Bil482 - Software Requirements Document: Wi$h Li$t

Contributors:

Akif Emre Reis
Alaaddin Gürsoy
Elif Serra Öncü
Gizem Elif Bayar

Department of Computer Engineering
TOBB University of Economics & Technology

Spring 2026

1. Project Perspective

The Wishlist and Price Tracking Application is designed as a stand-alone web-based system. It will operate as an independent platform that users can access through standard web browsers without requiring any additional desktop installation. The system is not a module of a larger enterprise application but may be integrated with external e-commerce websites for the purpose of retrieving product and pricing information.

From an architectural perspective, the application follows a client–server model. The front-end (web interface) enables users to create and manage wishlists, categorize products, assign priorities, and monitor price changes visually. The back-end is responsible for business logic, data storage, web scraping operations, scheduling price updates, and handling notifications. A database system stores user data, product details, price history, and product states.

Although the system functions independently, it interacts with external e-commerce platforms by extracting publicly available product information (such as price and image data) through web scraping or APIs where available. These integrations are loosely coupled to ensure maintainability and extensibility.

The software is designed with scalability and modularity in mind. By applying design patterns such as Factory, Strategy, Observer, and State, the system can easily accommodate new e-commerce sources, pricing strategies, notification mechanisms, and additional features without requiring major architectural changes.

As a web-based application, the system can later be extended into a mobile application or integrated into a larger e-commerce ecosystem if needed.

2. Product Functions

The main functions of the Wishlist and Price Tracking Application are summarized below:

2.1 Category Management

The system allows users to create and manage product categories. Users may select predefined categories (such as clothing, cosmetics, home, and cleaning products) or create custom categories based on their personal preferences. Categories help users organize their wishlist items in a structured and manageable way.

2.2 Product Management

Users can add products to their wishlist by providing product details such as name, category, and product link. Each product can be assigned a priority level to indicate its

importance. The system also allows users to enable automatic price tracking or manually enter a product price. Products can transition between different states such as active (wishlist), purchased, or abandoned.

2.3 Automatic Price Retrieval and Monitoring

The system automatically retrieves product prices from different e-commerce websites using web scraping techniques or APIs, depending on the platform's structure and availability. Since each website may have a different HTML structure or API format, the system applies flexible strategies to extract accurate price data.

Prices are checked periodically based on user-defined frequency settings (e.g., hourly, daily, or weekly). Every price change is stored in the system to maintain a price history, enabling users to observe trends over time.

2.4 Notification and Alert System

When a product price changes, the system notifies the user. Users can configure how frequently price checks are performed (e.g., hourly, daily, weekly) and can receive notifications accordingly. This ensures that users stay informed about price fluctuations and can make timely purchasing decisions.

3. User Characteristics

The expected users of the Wishlist and Price Tracking Application are general consumers who frequently shop online and want to monitor product prices before making purchasing decisions. The system is designed for non-technical end users and does not require advanced technical knowledge.

3.1 General Users

The primary users are individuals who shop from e-commerce websites and wish to organize products they plan to purchase. These users are expected to have:

- Basic computer literacy
- Experience using web browsers
- Familiarity with online shopping platforms
- Ability to copy and paste product links

No programming or technical expertise is required. The interface is designed to be intuitive and user-friendly so that users can easily create categories, add products, and configure price tracking preferences.

3.2 System Administrators

If the system includes an administrative panel, administrators are expected to have moderate technical skills. They may manage system configurations, monitor scraping processes, maintain integrations with external e-commerce platforms, and ensure system reliability.

Overall, the application targets everyday online shoppers and is designed to minimize complexity while providing powerful price tracking capabilities in the background.

4. Constraints

The development and operation of the Wishlist and Price Tracking Application are subject to the following constraints:

4.1 Platform Constraints

The system will be developed as a web-based application and must be accessible through modern web browsers (e.g., Chrome, Firefox, Safari, Edge). Therefore, the frontend must ensure cross-browser compatibility and responsive design principles.

The application follows a client–server architecture and requires a stable internet connection for accessing external e-commerce websites and retrieving price data.

4.2 Technology Constraints

The project must be implemented using the predefined technology stack:

- **Frontend:** React
- **Backend:** FastAPI
- **Backend-as-a-Service (BaaS):** Supabase

Because Supabase is used as the Backend-as-a-Service provider, the system depends on Supabase for:

- Database management (PostgreSQL)
- Authentication and user management
- Cloud storage (if needed)
- Hosting-related services

This introduces constraints such as:

- Dependency on Supabase’s service availability

- Supabase API limitations and rate limits
- Restrictions based on the selected Supabase plan

The backend logic must be compatible with FastAPI and follow RESTful API design principles.

