**Approach and Design Decisions:**

**Scalability**: The application is designed to be scalable by leveraging Flask, which is a lightweight and scalable micro-framework. It also uses MongoDB as the database, which is horizontally scalable and can handle large amounts of data.

**Security:** Security measures are implemented using Flask JWT Extended for authentication, ensuring that endpoints are protected and only accessible with valid access tokens. Passwords are not stored directly in the database; instead, only password hashes are stored, enhancing security. Additionally, MongoDB connections are secured and access is restricted.

**Reliability:** The application is built using Flask and MongoDB, which are reliable technologies known for their stability and robustness. Error handling is implemented to gracefully handle exceptions and provide meaningful error messages to users.

**Clean, Maintainable, Well-tested Code:** The code follows best practices for Python development, including modularization, separation of concerns, and adherence to PEP 8 coding standards. Unit tests can be added using frameworks like pytest to ensure code quality and maintainability. Additionally, clear documentation and comments are provided throughout the codebase to facilitate understanding and maintenance.

**Documentation for API Usage and Endpoints: Below is the documentation for example API usage and endpoints:**

**Endpoints:**

**POST /login:** Endpoint for user authentication. Requires a JSON payload containing the user's email and password. Returns access and refresh tokens upon successful authentication.

**POST /refresh:** Endpoint to refresh the access token. Requires a valid refresh token in the request payload.

**POST /products:** Endpoint to create a new product. Requires authentication. Accepts a JSON payload containing product details.

**GET /products:** Endpoint to retrieve all products. Requires authentication. Supports search, sorting, and filtering.

**GET /products/<product\_id>:** Endpoint to retrieve a specific product by ID. Requires authentication.

**PUT /products/<product\_id>:** Endpoint to update a specific product by ID. Requires authentication. Accepts a JSON payload containing updated product details.

**DELETE /products/<product\_id>:** Endpoint to delete a specific product by ID. Requires authentication.

**POST /customers**: Endpoint to create a new customer. No authentication required. Accepts a JSON payload containing customer details.

**GET /customers**: Endpoint to retrieve all customers. No authentication required. Supports search, sorting, and filtering.

**PUT /customers/<customer\_id>:** Endpoint to update a specific customer by ID. No authentication required. Accepts a JSON payload containing updated customer details.

**DELETE /customers/<customer\_id>:** Endpoint to delete a specific customer by ID. No authentication required.

**POST /orders:** Endpoint to create a new order. Requires authentication. Accepts a JSON payload containing order details.

**GET /orders:** Endpoint to retrieve all orders for the authenticated user. Requires authentication. Supports search, sorting, and filtering.

**PUT /orders/<order\_id>:** Endpoint to update a specific order by ID. Requires authentication. Accepts a JSON payload containing updated order details.

**DELETE /orders/<order\_id>:** Endpoint to delete a specific order by ID. Requires authentication.

**Tools/Frameworks/Libraries Used:**

Flask: Micro-framework for building web applications in Python.

Flask JWT Extended: Extension for Flask that adds JWT support for user authentication.

MongoEngine: Object-Document Mapper (ODM) for working with MongoDB in Python.

PyMongo: Python driver for MongoDB.

MongoDB: NoSQL database used for storing application data.

JSON Web Tokens (JWT): Used for user authentication and authorization.

**Elasticsearch: For implementing search functionality (though not used in the provided code).**

**This setup ensures that the application is secure, scalable, and reliable while maintaining clean, maintainable, and well-tested code. Further testing, documentation, and optimization can be done to enhance the application further.**