TED UNIVERSITY

CMPE 491

SENIOR PROJECT

Project Analysis Report

- Dilara Yargıcı - 20290006896
- Berna Tanrıverdi - 63295214002
- Aslı Dölek - 67915217862
- Yaşar Mert Dirican - 13630073160

Table of Contents

VERSION	DATE	AUTHOR	DESCRIPTION OF CHANGES
1.0	13.11.2025	Dilara Yargıcı	General Layout
1.1	17.11.2025	Berna Tanrıverdi	Nonfunctional Requirements
1.2	23.11.2025	Aslı Dölek	Diagram Initials
1.3	26.11.2025	Dilara Yargıcı	Final Draft

1.Introduction

Turkey's fashion retail sector is a highly dynamic and competitive industry that requires continuous price updates due to high economic volatility. One of the most critical operational processes in this sector is reflecting these price changes to thousands of products quickly, accurately, and simultaneously.

As the number of manual updates increases, store employees face important challenges while changing, printing, and placing these labels one by one by hand. The current manual process, besides being highly prone to errors, steals time from value-adding activities that employees should be focusing on, such as customer service and sales.

In today’s world, with the development of the Internet of Things (IoT) and low-power display (E-Ink) technologies, our "PriceLink" system can be one of the most important solutions in this field. This technology has many benefits that show why retailers should use it. For example, it instantly increases productivity. If store staff use this system, they can update all prices centrally in seconds and thus work much more efficiently. It also significantly reduces operational costs . With this technology, companies can permanently save on constant paper, plastic, and toner costs, which eliminates the waste from single-use tags . Other than these, it supports environmental sustainability (SDG 12) , reduces staff workload, and also saves time.

On the other hand, without this technology, companies need more human resources (staff) for working during price change periods or spend more time and money on these processes; which directly results in lower productivity and profitability.

Therefore, considering all these impacts, we decided to develop a system for our project that will fill this specific "technology-market gap". The main goal is to create a full-stack IoT ecosystem based on miniature E-Ink tags specifically designed for the apparel sector. So, basically, our project will help store operations teams to apply the prices previously set in central systems to thousands of products in the physical store instantly and accurately.

This Analysis Report constitutes the official technical foundation of the PriceLink project. The primary purpose of this document is to transform the high-level vision presented in the project specification into a verifiable, complete, and consistent system model for development. This report analyzes the problem domain, formalizes the functional and non-functional requirements, defines the system's boundary conditions, and thus serves as a clear technical agreement between the development team and the project stakeholders.

2.Current System

2.1.Overview

In the fashion and retail sector, product price management still heavily relies on manual processes. Store employees are required to replace paper or plastic tags one by one for every price update. Due to frequent price changes caused by economic fluctuations in Türkiye, this manual workflow becomes even more problematic.

Current Workflow

- The central management determines the price changes.
- This information is sent to store personnel through e-mail, messaging groups, or Excel sheets.
- Personnel locate the existing tags on all products, remove them, and attach new ones.
- Each product variation (size/color) requires a separate tag replacement.
- When the campaign ends, the process is repeated to revert the prices back to their original state.

Pain Points of the Current Shelf Tag System

Operational Problems

- High workload
- Long duration of tag replacement
- Delays in campaign execution timing

Accuracy Issues

- Human-caused mispricing
- Mismatch between POS price and shelf tag price
- Customer complaints and legal risks

Environmental Issues

- Waste generated from single-use paper and plastic tags
- Excessive waste due to frequent price changes in the fashion sector

3. Proposed System

3.1 Overview

PriceLink is a fully integrated IoT system designed specifically for the fashion retail sector. It consists of lightweight, compact, wearable electronic price tags and a cloud-based price management platform.

The system consists of three core components:

1. **PriceLink Tag Hardware**

- E-Ink display
- Ultra-low power consumption
- Wi-Fi/BLE communication
- Multi-year battery life
- Compact design

2. **PriceLink Cloud Platform (Backend)**
- Centralized product–price database
- Real-time price update broadcasting
- Tag health and status monitoring

3. **PriceLink Management Panel (Frontend)**
- Bulk price editing
- System analytics and monitoring
- Product–tag association tools

3.2 Functional Requirements

These are the fundamental prerequisites for the Price Link application.

User Management

- The administrator must be able to log in securely.
- The administrator must be able to add and remove products.
- Customers should only be able to view product information.

Product & Tag Management

- Product information (name, category, location, SKU) must be storable.
- A PriceLink Tag can be assigned to a product.
- Tag–product pairing must be reassignable.
- Editing and deleting product information must be supported.

Price Management

- Prices must be updatable instantly.
- Bulk price updates should be possible.
- When a campaign ends, the system should automatically revert to the original price.

Monitoring & Reporting

- Tag statuses must be displayed: Connected / Update Received / Disconnected / Low Battery
- Every action must be logged with username, timestamp, and previous–new state information.

3.3 Nonfunctional Requirements

These specifications dictate the system's speed, effectiveness, and quality. These specifications are essential to the project.

Performance

- All operations must be completed within a maximum of 10 seconds.
- Tags must have an average battery life of 4 years.

Reliability & Accuracy

- If a connection failure occurs, the previous state must be preserved to ensure operational safety.
- If more than 3 errors occur, the management panel must display a warning.
- The server must maintain 99% annual uptime.

Usability

- The system must offer high ease of use and learnability.
- The E-Ink display must be readable under normal lighting conditions.

Security

- Only authorized users may access the system.
- Data transmission must be encrypted using WPA2 or an equivalent secure protocol.

Sustainability

- The system must use E-Ink tags and reduce paper/plastic waste.

3.4 Pseudo Requirements

Pseudo requirements are external or non-technical constraints—such as branding, legal rules, or market expectations—that the system must follow but are not actual functional or nonfunctional requirements.

External Branding & Aesthetics

- The tag design must be compatible with the store’s visual aesthetics.
- Color and visual appearance should be customizable based on brand identity.

Regulatory Compliance

- Hardware must comply with CE safety standards.
- Data management must comply with Turkish KVKK regulations.

Hardware Constraints

- The battery must be safe and replaceable.
- The screen size must not occupy excessive space on clothing.

Corporate Deployment Requirements

- The system must support multi-store usage.
- The API must be extendable for future POS integration.

Environmental Expectations

- Materials must be recyclable.
- Sustainability must be prioritized during production processes.

3.5 System Models

3.5.1 Scenarios

1. **Scenario 1 – Admin Performs Bulk Campaign Price Update**

Primary Actor:

- Store Administrator (Admin)
- Stakeholders and Interests:
- Store Management: Wants campaign prices to be applied accurately and on time.
- Customers: Expect the shelf label price to match the POS price.
- Brand / Company: Aims to reduce customer complaints and legal risks caused by pricing inconsistencies.

Preconditions:

- The admin must have securely logged into the system (secure login).
- Products and product–tag pairings must already be defined in the database.

Trigger:

The headquarters defines a campaign for a specific category (e.g., “Winter Jackets”) and instructs the admin to apply this campaign using the PriceLink system.

