# Online Store

# Software Requirements Specification

REVISION HISTORY

| VERSION | DATE | DESCRIPTION | AUTHOR |
|---|---|---|---|
| Alpha 1.0 | 22.09 | First prototype | **GRUPPA** |
| | | | |
| | | | |
| | | | |
| | | | |

# CONTENTS

# 1 SW System Overview

This SRS describes the requirements for a software system that supports product publishing, ordering, and delivery operations for a company's online store. The system will automate key tasks for both customers and administrators, including publishing product information, searching for products by parameters, placing orders with a chosen delivery date, and managing customer records.

## 1.1 Purpose

The system is made to automate the main processes of a small store. Customer can search for products, make orders, and change delivery dates if items are not in stock. Administrators can add and update products, check stock, manage customer records.

## 1.2 Scope

1. **Included:** product publishing, product search by parameters, order placement with delivery date, cash-on-delivery recording, out-of-stock handling (change order or reschedule), customer record management, discount application, simple reporting.
2. **Excluded:** online banking, online payment processing, tax reporting, multi-user, external system integration
3. **Benefits:** faster product search and order placement, reduced manual effort for administrators, improved customer satisfaction through rescheduling and discounts, better accuracy in order tracking.
4. **Key Features:** console-based interface, order rescheduling, customer record management, discount handling, delivery date scheduling, basic reporting.

## 1.3 General Constraints

- Language: C++
- OS: Unix-based
- Performance: 5 secs for operations on products (change in listings)

## 1.4 Assumptions and Dependencies

- The store has stable access to electricity and a local PC.
- The administrator manually adds new products  into the system.
- The system depends on the local file system being accessible for storing products, orders, and deliveries.
- No internet connection or external APIs are required.

## 1.5 Acronyms and Abbreviations

| Terms Used | Description of terms |
|---|---|
| **SW** | Software |
| **SRS** | Software Requirements Specifications |
| **UML** | Unified Modeling Language |
| **KGB** | Komitet Gosudarstvennoi Bezopasnosti |
| **OS** | Operating System |
| **CRC** | Class–Responsibility–Collaboration |
| **STL** | Standard Template Library |
| **CSV** | Comma Separated Values |
| **API** | Application Programming Interface |

# 2 SW Functional Requirements

## *2.1 Features / Functions to be Implemented*

All functional requirements should be derived from User Stories or Use Cases.

## *2.2 User Stories and Acceptance Criteria*

**1). As a Customer**, I want to search for a product by name or price range so that I can quickly find what I need.

**Acceptance Criteria:**

- Given the product list is available, when I enter a search keyword or price range, then I should see a list of products matching my criteria within 2 seconds.
- The system should display product name, description, price, and availability status.

**2). As a Customer**, I want to place an order with a chosen delivery date so that I can receive products on a convenient day.

**Acceptance Criteria:**

- Given I have selected one or more products, when I choose a delivery date and confirm, then the system should create an order with the selected date stored.
- The order confirmation should display product details, quantity, total price, and delivery date.

**3). As a Customer**, I want to be notified if an item is out of stock so that I can decide whether to change my order or reschedule delivery.

**Acceptance Criteria:**

- Given a selected product is out of stock, when I try to place an order, then the system should display a message about its unavailability.
- The system should allow me to either remove the product from the order or choose a new delivery date.

**4). As a System Administrator**, I want to publish product information so that customers can view and order products.

**Acceptance Criteria:**

- Given I am logged in as an administrator, when I add product details (name, description, price, stock), then the product should appear in the product list for customers.
- Changes (add, update, delete) should reflect immediately in the system.

**5). As a System Administrator**, I want to manage customer discount eligibility records.

**Acceptance Criteria:**

- Given customer records exist, I should be able to edit their discount eligibility if I need to add special offers.
- New customers should be automatically added when they place their first order.

## *2.1 Implementation Requirements*

- All products must be stored in a CSV file with ID, product name, description, price, availability info.

- All orders must be stored in a CSV file with product ID, delivery date, delivery status.
- The program must work in console mode (CLI) only.
- UML diagrams must be delivered for use cases, classes, and sequence flows

# 3 SW Non-Functional Requirements

## 3.1 Resource Consumption

Resource Consumption
- Response time for exchange operation: ≤ 2 seconds
- Maximum memory usage: ≤ 100 MB
- Maximum file size for daily logs: ≤ 5 MB

## 3.2 License Issues

License Issues
- Only standard C++ STL libraries are allowed.
- No proprietary third-party libraries are permitted.
- External libraries may only be used if they have permissive open-source licenses (MIT, Apache2.0).

