**LAPORAN HASIL PRAKTIKUM 6**

**PEMROGRAMAN BERORIENTASI OBJEK LANJUT**

**“Spring Boot-2”**

**Dosen Pengampu :**

**Sri Hartati Wijono, M.Kom.**



Oleh

Nama : Maria Gresia Plena Br Purba

NIM : 235314094

Kelas : DP

**PROGRAM STUDI INFORMATIKA**

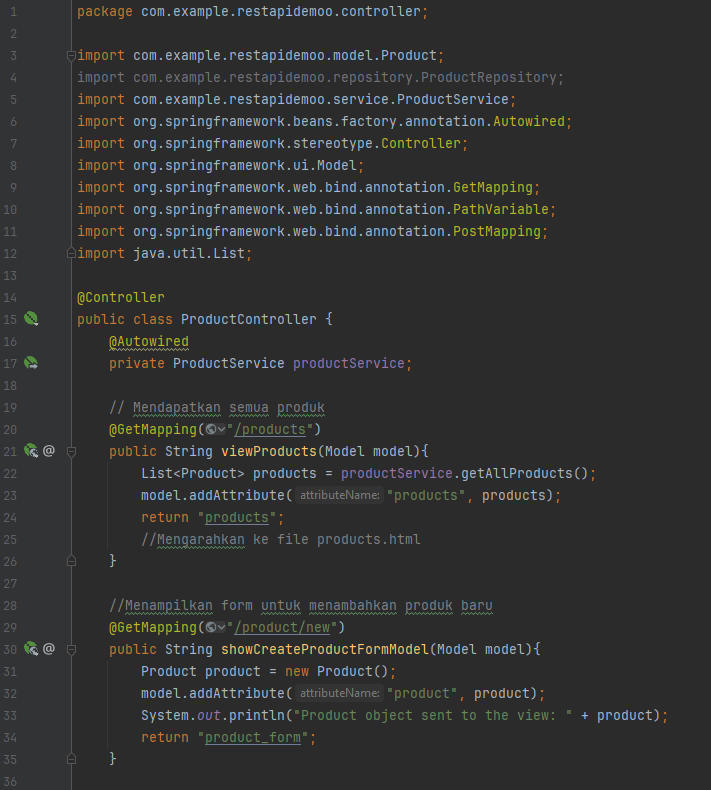
**FAKULTAS SAINS DAN TEKNOLOGI**

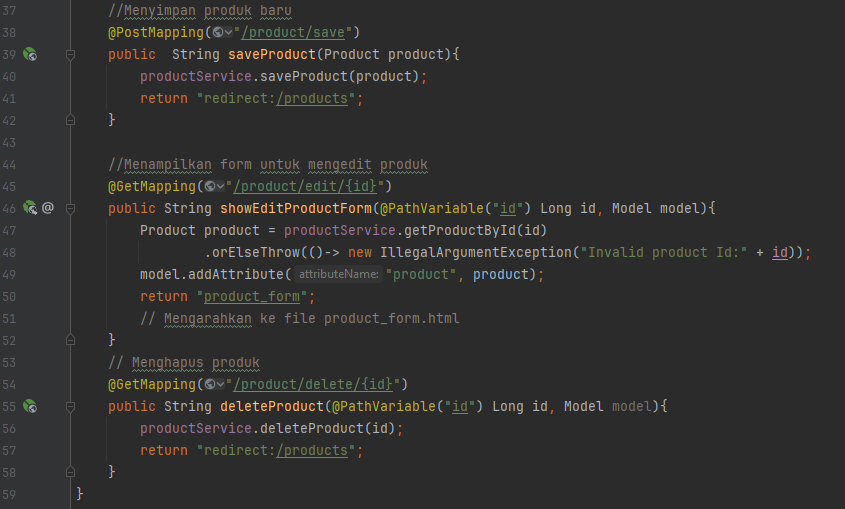
**UNIVERSITAS SANATA DHARMA**

**YOGYAKARTA**

**2024**

1. **TUJUAN**
2. Memahami cara membuat aplikasi Spring Boot REST API.
3. Memahami konsep CRUD (Create, Read, Update, Delete).
4. Menggunakan H2 Database untuk menyimpan data produk.
5. Menggunakan Spring Data JPA untuk operasi CRUD.
6. **PRAKTIKUM**
   1. **Product Controller**