Main Success Scenario:

- The admin logs into the PriceLink Management Panel (web interface) using secure credentials.
- The admin navigates to the “Campaign / Bulk Price Update” screen.
- The admin selects the category of products to be updated (e.g., “Coats > Winter Jackets”).
- The admin chooses the type of campaign (e.g., “30% discount” or “fixed new price”).
- The system displays a preview screen containing the selected products and their new prices.
- The admin enters the start and end times of the campaign (used for automatic price reversion).
- The admin confirms the campaign and clicks “Apply.”
- The PriceLink Cloud Platform receives the update request, performs validation, and identifies the matching product–tag pairs.
- The backend sends the new campaign prices instantly to all relevant PriceLink Tags.
- The tags display the updated prices on their E-Ink screens and return a status message indicating “Update Received.”
- The management panel displays Connected / Update Received statuses on the dashboard for each tag.
- The system logs the bulk update operation: admin username, timestamp, old prices, new prices, and campaign duration.

Alternate / Exception Flows:

A1 – Connection Failure to Some Tags:

- If some tags cannot be reached in step 9, they are marked as “Disconnected.”
- The system preserves the previous price for those tags.
- A warning appears on the dashboard and the admin is offered a “Retry update” option.

A2 – Validation Error (e.g., Campaign Expired):

- If the campaign period is invalid or parameters fail validation, the backend does not start the update.
- The management panel displays an error message.
- The operation is logged as “Failed validation.”

Postconditions:

- All reachable tags display the updated campaign price.
- After the campaign ends, prices automatically revert to their original values.
- All operation details are stored in the logs and can be retrieved when needed.

2. Scenario 2 – Admin Reassigns a PriceLink Tag to a New Product

Primary Actor:

- Store Administrator (Admin)
- Stakeholders and Interests:
- Store Staff: Want to quickly update tag–product associations during shelf changes or seasonal rotations.
- Inventory Management: Requires accurate product–tag matching for stock consistency.

Preconditions:

- The admin is logged into the system.
- The PriceLink Tag is already assigned to a product (existing pairing).

Trigger:

Due to reorganizing shelves or seasonal changes, the existing PriceLink Tag needs to be reassigned to a different product.

Main Success Scenario:

- The admin opens the "Products & Tags" screen from the PriceLink Management Panel.
- The admin searches for the Tag ID attached to a product (the ID may also be scanned via barcode/QR).
- The system displays the current product information linked to that Tag ID (SKU, name, category, location).
- The admin selects "Reassign Tag."
- The admin selects the new product from the product list or by entering its SKU.
- The system loads the new product information and displays an "Old product ↔ New product" comparison.
- The admin confirms the reassignment.
- The backend updates the tag–product mapping in the database.
- The backend sends the updated product details and current price to the tag.
- The tag updates its E-Ink screen to show the new product name, price, and other necessary information.
- The management panel displays a success message and logs the operation (admin, old product, new product, timestamp).

Alternate / Exception Flows:

B1 – Tag Offline:

- If the tag is Disconnected in step 9, the backend still updates the database.
- The tag is marked as “Pending Update.”
- When the tag comes online again, the pending update is applied automatically.

B2 – Wrong Product Selection:

- If the admin selects the wrong product, they may repeat the “Reassign Tag” action to correct it.
- All changes are stored in historical logs for traceability.

Postconditions:

- The tag now displays the new product information.
- All reassignment changes are fully traceable in the system.

3. Scenario 3 – Customer Checks Product Information Using PriceLink

Primary Actor:

- Customer
- Stakeholders and Interests:
- Customer: Wants accurate, clear, and transparent pricing and campaign information.
- Store: Seeks to reduce price complaints and increase customer satisfaction.

Preconditions:

- The related product must be paired with a PriceLink Tag.
- The tag must have successfully received the latest price and campaign update (Update Received).

Trigger:

A customer is browsing a product in the store and wants to check its price and campaign status.

Main Success Scenario:

- The customer examines the product and sees the PriceLink Tag attached to it.
- The E-Ink screen displays the product name/short description, current price, campaign price (if any), and basic product details.
- The customer sees that the shelf label price matches the POS price (ensured by the system).
- The customer can view the campaign details and duration (e.g., “20% OFF – until Sunday”).
- The customer makes a purchase decision based on the accurate information presented.

Alternate / Exception Flows:

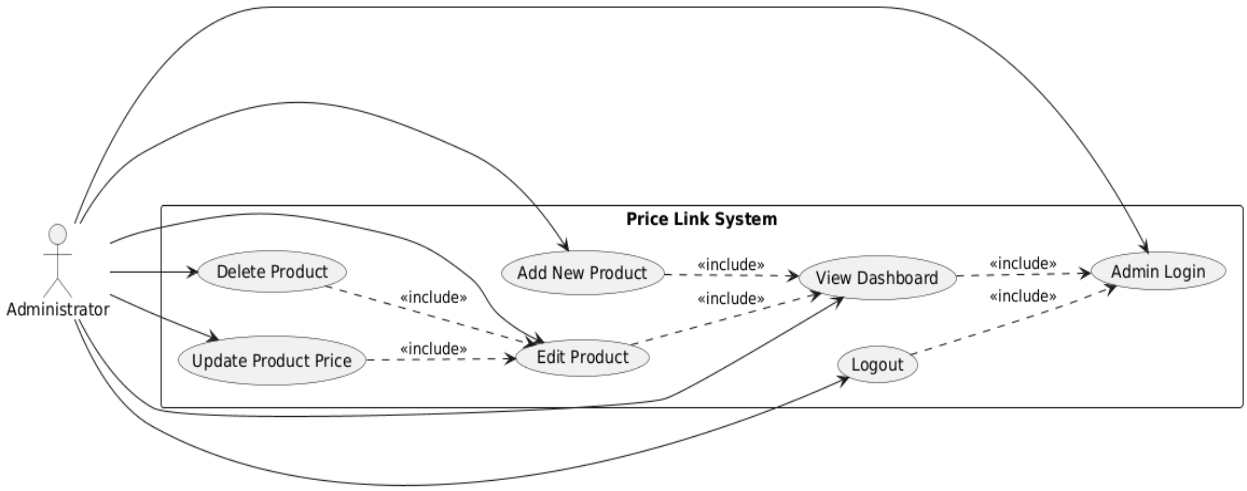
C1 – Tag Displays an Old Price:

- If the tag fails to receive the latest update due to a connection issue, the management panel marks it as “Disconnected” or “Error.”
- Store staff identify the issue via the dashboard and manually verify or trigger a “retry update.”

Postconditions:

- The customer accesses correct and readable product information.
- The system minimizes pricing discrepancies.

3.5.2 Use Case Model



1) **Table 1:**

Title	**Description**
Use Case Name	Admin Login
Actor	Administrator
Description	Logs into the system using admin user information.
Pre-conditions	the system must be in working order.
Basic Flow	1**.** The system login screen opens. 2. admin writes username and password information.. 3. Clicks on the “log in” button. 4. The system verifies the entered username and password information. 5. If the login information is correct, you will be logged into the system.
Post-conditions	The administrator logs into the system and can see the main screen.

2) **Table 2:**

Title	**Description**
Use Case Name	View Dashboard
Actor	Administrator
Description	Displays the system overview and the status of the tags (battery, connection)[3].
Pre-conditions	"Admin Login" process must be completed.(Include).
Basic Flow	1. The system retrieves current tag statuses from the database. 2. Label statuses are listed on the screen. 3. The administrator can now see the tag status on the screen.
Post-conditions	The admin can see what kind of label problems there are in the system.

