**Technical Backend Task**

**Backend Development for Order and Inventory Management Platform using Golang**

**Objective:** Develop a backend system in Golang for an order and inventory management platform. This platform will feature dynamic pricing for products influenced by factors such as demand and availability.

**Preferred Framework:** Go Fiber or any other suitable Golang framework.

**Requirements:**

**1. Database:**

- Utilize a Relational Database Management System (RDBMS), with a preference for PostgreSQL.

- Incorporate GORM as the Object-Relational Mapping (ORM) tool.

- Design database models for tables such as products, orders, customers, inventory, etc. Ensure these tables include relationships between them.

- Employ appropriate SQL joins for data retrieval in APIs, and implement methodologies to optimize query response time.

**2. APIs:**

**- User-Side APIs:**

- Features such as sign-up, login, product search, viewing products, placing orders, and accessing the user dashboard to view orders.

**- Admin-Side APIs:**

- Management of admin roles and inventory (adding, removing, updating products, etc.).

- Order management with filters (user, product, etc.) and sorting options, including details of the ordering user.

- APIs for generating statistics related to orders, users, inventory, etc., with appropriate filters and sorting capabilities.

- Implement middleware for access control across different APIs.

**3. Database Trigger for Dynamic Pricing:**

- Develop a database trigger to adjust product pricing based on demand and availability.

- Implement a trigger to update the inventory when orders are placed.

**4. Documentation:**

- Document all developed APIs with examples (Postman documentation preferred).

- Add necessary code comments for clarity and maintenance.

**Evaluation Criteria:**

**- Code Quality and Organization:** The code should be clean, readable, and well-structured.

**- Functionality:** Complete implementation of all specified requirements.

**- Scalability and Performance:** Optimized for high-volume data processing.

**- Error Handling:** Implement robust error handling and logging mechanisms.

**- Testing:** Extensive test coverage for various use cases.

**- Documentation:** Detailed and clear documentation of the system.

**Submission Instructions:**

- Complete the task within a **private** GitHub repository.

- Ensure the repository contains a comprehensive README with setup and usage instructions.

- Invite [recruitments@trademarkia.com](mailto:recruitments@trademarkia.com) for repository review.

- The submission deadline is 3 days from the date of the email.