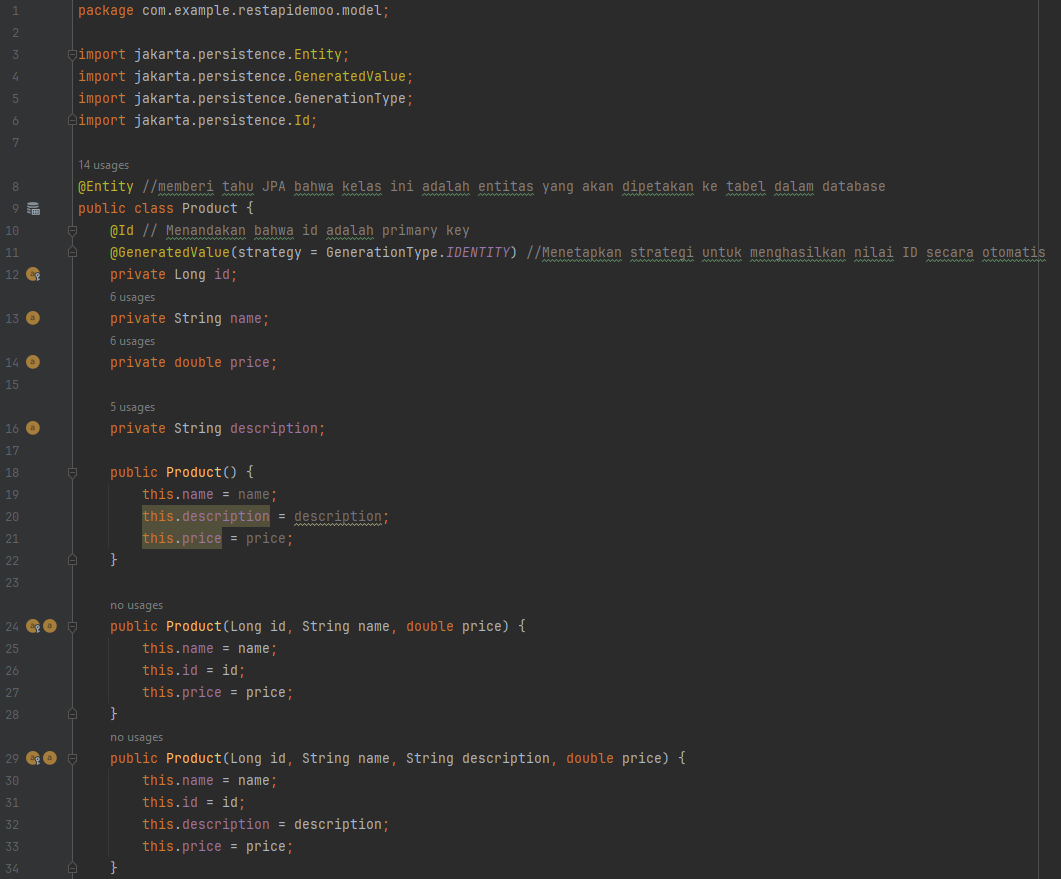




**Penjelasan :**

Method viewProducts berfungsi untuk mendapatkan semua produk dari productService menggunakan method getAllPorducts() yang akan disimpan ke dalam variabel “products” dan menambahkannya sebagai attribute model dengan menggunakan addAttribute() dan disimpan dengan nama “products”. Kemudian, mengembalikan tampilann yang diarahkan ke file products.html. Pada method showCreateProductFormModel() membuat objek yang bernama “product” dan menambahkan attribute dengan menggunakan addAttribute() pada model. Kemudian, mencetak pesan yang menunjukkan produk apa yang dikirim dan mengembalikan tampilan yang diarahkan ke file product\_form.html. Method saveProduct() berfungsi untuk menyimpan produk baru ke productService dengan memanggil method saveProduct(). Kemudian, halaman akan dialihkan ke /products. Method showEditProductForm() berfungsi untuk menampilkan form untuk mengedit produk. Mendapatkan id dengan menggunakan getProductById dari productService yang akan disimpan ke dalam “product”. Jika tidak ditemukan, maka akan menampilkan pesan yang menunjukkan bahwa id yang dicari tidak valid. Kemudian, menambahkan attribute baru dengan menggunakan addAttribute pada model yang akan diarahkan ke file products\_form.html. Method productService() berfungsi untuk menghapus produk berdasarkan id dengan menggunakan method deleteProduct() pada productService. Kemudian, halaman akan dialihkan ke /products.

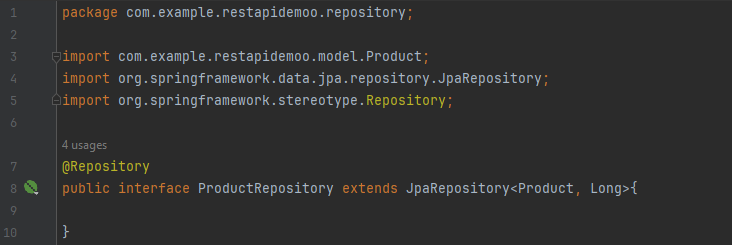
* 1. **Product**

****

**Penjelasan :**

Mendeklarasikan atribut bernama id bertipe Long, name dan description bertipe String, dan price bertipe double yang masing-masing bersifat private. Terdapat 3 constructor yang di mana satu default, satu dengan tiga parameter, dan satu dengan empat parameter. Menggunakan getter setter yang berfungsi untuk mendapatkan nilai dan mengubah nilai dari atribut.