3) Table 3:

Title	**Description**
Use Case Name	Add New Product
Actor	Administrator
Description	The administrator adds a new product to the system and matches it with a tag.
Pre-conditions	The Administrator should be on the "View Dashboard" screen (Include).
Basic Flow	1. Admin clicks on the "Add New Product" button. 2. The information of the product to be added (Name, Barcode, Price) is entered into the system. 3. A new e-link tag ID is assigned to the product. 4. "Save" button is pressed. 5. Added products and their information are saved in the database.
Post-conditions	A new product is added to the system.

4) Table 4:

Title	**Description**
Use Case Name	Edit Product
Actor	Administrator
Description	The product you want to edit can be selected and edits can be made.
Pre-conditions	The Administrator should be on the "View Dashboard" screen (Include).
Basic Flow	1. The admin finds and selects the product he wants to edit from the list. 2. Clicks the “edit” button. 3. The system displays the product's current information in editable form.
Post-conditions	The product has become editable and update and delete operations can be performed.

5) Table 5:

Title	Description
Use Case Name	Update Product Price
Actor	Administrator
Description	The product price is changed and reflected on the label instantly.
Pre-conditions	The product must be opened in "Edit Product" mode (Include).
Basic Flow	1. The desired price is written in the new price field. 2. “Update” button pressed. 3. Updated price is recorded in the database. 4. The system sends the signal wirelessly to the E-Ink label. 5. The label screen is updated with the new price information.(In 5 seconds).
Post-conditions	The product price is updated in the system.

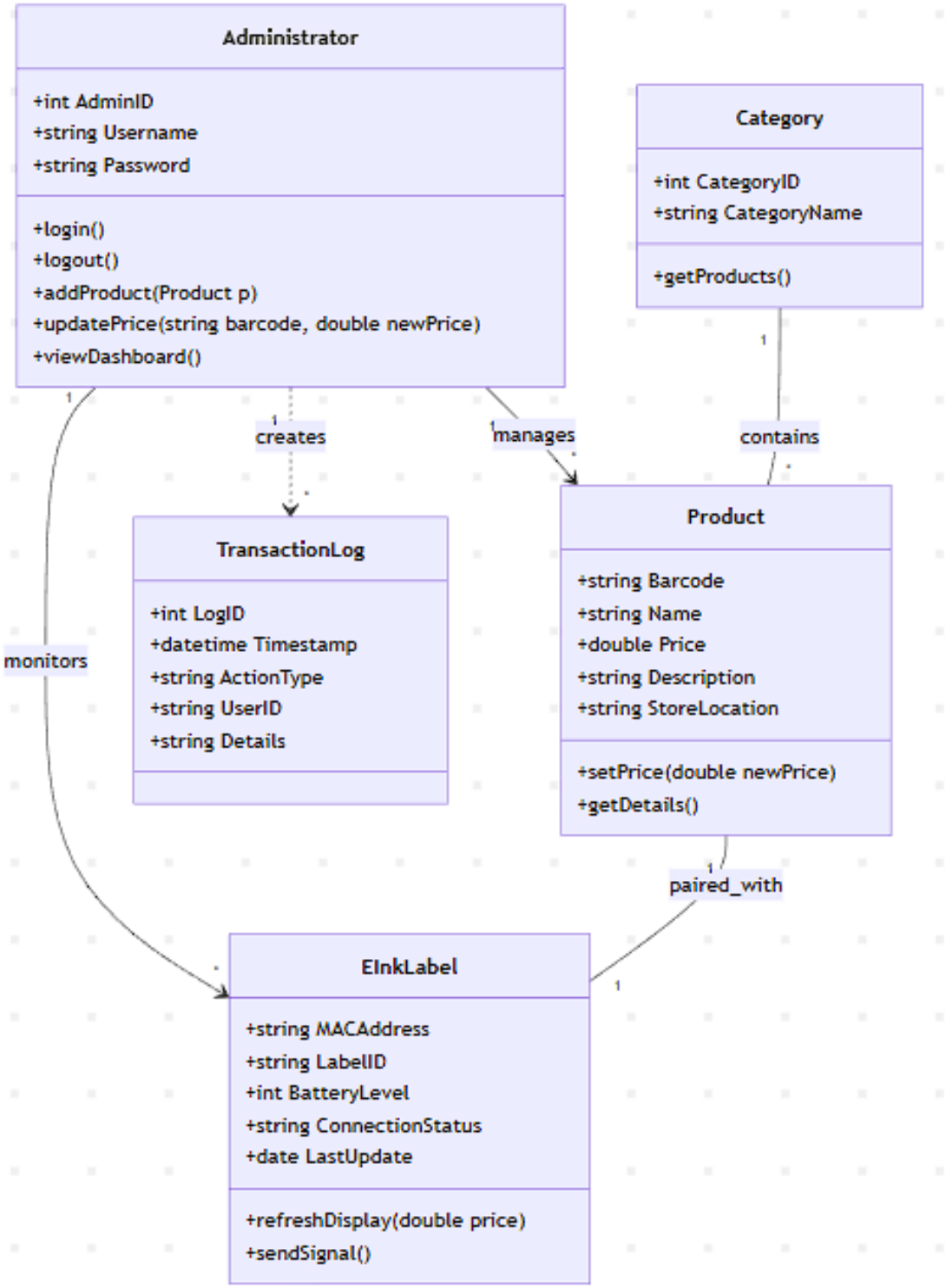
6) Table 6:

Title	**Description**
Use Case Name	Delete Product
Actor	Administrator
Description	Admin deletes the product, its information and tag match from the system.
Pre-conditions	The product must be opened in "Edit Product" mode (Include).
Basic Flow	1. Click the delete product button. 2. The system sends confirmation information. 3. Admin clicks the confirm button. 4. The system deletes the product from the database and nullifies the tag ID number.
Post-conditions	The product is deleted from the system.

7) **Table 7:**

Title	Description
Use Case Name	Logout
Actor	Administrator
Description	The administrator closes access to the administration panel and ends his active session in the system.
Pre-conditions	You must be logged into the system and be on the “View Dashboard” page.
Basic Flow	1. Admin clicks the “Log Out” button on the panel. 2. The system asks for confirmation to log out. 3. The admin approves. 4. System logs off. 5. The user is redirected back to the login page.
Post-conditions	The session ends. The log in page is returned.

3.5.3 Object And Class Model



The descriptions of the classes, methods and objects that Diagaram has are given below according to the project requirements.

1. Administrator: This class contains the privileges granted to the administrator who is allowed to access the system. It performs the operations specified in section 3.1.1 User Management.

Attributes: AdminID,Username,Password

a) **AdminID:** The system keeps the personal identification information of the person being managed.
b) **Username:** Specifies the username that the authorized person who manages the system must use to log into the system.
c) **Password:** It specifies the password that the authorized person who manages the system must use to enter the system.

Methods:

d) **login():** Checks the login information (username, password) at the time of logging into the system.

e) **updatePrice():** This is the method that will be used to update the product price according to the requirement in the 3.1.3 Price Change section.

f) **viewDashboard():** This is the method to be used to control tag statuses according to the requirement in the 3.1.4 Monitoring and Reporting section.

g) **logOut():** This is the method used when logging out of the system and terminating the session.

h) **addProduct():** This is the method used when adding a new product.