4.3 External System Constraints

The system retrieves product price data using web scraping or APIs. Therefore:

- Changes in the HTML structure of e-commerce websites may break scraping functionality.
- Some websites may restrict automated data extraction.
- API rate limits or anti-bot mechanisms may limit price-check frequency.

These external dependencies are beyond the direct control of the system.

4.4 Performance and Scheduling Constraints

Since users can configure price check frequency (hourly, daily, weekly), the system must manage background tasks efficiently to avoid excessive server load.

Frequent price checks may be limited due to:

- Server resource limitations
- Supabase usage limits
- Anti-scraping protections from external platforms

4.5 Legal and Ethical Constraints

The system will only retrieve publicly available product information. It will not collect personal user data from third-party e-commerce websites.

All scraping activities must comply with the terms of service of the respective platforms.

4.6 Project Deadline Constraint

As part of a course project, the system must be completed within the academic semester timeline. Therefore, the scope of advanced features may be limited due to time constraints.

5. System Features

5.1 Category Creation and Management

Description

This feature allows users to create and manage product categories in order to organize wishlist items efficiently.

Actors

User

Functional Requirements

- The system shall allow users to create a new custom category.
- The system shall allow users to select predefined categories (e.g., clothing, cosmetics, home, cleaning products).
- The system shall allow users to edit or delete previously created custom categories.
- The system shall associate products with a selected category.
- The system shall display products grouped under their respective categories.

5.2 Product Addition and Management

Description

This feature allows users to add products to their wishlist and configure product-related settings such as priority and price tracking.

Actors

User

Functional Requirements

- The system shall allow users to add a product by providing at least a product name and product link.
- The system shall allow users to assign a category to the product.
- The system shall allow users to assign a priority level to each product.

- The system shall allow users to enable automatic price tracking or manually enter a price.
- The system shall store product details in the database using Supabase.
- The system shall allow products to transition between states (wishlist, purchased, abandoned).
- The system shall archive purchased or abandoned products.

5.3 Automatic Price Monitoring

Description

This feature enables the system to automatically retrieve and update product prices from external e-commerce websites.

Actors

User (indirectly), System

Functional Requirements

- The system shall retrieve product price information using web scraping or API integration, depending on the e-commerce platform.
- The system shall support different extraction strategies for different website structures.
- The system shall store price history for each product.
- The system shall update product prices based on the user-defined frequency (hourly, daily, weekly).
- The system shall handle failures in scraping or API requests gracefully and log errors.

5.4 Notification and Frequency Configuration

Description

This feature allows users to configure how often price checks occur and receive notifications when price changes are detected.

Actors

User

Functional Requirements

- The system shall allow users to select price-check frequency (e.g., hourly, daily, weekly).
- The system shall notify users when a price change occurs.
- The system shall trigger notifications after detecting a price update.
- The system shall support configurable notification logic (e.g., notify on any change).
- The system shall ensure notifications are sent without significantly affecting system performance.