## 3.3 Coding Standard

Coding Standard
- Each function and class must include descriptive comments.
- Unit tests must cover all critical components (e.g., calculation of exchanged amount).

## 3.4 Modular Design

Modular Design
- The system shall consist of separate modules for:
• Exchange calculation
• File logging
• Reporting
• User interaction
- Modules must be designed for low coupling and high cohesion.

## 3.5 Reliability

Reliability
- The system must reject invalid input without crashing.
- File writes must be atomic to avoid corruption.
- Error messages must be logged in a text file for troubleshooting.

## 3.6 Portability

Portability
- The system must compile and run on Windows 10+ and Ubuntu Linux.
- Identical inputs must produce identical outputs on both platforms.

## 3.7 General Operational Guidelines

General Operational Guidelines
- The system must be robust, easy to maintain, and simple to use.
- Daily reset functionality must be provided to start each workday with a clean state.
- All operations must be logged for accountability and auditing purposes.

# 4 SW Design Artifacts

## 4.1 CRC Cards (Class–Responsibility–Collaboration)

1. **Product**
   **Responsibilities**: store name/price/stock, check availability, update after sales, support search.
   **Collaborators**: Administrator, Order.

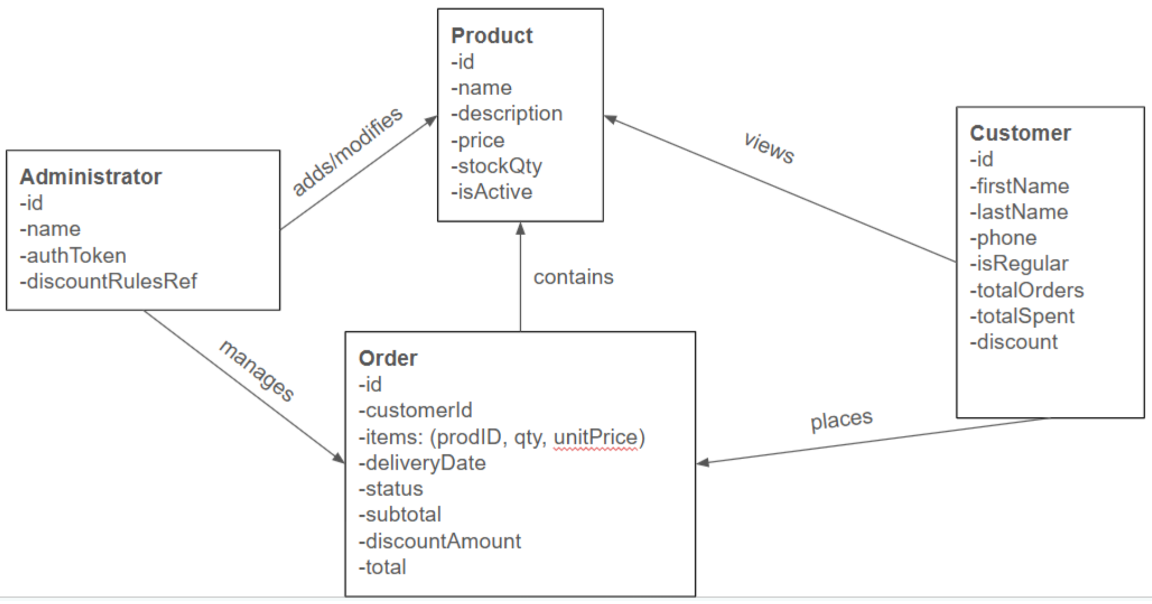
2. **Customer**
   **Responsibilities**: hold personal data, track discount eligibility, request reschedules, view orders.
   **Collaborators**: Order, Delivery, Administrator.

3. **Order**
   **Responsibilities**: keep product list, total, status; reserve stock; apply discounts; manage reschedules.
   **Collaborators**: Customer, Product, Delivery, Administrator.

4. **Administrator**
   **Responsibilities**: manage products, customers, and discounts; oversee orders and reporting.
   **Collaborators**: Product, Customer, Order, Delivery.

5. **Delivery**
   **Responsibilities**: schedule/reschedule deliveries, update delivery status, confirm COD completion.
   **Collaborators**: Order, Customer, Administrator.

## 4.2 Conceptual UML Diagram (entities & relationships)


Product
-id
-name
-description
-price
-stockQty
-isActive
Administrator
-id
-name
-authToken
-discountRulesRef
Customer
-id
-firstName
-lastName
-phone
-isRegular
-totalOrders
-totalSpent
-discount
Order
-id
-customerId
-items: (prodID, qty, unitPrice)
-deliveryDate
-status
-subtotal
-discountAmount
-total
adds/modifies
views
contains
manages
places