2. Product: It is an object that represents any product. It has Name, Barcode, Price, Description, and StoreLocation variables according to the requirements specified in section 3.1.2 Label and Product Management.

Attributes: Barcode, Name, Price, StoreLocation,Description

a) **Barcode:** It is a variable that holds the unique identification number of each product.
b) **Name:** It is the variable that holds the name of each product.
c) **Price:** It is the variable that holds the price of each product.
d) **StoreLocation:** It is the variable that holds the store location of each product.
e) **Description:** It is a variable that holds the description, if any, about each product.

- **Relationships:** Each product belongs to a category and is physically matched to an E-Ink label.

Methods:

f) **setPrice():** It is the method used when updating the new price information of the product and during price verification processes.
g) **getDetails():** It is the method used to access information about each product.

3. EInkLabel: Represents the physical hardware that displays the price of each product. It features e-ink technology, as per section 3.2.5 Sustainability and Constraints. This object allows you to check battery level and connection status.

Attributes: LabelID, BatteryLevel, LastUpdate, ConnectionStatus (Online/Offline), MACAddress. (**For 3.2.1 Performance and 3.1.4 Monitoring requirements**).

a)LabelID: It is a variable that contains the unique tag number of each tag.

b)BatteryLevel: It is a variable that contains the battery life of each tag.

c)ConnectionStatus: A variable that contains the connection status of each tag.

d)LastUpdate: This variable contains information about when the tag was last updated.

e) MACAddress: It is a variable that contains the private address information of each tag, which cannot be changed later.

Methods: refreshDisplay(), sendSignal().

a) **refreshDisplay():** After the new price update is made, the page is refreshed.
b) **sendSignal():** It performs the signal sending process at the required times (after the transaction is completed).

4. Category: This class allows all products to be grouped. The "bulk price update" operations specified in the 3.1.3 Price Change requirement are managed through this class.

Attributes: CategoryName, CategoryID.

a) **CatagoryName:** It is a variable that holds the name of each category.
b) **CatagoryID:** It is a variable that holds the unique identification number of each category.

Methods: getProducts():

c) **getProducts():** It is the method used to access the list of products in each category.

5. TransactionLog: It is a record class created to increase the reliability and traceability of the system. It stores detailed information such as when and by whom all transactions were performed, as required in the 3.1.4 Monitoring and Reporting section.

Attributes: Timestamp, ActionType, Details, UserID, LogID

a) **Timestamp:** It is a variable that keeps track of the time at which each operation occurs.
b) **ActionType:** It is a variable that holds the information about which category each transaction belongs to.
c) **Details:** It is the variable that keeps the status before and after the operations performed.
d) **UserID:** For each transaction, it is a variable that keeps track of who performed that transaction.
e) **LogID:** It is a variable that holds the unique identification number for each process.

3.5.4 Dynamic Model

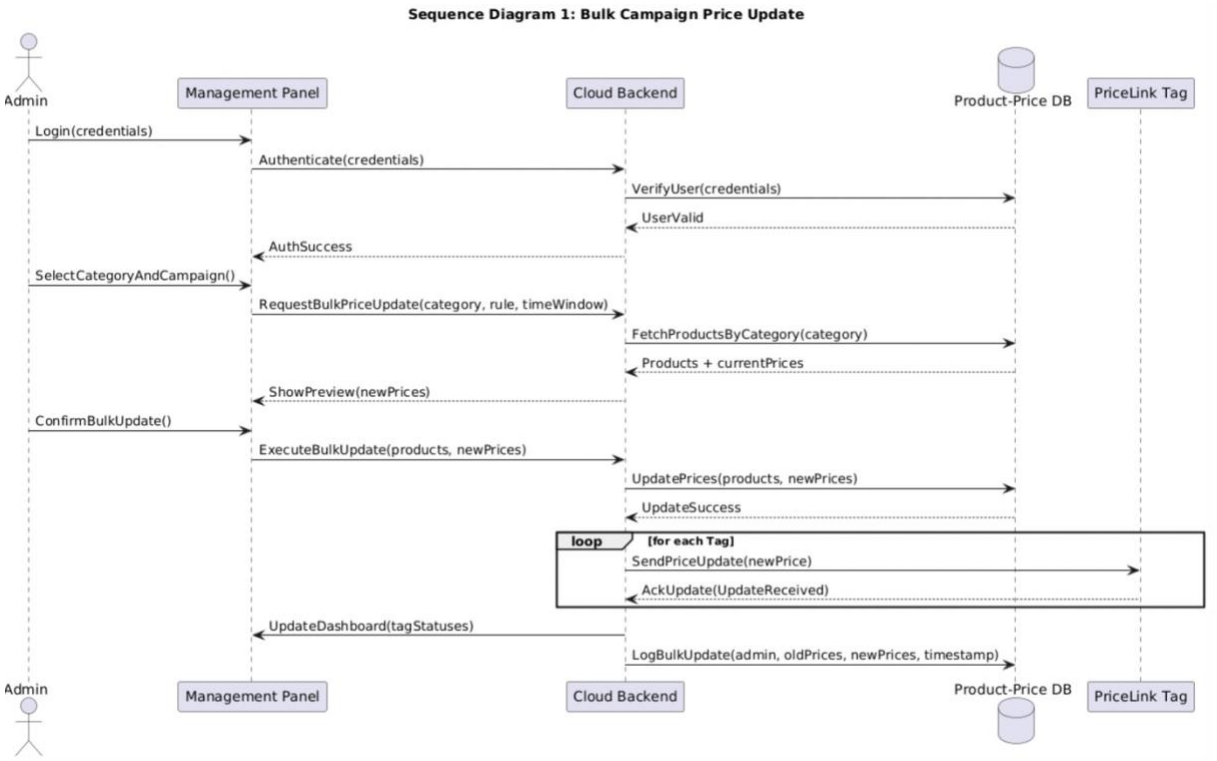
3.5.4.1 Sequence Diagram

1 – Bulk Campaign Price Update

This sequence diagram illustrates the complete interaction flow when an administrator performs a bulk campaign price update within the PriceLink system. It models the coordination between:

- Management Panel (frontend interface),
- Cloud Backend (business logic executor),
- Product–Price Database,
- and multiple PriceLink Tags.

The diagram outlines steps such as authentication, campaign selection, previewing updated prices, updating the database, distributing new prices to the tags, receiving acknowledgments from each tag, and logging the entire operation. It also reflects the system's real-time behavior and aligns with functional requirements related to bulk updates, monitoring, and logging.



