SIMPLE BACKEND SYSTEM DESIGN

1. Introduction

A simple backend server to store and retrieve product data submitted from frontend. The server is built using Node.js with Express framework. The goal is to demonstrate handling of API requests, JWT-based authentication, in-memory storage, and structured code with Swagger documentation.

2. Scope

- Provide authentication endpoints
- Protect product endpoints with JWT authentication
- Product CRUD endpoints
- Use in-memory storage (array) for users and products
- Auto-generated API documentation using Swagger

3. Requirements

Functional Requirements:

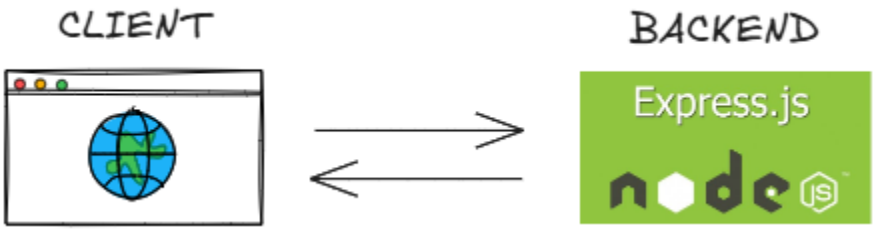
- Authentication:
  - Register new user with hashed password
  - Login returns JWT token
- Accept product data from frontend including:
  - id (UUID string)
  - name (string)
  - description (string)
  - price (number)
  - stock (number)
- CRUD operations stored in memory
- Secure product endpoints with JWT

Non-Functional Requirements:

- **Performance**: Handle up to 50 RPS for storing/retrieving products.
- **Scalability**: Designed as a single instance, no persistent database required.
- **Maintainability**: Organized using routes, controllers, and service layers.
- **Observability**: Logging via Morgan, API docs via Swagger

4. High-Level Design

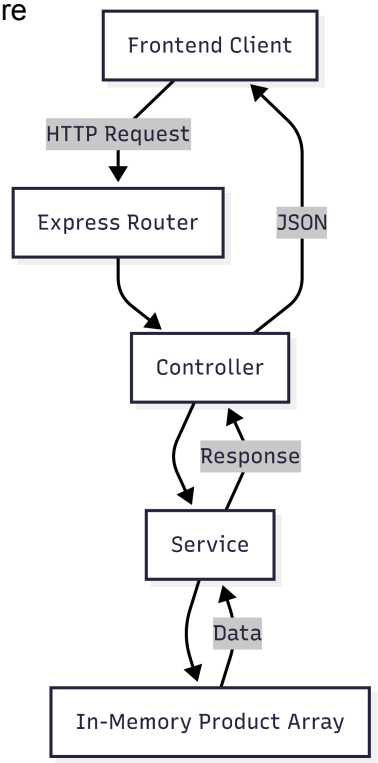
- **Client:** Sends HTTP requests with JSON body containing product data.
- **Server (Express):** Handles routes, calls service to store or retrieve products.
- **Storage:** In-memory array products[] holds all product objects.



5. Low-Level Design

Express setup:

- Middlewares: JSON parser, logger (morgan), JWT verification middleware
- Swagger UI at /api-docs



Router:

- **Auth**
  - POST /auth/register
  - POST /auth/login
- **Products**
  - POST /products
  - GET /products
  - GET /products/:id
  - PATCH /products/:id
  - DELETE /products/:id

Controller:

- Parses request body
- Calls service to store/retrieve data
- Returns JSON response

Service:

- **AuthService**: manages users, password hashing, token generation
- **ProductService**: manages CRUD operations for products[]