TownSq Backend Technical Challenge

Objective:

This test assesses the candidate's ability to design and implement a RESTful API using Java and Spring Boot for managing purchase orders and payments, with a focus on security and code quality.

Story:

Our company heavily relies on efficient purchase order and payment processing. Your task is to build a simplified API to handle these core functions.

Requirements:

★ **API Endpoints:**

1. Purchase Orders:
   - `/orders`: Create a new purchase order (including items, quantities and total price).
   - `/orders/{id}`: Retrieve a specific purchase order by ID.
   - `/orders`: List all purchase orders for the authenticated user.
2. Payments:
   - `/payments`: Process a payment for a given purchase order (simulate payment processing, no real transactions).
   - `/payments/{id}`: Retrieve a payment by ID.
3. Users:
   - `/users`: Create a new user (user name, password, role).
   - `/users`: Return the current user information.
   - `/users/{id}`: Edit a specific user by ID (user name and role).

★ **Authentication:**

1. Admin User with the credentials:
   - User Name: `townsq_super`
   - Password: `verySecurePassword`
   - Role: `ADMIN`
2. Only users with the `ADMIN` role can create users.
3. Only users with the `ACCOUNT_MANAGER` can process payments.
4. Other users should be created with the `DEFAULT` role.

★ **Data Model:**

1. Design a suitable data model for purchase orders, payments and users, considering relationships between entities.

2. Use the database that you think better suits the solution.
3. The database should be running locally in a Docker container or embedded in the application.

- **Technology Stack:**
  1. Java
  2. Spring Boot
  3. Spring Data

Evaluation Criteria:

- **Functionality:** Does the API meet all the requirements and function correctly?
- **Code Quality:** Is the code well-structured, readable, and maintainable?
- **Security:** Is the API properly secured?
- **Testing:** Are there sufficient unit and integration tests?
- **Ease of Use:** Is the application easy to build, run, and understand?
- **Error Handling:** Does the API handle errors gracefully and provide informative messages?

Submission:

The candidate should submit their solution as a link to a Git repository (e.g., GitHub, GitLab, Bitbucket). The repository should include:

- Complete source code
- Clear instructions on how to build and run the application
- Any relevant documentation