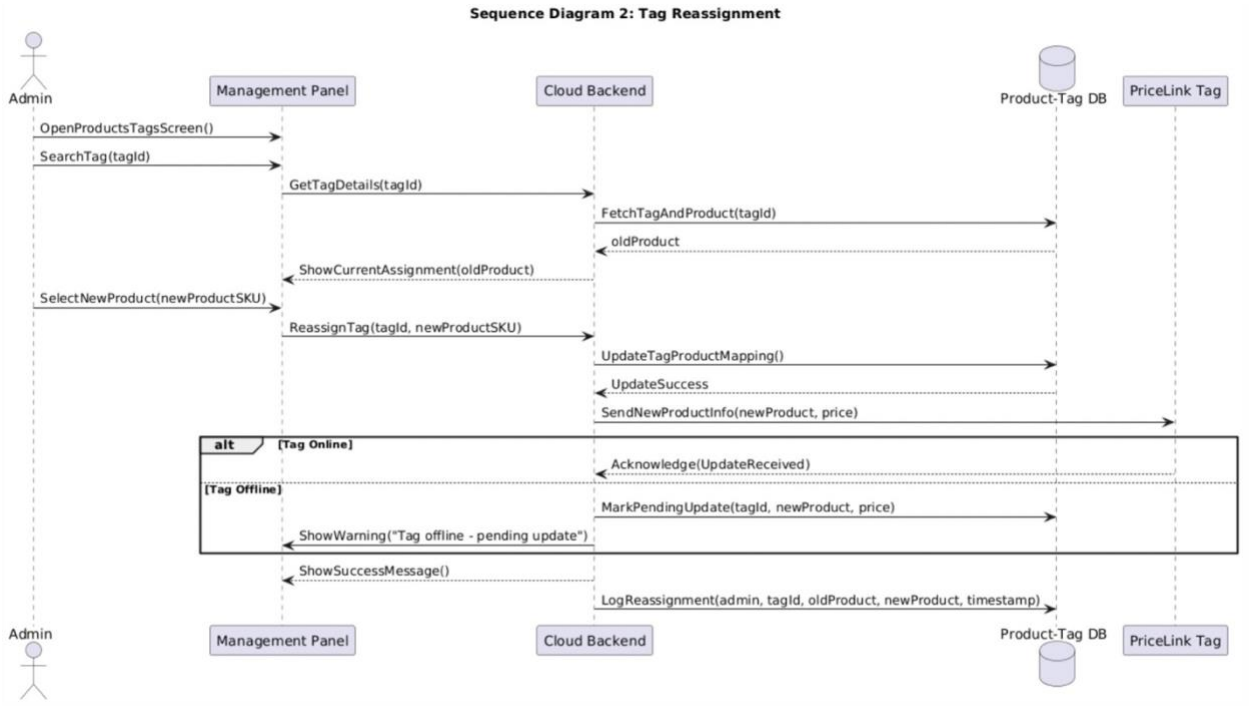
2 – Tag Reassignment

This sequence diagram represents the workflow when an administrator reassigns a PriceLink Tag to a new product. It shows how the admin retrieves the current assignment, selects a new product, and how the backend updates both the database and the tag. The diagram also handles alternative flows: if the tag is offline, the system stores the update as pending and warns the admin.

The diagram highlights interactions between:

- Management Panel,
- Cloud Backend,
- Product–Tag Database,
- PriceLink Tag.

This diagram supports functional requirements related to product–tag pairing, editing, and system monitoring.



3.5.4.2 State Diagram

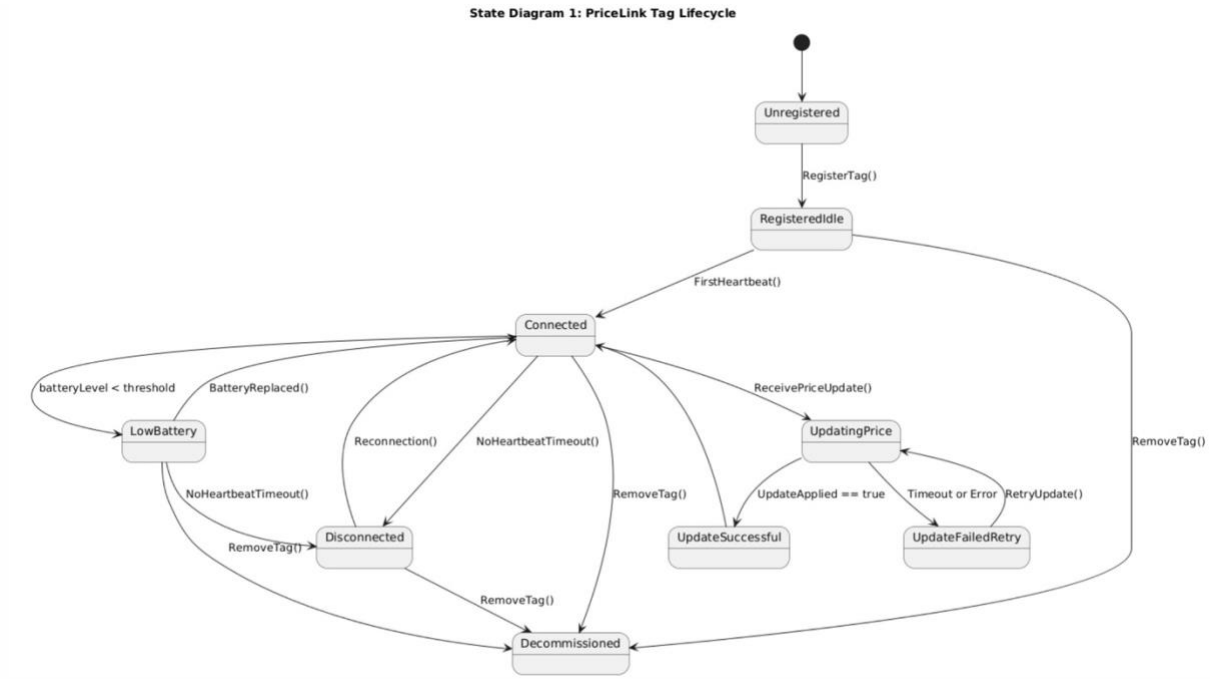
1 – PriceLink Tag Lifecycle

This state diagram describes the lifecycle of a PriceLink Tag, capturing how it behaves from the moment it is introduced into the system until it is removed.

The states reflect real-world device conditions such as:

- Registration,
- Connectivity status,
- Battery health,
- Update processes,
- Error recovery mechanisms and decommissioning.

This model aligns directly with nonfunctional requirements such as reliability, sustainability (battery life), and monitoring. It also integrates functional requirements related to tag status reporting and update acknowledgment.



