Product Requirements & Specification Document

Project Name

ShopList - Inventory Management API

Description

A backend API for small e-commerce shops to manage product inventory, supporting CRUD operations, pagination, and search. Built with Node.js, Express, and PostgreSQL.

1. Goals & Objectives

Goal	Description
Inventory Management	Enable CRUD operations for product inventory
Efficient Data Access	Support pagination and search for product listings
Simple Integration	Provide RESTful endpoints for easy frontend integration
Reliability & Security	Ensure data integrity and secure access

2. Stakeholders

Role	Responsibility
Shop Owners	Use API to manage inventory
Developers	Implement and maintain the API
Admins	Oversee system and data integrity

3. Functional Requirements

3.1 Product Entity

Field	Type	Constraints
id	UUID	Primary Key, auto-generated
name	String	Required, max 100 chars
description	String	Optional, max 500 chars
price	Decimal	Required, >= 0
quantity	Integer	Required, >= 0
sku	String	Unique, required, max 50

created_at	Timestamp	Auto-generated
updated_at	Timestamp	Auto-updated

3.2 API Endpoints

Method	Endpoint	Description	Auth Required
GET	/products	List products (pagination, search)	No
GET	/products/:id	Get product by ID	No
POST	/products	Create new product	Yes
PUT	/products/:id	Update product	Yes
DELETE	/products/:id	Delete product	Yes

Pagination & Search

- **Pagination:** `?page=<number>&limit=<number>`
- **Search:** `?search=<query>` (searches name and SKU)

---

4. Non-Functional Requirements

Requirement	Specification
Performance	≤ 300ms response time for standard queries
Security	Basic authentication for write operations
Scalability	Support up to 10,000 products
Documentation	OpenAPI (Swagger) spec provided
Error Handling	Consistent error responses (JSON)

---

5. Data Model (ERD)

```
Product
-------
id (PK)
name
description
price
quantity
sku (unique)
created_at
updated_at
```

---

6. Technology Stack

Layer	Technology
Language	Node.js (>=18.x)
Framework	Express (>=4.x)
Database	PostgreSQL (>=13)
Auth	Basic Auth (HTTP)
Docs	Swagger/OpenAPI

---

7. API Example

Create Product (POST /products)

Request:

```
{
  "name": "T-shirt",
  "description": "Cotton, size M",
  "price": 19.99,
  "quantity": 100,
  "sku": "TSHIRT-M-001"
}
```

Response:

```
{
  "id": "uuid",
  "name": "T-shirt",
  "description": "Cotton, size M",
  "price": 19.99,
  "quantity": 100,
  "sku": "TSHIRT-M-001",
  "created_at": "timestamp",
  "updated_at": "timestamp"
}
```

---

8. Success Metrics

Metric	Target
Uptime	≥ 99%
Response Time	≤ 300ms (average)
Error Rate	< 1% (per 1000 requests)
API Test Coverage	≥ 80%

---

9. Out of Scope

- No user management or roles beyond basic auth
- No frontend/UI
- No advanced analytics or reporting

---

10. Milestones

Milestone	Timeline
Schema & API Design	Week 1
CRUD Implementation	Week 2
Pagination & Search	Week 3
Auth & Documentation	Week 4
Testing & Deployment	Week 5

---

11. Risks & Mitigations

Risk	Mitigation
Data loss/corruption	Regular DB backups
Unauthorized access	Enforce authentication
Performance bottlenecks	Optimize queries, add indexes

---

12. Appendix

- [OpenAPI Spec] (to be delivered)
- [DB Migration Scripts] (to be delivered)