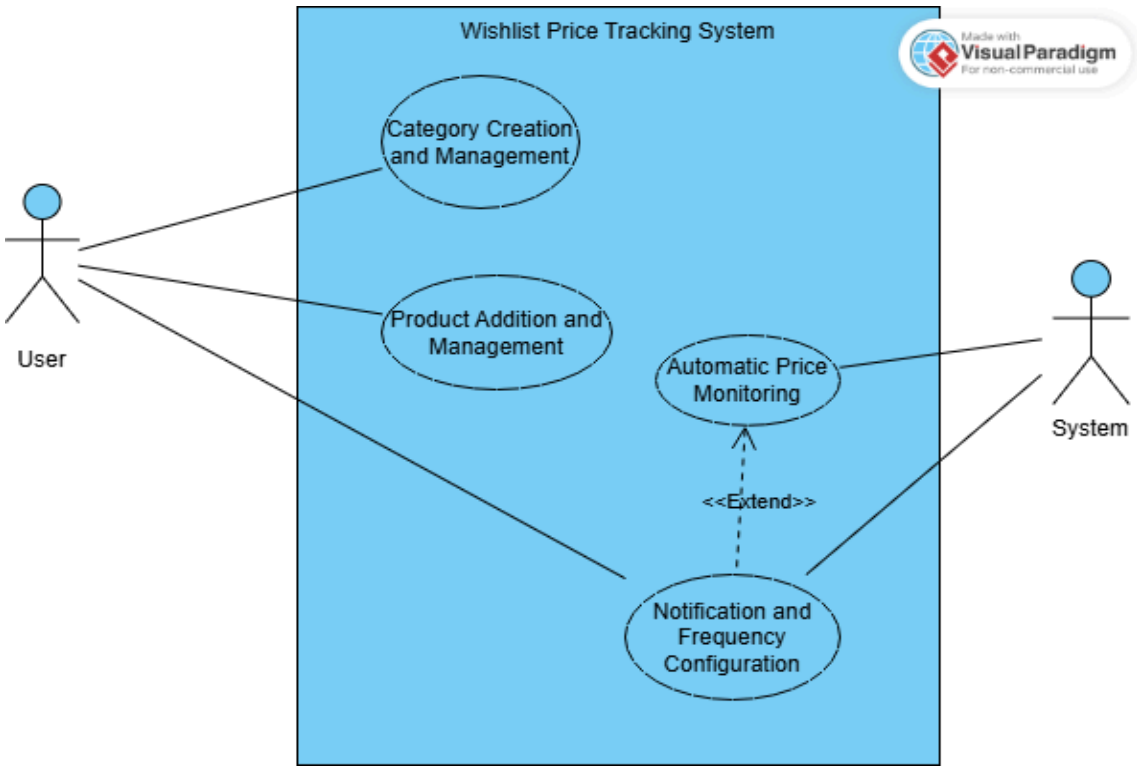
6. Use Case Diagram

The use case diagram represents the functional interactions between the User, the System and the Wishlist and Price Tracking Application. The system is primarily user-driven, allowing users to manage categories, add and manage products, and configure price tracking preferences. In addition, the system performs background operations such as automatic price monitoring and notification delivery.

The diagram illustrates the following four main use cases, each corresponding directly to the system features defined in Section 5:

- Category Creation and Management
- Product Addition and Management
- Automatic Price Monitoring
- Notification and Frequency Configuration

The User initiates actions related to category management, product management, and configuration of notification preferences. The System actor represents background services responsible for periodically checking prices, detecting changes, and sending notifications. The Notification and Frequency Configuration use case extends Automatic Price Monitoring, since notifications are only triggered conditionally when a price change is detected and notification settings are enabled.



6.1 Use Case 1: Category Creation and Management

Actors: User

Description:
This use case allows users to create, modify, and manage product categories in order to organize wishlist items efficiently.

Main Flow:

1. The user accesses the category management interface.
2. The user creates a new custom category or selects from predefined categories.
3. The user edits or deletes previously created custom categories if needed.
4. The system associates products with the selected categories.
5. The system displays products grouped under their respective categories.
6.

Exceptions:

- If a category name already exists, the system prevents duplication and prompts the user to choose a different name.

6.2 Use Case 2: Product Addition and Management

Actors: User

Description:
This use case allows users to add products to their wishlist and manage product-related settings such as category, priority, price tracking method, and product state.

Main Flow:

1. The user opens the product addition interface.
2. The user enters the product name and product link.
3. The user assigns a category and priority level to the product.
4. The user selects automatic price tracking or manual price entry.
5. The system stores the product details in the Supabase database.
6. The product is added to the active wishlist.

Exceptions:

- If automatic price retrieval fails, the system allows the user to manually enter the product price.
- Products marked as purchased or abandoned are archived and removed from the active wishlist.

6.3 Use Case 3: Automatic Price Monitoring

Actors: System (primary), User (indirect)

Description:
This use case represents the system's background process for periodically retrieving and updating product prices from external e-commerce platforms.

Main Flow:

1. The system triggers price checks based on the user-defined frequency (hourly, daily, or weekly).
2. The system retrieves product prices using web scraping or API integration.
3. The system applies appropriate extraction strategies depending on the website structure.
4. The retrieved price is compared with the previously stored price.
5. The system stores the updated price in the price history.

Exceptions:

- If price retrieval fails, the system logs the error and retries at the next scheduled interval.

6.4 Use Case 4: Notification and Frequency Configuration

Actors: User, System

Description:
This use case allows users to configure price-check frequency and receive notifications when a price change is detected. Notification delivery is handled automatically by the system.

Main Flow:

1. The user selects the desired price-check frequency (hourly, daily, or weekly).
2. The user enables or disables price change notifications.
3. The system monitors product prices through the Automatic Price Monitoring use case.
4. When a price change is detected, the system triggers a notification.
5. The system delivers the notification to the user without affecting overall system performance.

Relationship:

- This use case **extends Automatic Price Monitoring**, since notifications are sent only when a price change is detected and notification settings are enabled.

7. Non-Functional Requirements

7.1 Usability

The system shall be designed to be intuitive and user-friendly for non-technical users.

- The user interface shall be simple, clear, and easy to navigate.
- Users shall be able to create categories and add products without requiring technical knowledge.
- The system shall minimize the number of steps required to add a new product.
- The interface shall provide clear feedback for actions such as successful product addition, price updates, or errors.
- The system shall use consistent design principles across all pages.

7.2 Performance

The system shall operate efficiently under normal usage conditions.

- API response time (FastAPI backend) shall not exceed 2 seconds under normal load.
- Price-check operations shall be scheduled efficiently to avoid unnecessary server load.
- Background tasks (e.g., scraping operations) shall not block user interface interactions.