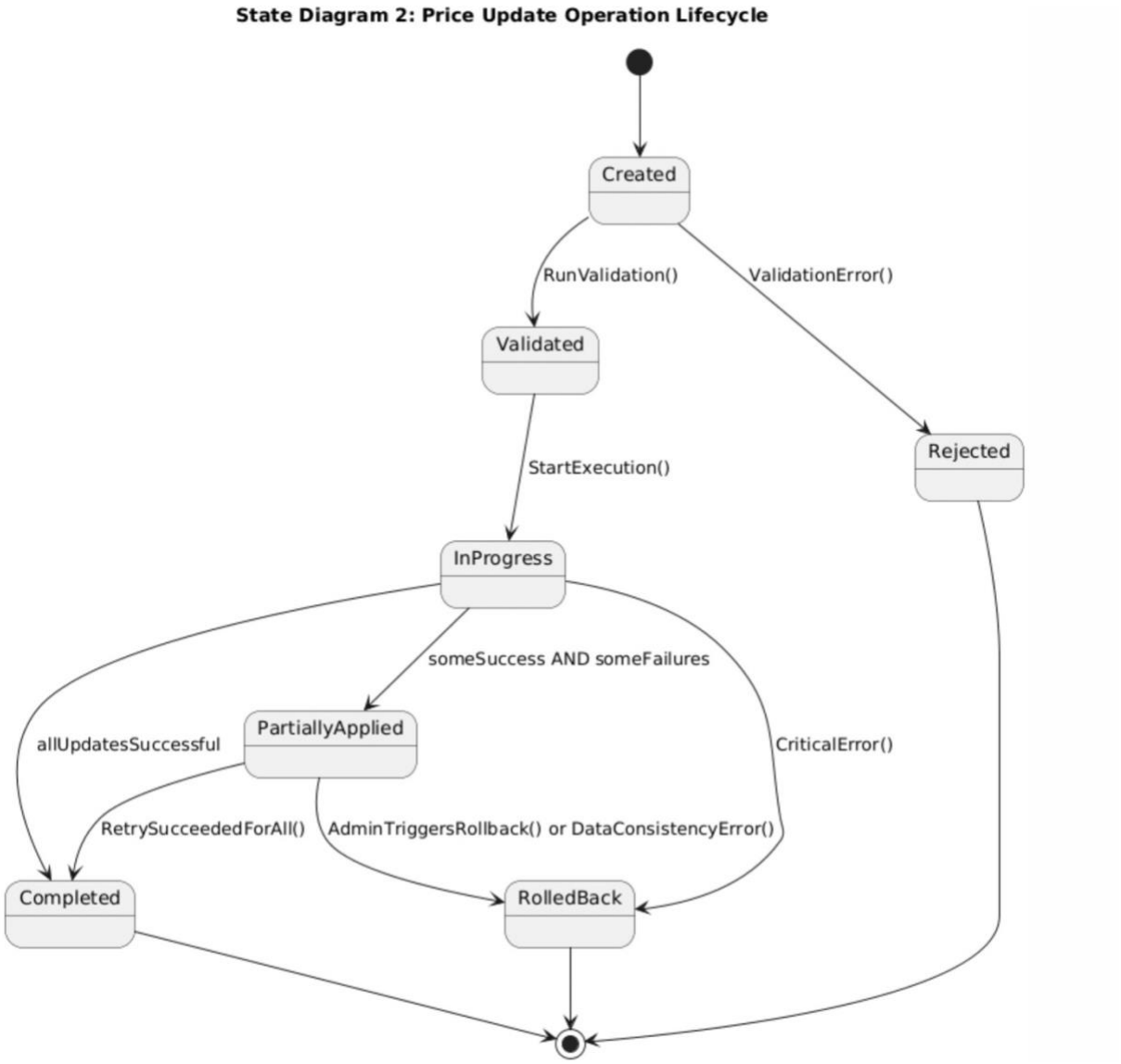
2 – Price Update Operation Lifecycle

This state diagram models the internal lifecycle of a price update operation such as a bulk campaign. It includes validation, execution, partial success, completion, and rollback behaviors.

It ensures that:

- Invalid operations are rejected early,
- Errors trigger fallback behaviors,
- Partial updates can be retried,
- Full completion is clearly identified,
- Rollback is possible in case of critical failures.

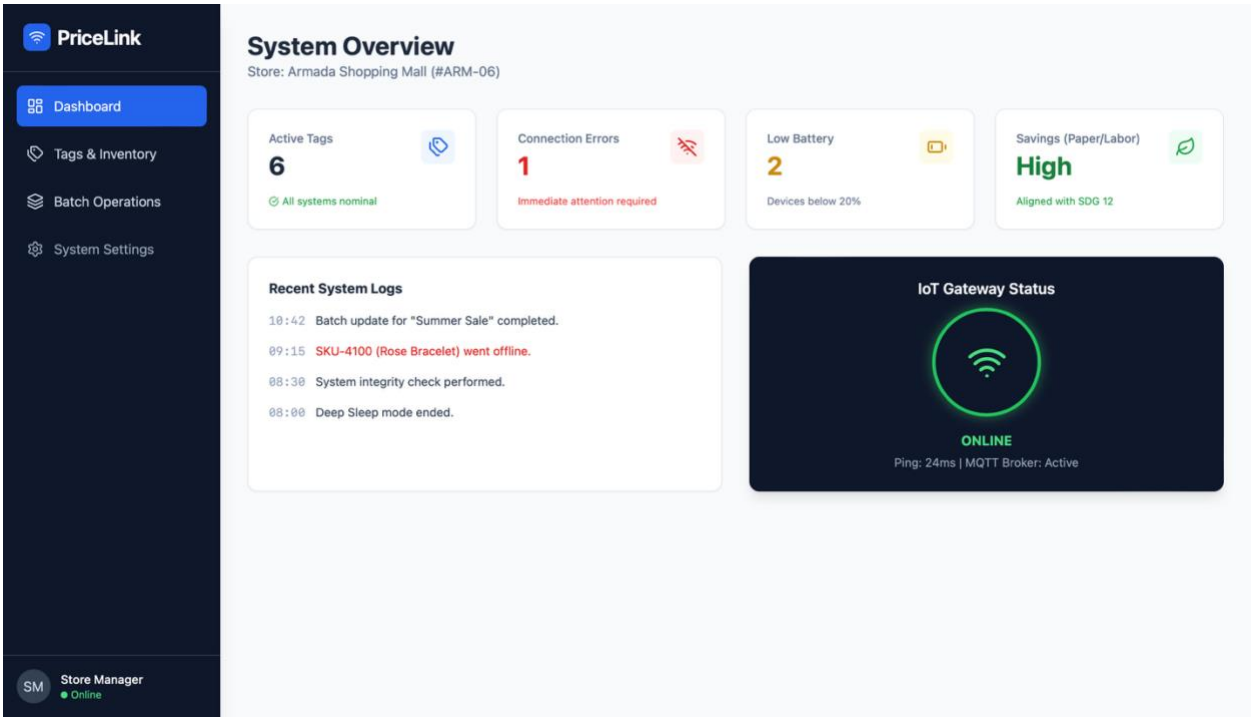
This is aligned with nonfunctional requirements such as reliability, accuracy, and error handling.

