https://test.imocha.io/AuthenticateKey?id=e7d40a9d7b

### Problem Statement: Java Spring Boot Application with CRUD Operations and Docker Integration

#### Overview:

Develop a Java Spring Boot application with CRUD operations on a `Product` entity. The application should use an H2 database for persistence and Lombok for boilerplate code reduction. The application may also include Swagger for API documentation. Additionally, you should create a Dockerfile to containerize the application and push the Docker image to Docker Hub.

#### Requirements:

1. Entity:

- The application will perform CRUD (Create, Read, Update, Delete) operations on a `Product` entity.

- Each `Product` will have the following fields: `id`, `name`, `description`, and `price`.

2. Database:

- Use an H2 in-memory database for storage.

3. Technologies:

- Spring Boot for creating the REST API.

- Lombok for reducing boilerplate code (Getters, Setters, Constructors, etc.).

- Swagger for generating API documentation.

- Docker for containerizing the application.

4. Operations:

- Create a new product.

- Retrieve all products or a single product by ID.

- Update product details by ID.

- Delete a product by ID.

5. Deployment:

- Build a Docker image for the application.

- Push the Docker image to Docker Hub.

Bonus (Optional)

* Adding a Selenium IDE recorded test script for swagger end points testing.
* Swagger Dependency

<dependency>

            <groupId>org.springdoc</groupId>

            <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>

            <version>2.2.0</version>

        </dependency>

* SwaggerConfig.java

import org.springdoc.core.models.GroupedOpenApi;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

@Configuration

public class SwaggerConfig {

    @Bean

    public GroupedOpenApi publicApi() {

        return GroupedOpenApi.builder()

                .group("customers-api")

                .pathsToMatch("/\*\*")

                .build();

    }

}