7.3 Portability

The system shall be accessible across multiple platforms.

- The application shall run on modern web browsers (Chrome, Firefox, Safari, Edge).
- The frontend (React) shall support cross-browser compatibility.
- The backend (FastAPI) shall be deployable on cloud environments compatible with Supabase.
- The system shall not require installation of additional software on the user’s device.

7.4 Reliability

The system shall be stable and provide consistent service.

- The system shall handle scraping or API failures gracefully without crashing.
- If a price retrieval operation fails, the system shall log the error and retry at the next scheduled interval.

- The system shall ensure data consistency in product and price history records.
- The system shall prevent data loss by securely storing all user data in Supabase.
- The system shall maintain consistent behavior across sessions.

8. External Interface Requirements

8.1 User Interface

The system shall provide a web-based user interface developed using React.
Users shall interact with the system through standard web browsers (e.g., Chrome, Firefox, Safari, Edge).

The interface shall allow users to:

- Create and manage categories
- Add and manage products
- Configure price tracking frequency
- View price history and notifications

The interface shall follow responsive design principles.

8.2 Software Interfaces

The system interacts with the following external software components:

Supabase

- Used for database management (PostgreSQL)
- Provides authentication and user management
- Stores product, category, and price history data

External E-Commerce Websites

- The system retrieves product price and image information via web scraping or APIs
- Integration depends on publicly accessible product pages

REST API (FastAPI Backend)

- The frontend communicates with the backend through RESTful API endpoints
- JSON is used as the data exchange format

8.3 Communication Interfaces

- Communication between frontend and backend shall occur over HTTPS.
- External API requests and scraping operations shall use secure HTTP protocols.

9. User Interfaces

The system provides a web-based user interface developed using React. The main screens and user interactions are described below:

9.1 Authentication Screen

- Allows users to register and log in.
- Users provide email and password credentials.
- Authentication is managed through Supabase.
- Displays validation messages for incorrect or missing input.

9.2 Dashboard (Main Wishlist Screen)

- Displays all active categories and products.
- Products are grouped by category.
- Each product shows:
    - Product name
    - Current price
    - Priority level
    - Price change indicator (if applicable)
- Users can navigate to product details or add new products from this screen.

9.3 Category Management Interface

- Allows users to create new custom categories.
- Allows editing or deleting existing custom categories.
- Displays predefined categories for selection.

9.4 Product Addition Interface

- Allows users to add a product by entering:
    - Product name
    - Product link
    - Category
    - Priority level
    - Price tracking preference (manual or automatic)
- Allows users to configure price-check frequency (hourly, daily, weekly).

9.5 Product Detail Screen

- Displays detailed product information.
- Shows price history in visual format (e.g., chart or timeline).
- Allows users to:
    - Change priority
    - Modify tracking settings
    - Mark as purchased

    - Mark as abandoned

9.6 Notification Interface

- Displays recent price change notifications.
- Shows previous and updated price information.

9.7 Archived Products Screen

- Displays products marked as purchased or abandoned.
- Removes archived products from the active wishlist view.

10. Software Interfaces

The Wishlist and Price Tracking Application interacts with the following external software systems and libraries:

10.1 Supabase

The system integrates with Supabase as a Backend-as-a-Service (BaaS) provider.

Supabase is used for:

- PostgreSQL database management
- User authentication and session management
- Secure data storage
- Backend service integration

The application communicates with Supabase using its provided APIs and SDKs.

10.2 FastAPI Framework

The backend of the system is implemented using the FastAPI framework.

FastAPI is responsible for:

- Handling RESTful API endpoints
- Managing business logic
- Processing price retrieval operations
- Communicating with Supabase

The frontend communicates with the FastAPI backend via HTTP requests using JSON as the data exchange format.

10.3 React Framework

The frontend is implemented using the React library.

React is responsible for:

- Rendering user interfaces
- Managing client-side state
- Communicating with the backend through REST APIs

10.4 External E-commerce Websites

The system interacts with external e-commerce platforms to retrieve product information such as:

- Current price
- Product image

This interaction occurs via:

- Web scraping techniques
- Public APIs (if available)

Since these platforms are external systems, changes in their structure may affect the price retrieval process.

10.5 Scraping Libraries

The system may use web scraping libraries (e.g., HTTP clients, HTML parsers) to extract product data from web pages. These libraries enable structured data extraction from different website layouts.

Task Matrix

Tasks	Akif Emre Reis	Alaaddin Gürsoy	Elif Serra Öncü	Gizem Elif Bayar
Project Perspective				

Product Functions				
User Characteristics				
Constraints				
System Features (Use Case Based)				
Use Case Diagram				
Non-Functional Requirements				
External Interface Requirements				

User Interfaces				
End-to-end document review and conflict resolution