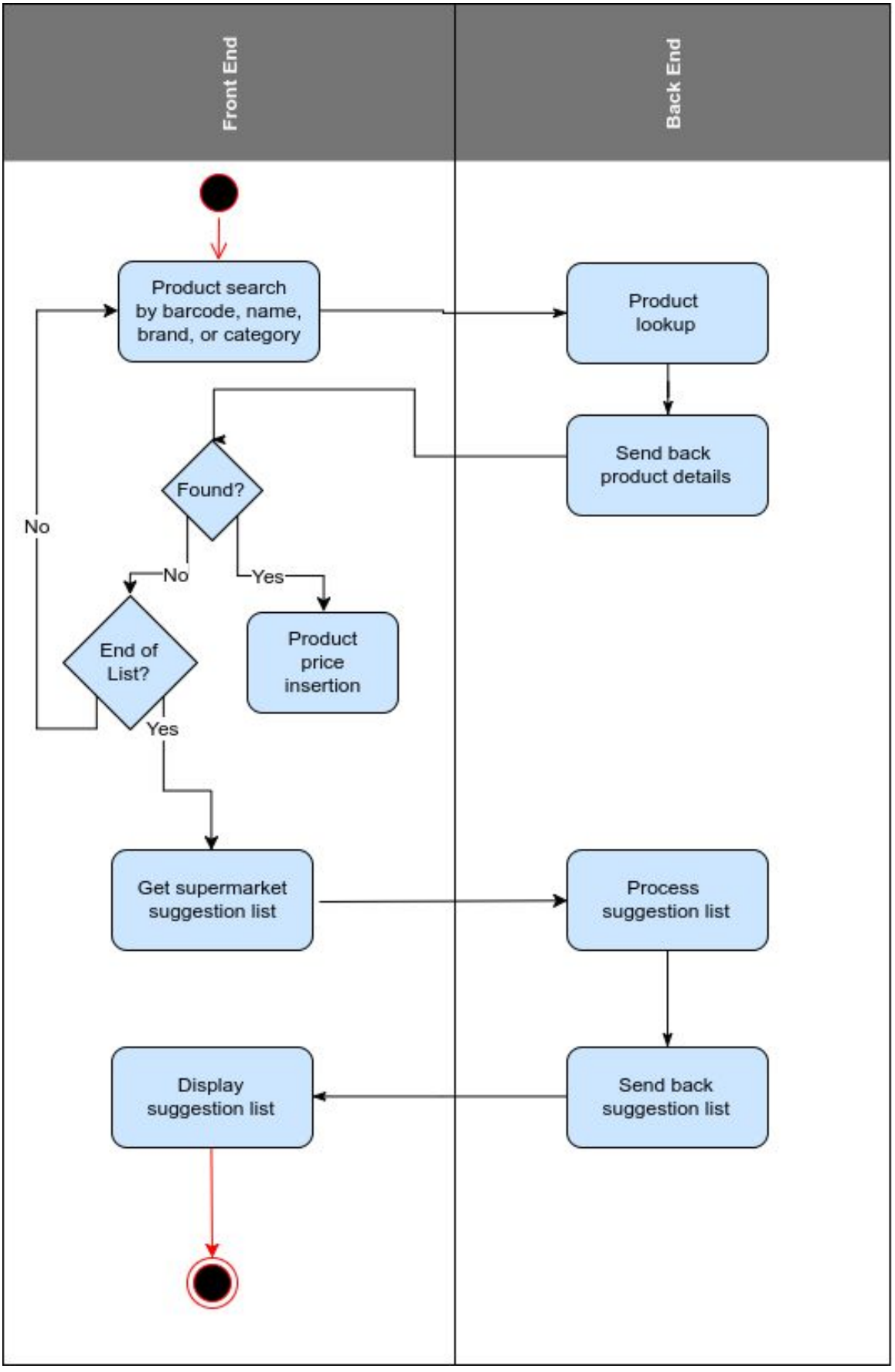

Fig.6: Activity flow for the supermarket suggestion feature

Premium Supermarket Suggestion