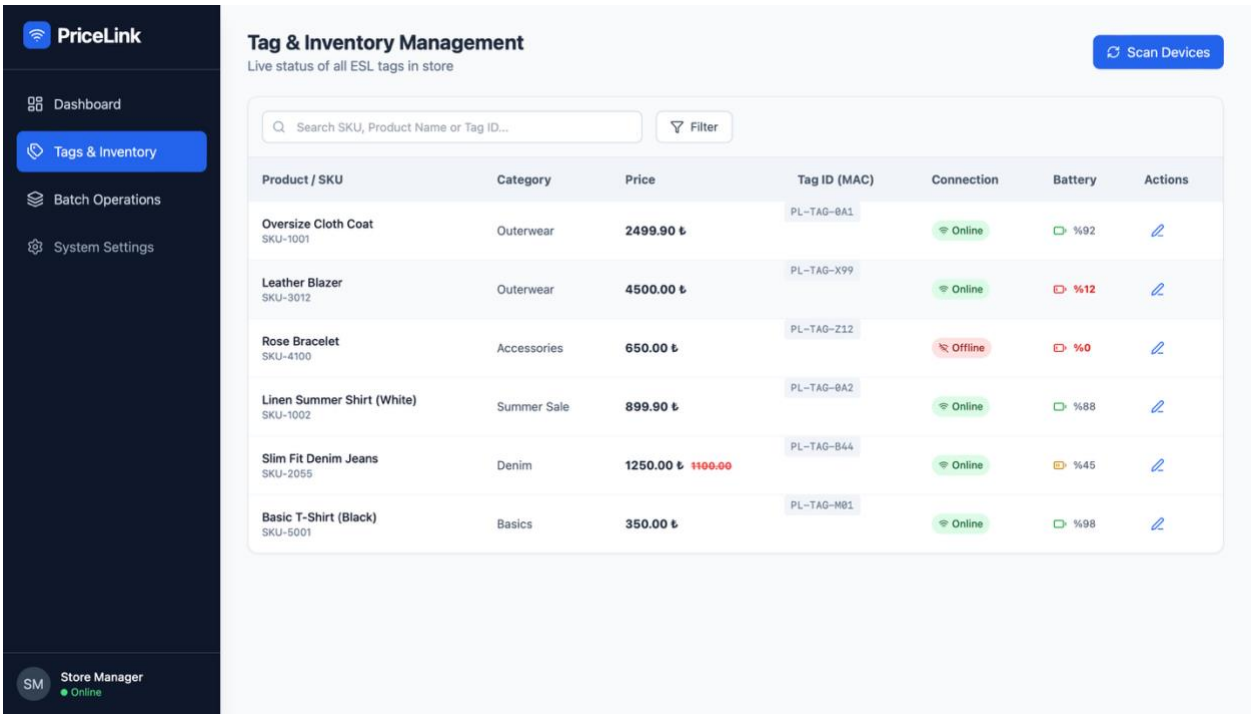


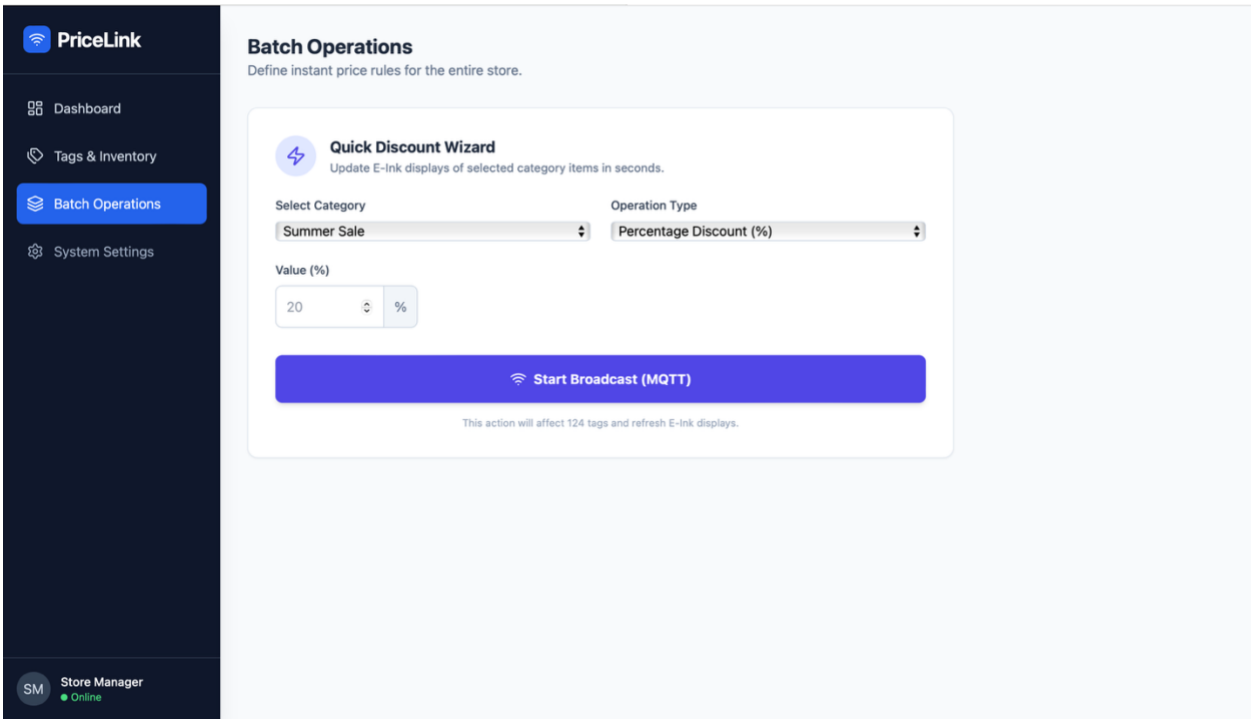
3.5.5 User Interface – Mock Up

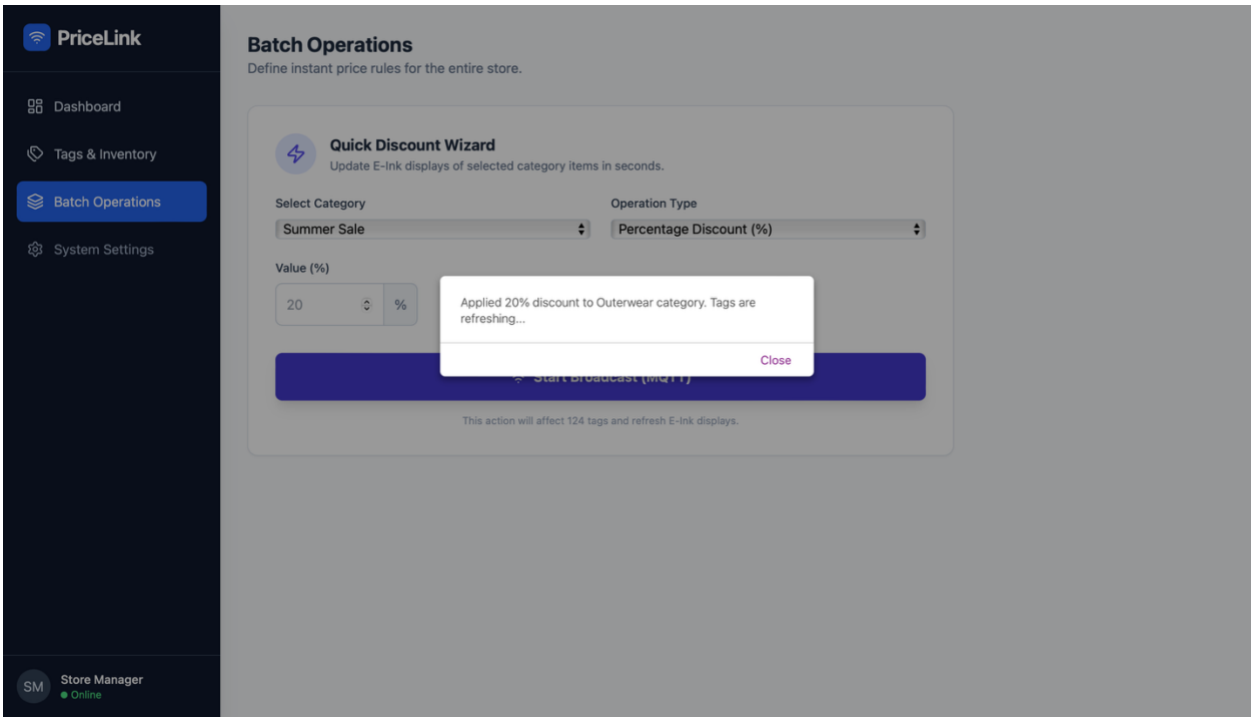
The proposed user interface is a centralized, web-based Management Panel designed for store administrators to manage the PriceLink IoT ecosystem efficiently. Unlike traditional manual labeling, this dashboard serves as the command center for the entire store, addressing the "Operational Burden" defined in the problem analysis. The interface prioritizes data visualization for system health (battery levels, connectivity status) and streamlines bulk operations to minimize labor hours. It is designed with a responsive layout to be accessible via store tablets or desktop computers.

For online check : https://pricelinkmockup.stackblitz.io


PriceLink
Dashboard
Tags & Inventory
Batch Operations
System Settings
SM
Store Manager
Online
System Overview
Store: Armada Shopping Mall (#ARM-06)
Active Tags
6
All systems nominal
Connection Errors
1
Immediate attention required
Low Battery
2
Devices below 20%
Savings (Paper/Labor)
High
Aligned with SDG 12
Recent System Logs
10:42 Batch update for "Summer Sale" completed.
09:15 SKU-4100 (Rose Bracelet) went offline.
08:30 System integrity check performed.
08:00 Deep Sleep mode ended.
IoT Gateway Status
ONLINE
Ping: 24ms | MQTT Broker: Active


PriceLink
Dashboard
Tags & Inventory
Batch Operations
System Settings
SM
Store Manager
Online
Tag & Inventory Management
Live status of all ESL tags in store
Scan Devices
Search SKU, Product Name or Tag ID...
Filter
Product / SKU
Category
Price
Tag ID (MAC)
Connection
Battery
Actions
Oversize Cloth Coat
SKU-1001
Outerwear
2499.90 ₺
PL-TAG-0A1
Online
%92
Leather Blazer
SKU-3012
Outerwear
4500.00 ₺
PL-TAG-X99
Online
%12
Rose Bracelet
SKU-4100
Accessories
650.00 ₺
PL-TAG-Z12
Offline
%0
Linen Summer Shirt (White)
SKU-1002
Summer Sale
899.90 ₺
PL-TAG-0A2
Online
%88
Slim Fit Denim Jeans
SKU-2055
Denim
1250.00 ₺ 1100.00
PL-TAG-B44
Online
%45
Basic T-Shirt (Black)
SKU-5001
Basics
350.00 ₺
PL-TAG-M01
Online
%98


PriceLink
Dashboard
Tags & Inventory
Batch Operations
System Settings
SM
Store Manager
Online
Batch Operations
Define instant price rules for the entire store.
Quick Discount Wizard
Update E-Ink displays of selected category items in seconds.
Select Category
Summer Sale
Operation Type
Percentage Discount (%)
Value (%)
20
%
Start Broadcast (MQTT)
This action will affect 124 tags and refresh E-Ink displays.


PriceLink
Dashboard
Tags & Inventory
Batch Operations
System Settings
SM
Store Manager
Online
Batch Operations
Define instant price rules for the entire store.
Quick Discount Wizard
Update E-Ink displays of selected category items in seconds.
Select Category
Summer Sale
Operation Type
Percentage Discount (%)
Value (%)
20
%
Applied 20% discount to Outerwear category. Tags are refreshing...
Close
This action will affect 124 tags and refresh E-Ink displays.

4.Glossary

- **Actor:** An entity (human user or external system) that interacts with the system to perform a task (e.g., Administrator, Customer).
- **Backend:** The part of the software system (cloud platform) that runs on the server, managing the database, logic, and communication with the tags, invisible to the user.
- **BLE (Bluetooth Low Energy):** A wireless communication technology designed for short-range communication with very low power consumption, ideal for battery-operated tags.
- **Class Diagram:** A static model that shows the system's classes, their attributes, methods, and the relationships between objects.
- **Dynamic Model:** A representation of how the system behaves over time and how objects interact with each other (includes Sequence and State diagrams).
- **E-Ink (Electronic Ink):** A display technology that mimics the appearance of ink on paper. It consumes power only when the image changes, allowing for very long battery life.
- **ESL (Electronic Shelf Label):** A digital display system used by retailers to show product prices on shelves, replacing traditional paper labels.
- **Frontend:** The visual part of the software that users interact with, such as the web-based Management Panel.
- **IoT (Internet of Things):** A network of physical objects (like PriceLink tags) that are embedded with sensors and software to connect and exchange data over the internet.
- **KVKK (Personal Data Protection Law):** The Turkish law regulating the protection of personal data, which the system complies with regarding user information.
- **MAC Address:** A unique identifier assigned to a network interface controller for use as a network address. In this project, it serves as the unique **Tag ID**.
- **Mock-up:** A visual prototype of the user interface (UI) design, showing how the application will look and function before it is fully built.
- **MQTT:** A lightweight messaging protocol used for small sensors and mobile devices, optimized for high-latency or unreliable networks.
- **POS (Point of Sale):** The place where a customer executes the payment for goods or services (e.g., the cash register system).
- **SDG (Sustainable Development Goals):** A collection of 17 interlinked global goals designed to be a "blueprint to achieve a better and more sustainable future for all". PriceLink focuses on **SDG 12** (Responsible Consumption).
- **Sequence Diagram:** A type of dynamic diagram that shows how objects interact with each other in a specific order of time to complete a process.
- **SKU (Stock Keeping Unit):** A unique code consisting of letters and numbers that identifies a specific product in inventory.
- **State Diagram:** A diagram that shows the different states an object (like a PriceLink Tag) can be in (e.g., Connected, Offline, Low Battery) and the transitions between these states.
- **Use Case:** A description of a specific interaction between a user (actor) and the system to achieve a goal (e.g., "Update Product Price").

5.References

- Cantactix. (2024, March 19). *The end of paper price tags: How digital shelf labels are transforming retail*. Cantactix Blog. Retrieved November 13, 2025, from https://cantactix.com/digitalshelflabels/
- European Commission. (2022). *CE marking – Safety and compliance rules for electronic devices*. Retrieved from https://ec.europa.eu/growth/single-market/ce-marking/
- Google. (2025). *Gemini* [Large language model]. https://gemini.google.com
- Kişisel Verileri Koruma Kurumu. (2016). *Law on the Protection of Personal Data (KVKK No. 6698)*. https://www.kvkk.gov.tr/
- Pressman, R. S., & Maxim, B. R. (2020). *Software engineering: A practitioner's approach* (9th ed.). McGraw-Hill Education.
- [PriceLink Team]. (2025). *Price Link project proposal report*. Unpublished manuscript, Department of Computer Engineering, TED University.
- SageNet. (n.d.). *Top shelf: How electronic labels help improve retail profitability*. SageNet Insights. Retrieved November 13, 2025, from https://www.sagenet.com/insights/top-shelf-how-electronic-labels-help-improve-retail-profitability/
- Slimstock. (2025, October 20). *Electronic shelf labels: A step forward in retail digitalisation*. Slimstock Blog. Retrieved November 13, 2025, from https://www.slimstock.com/blog/electronic-shelf-labels/
- Sommerville, I. (2016). *Software engineering* (10th ed.). Pearson.
- United Nations. (n.d.). *Goal 12: Ensure sustainable consumption and production patterns*. Sustainable Development Goals. Retrieved November 13, 2025, from https://www.un.org/sustainabledevelopment/sustainable-consumption-production/