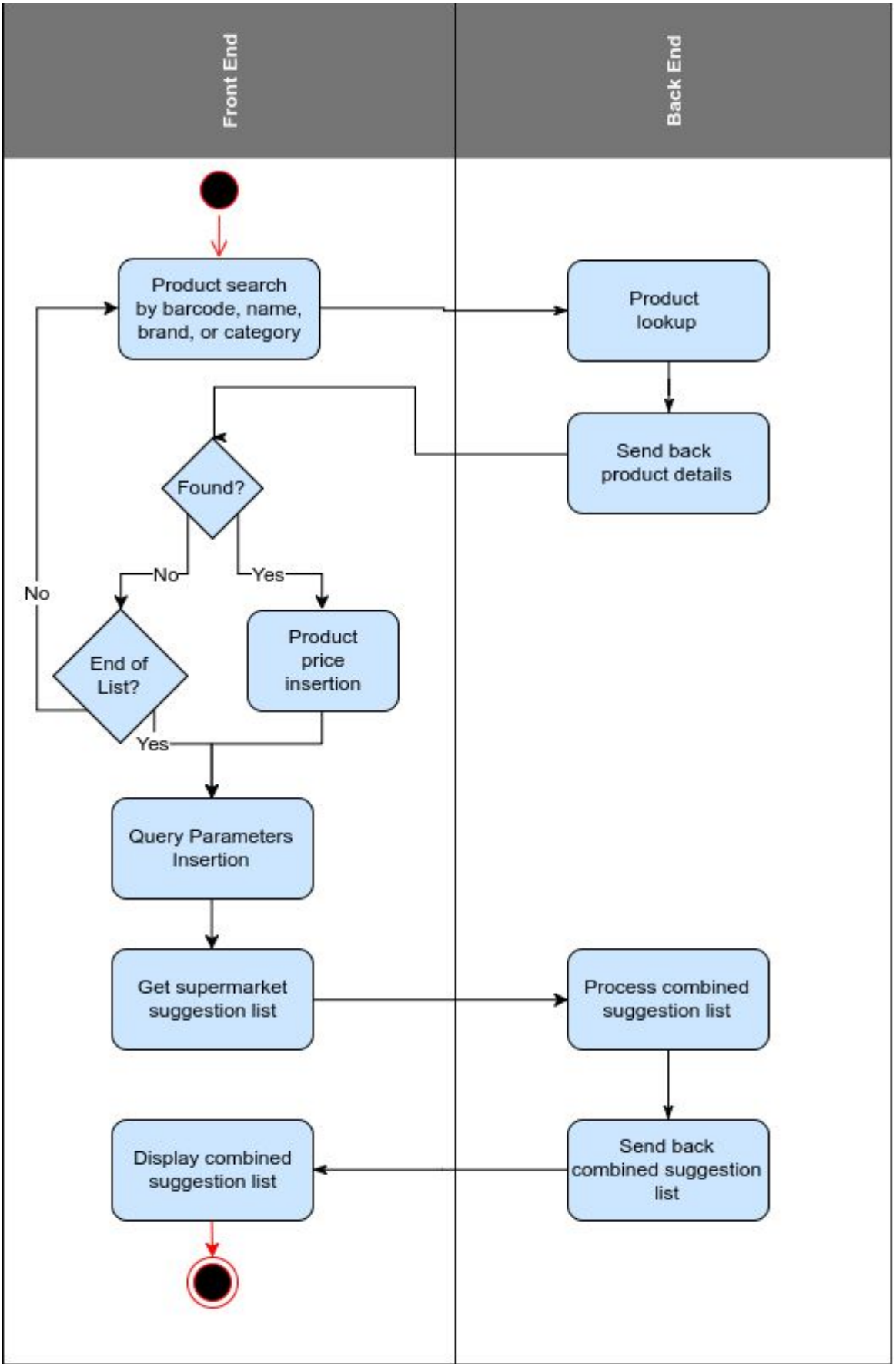

Fig.7: Activity flow for the premium supermarket suggestion feature