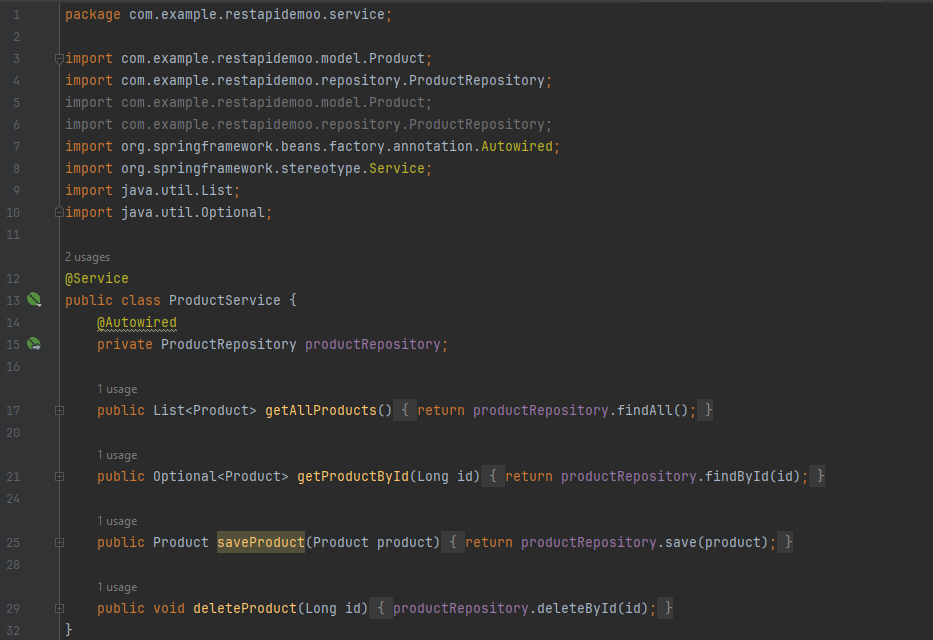
* 1. **ProductRepository**

****

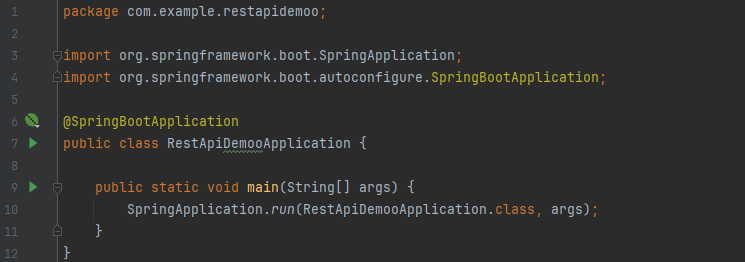
**Penjelasan :**

Interface ini mewarisi JpaRepository yang memiliki method method yang dapat digunakan, seperti findAll(), save(), findById(), dan deleteById().

* 1. **ProductService**

**Penjelasan :**

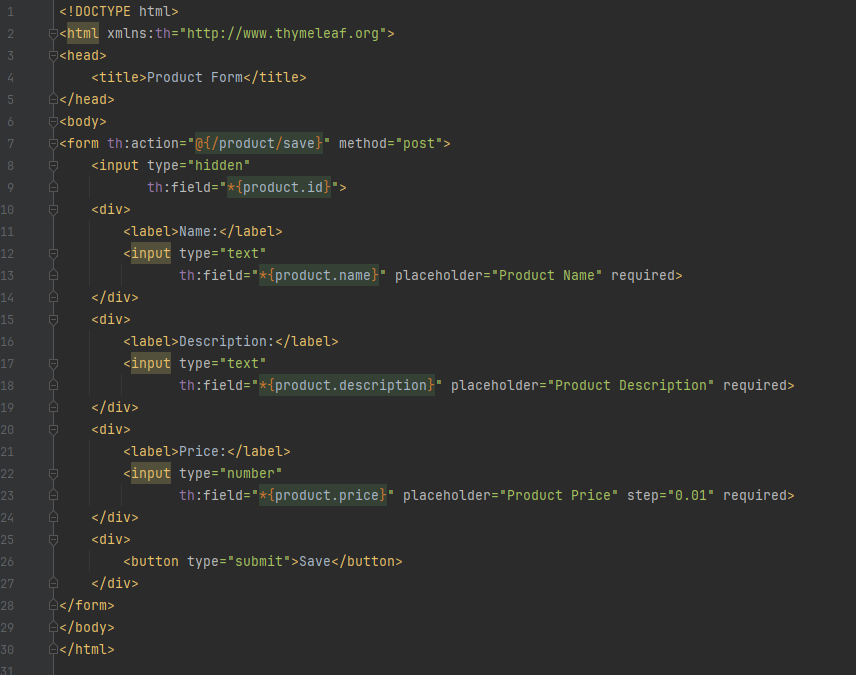
Method getAllProducts berfungsi untuk mendapatkan semua produk dengan memanggil method findAll() pada productRepository. Method getProductById berfungsi untuk mendapatkan data berdasarkan id dengan menggunakan method findById() pada productRepository. Method saveProduct() akan berfungsi untuk menyimpan produk baru dengan menggunakan save() pada productRepository. Kemudian, method deleteProduct() berfungsi untuk menghapus data dengan menggunakan deleteById() pada productRepository.

* 1. **RestApiDemooApplication**

**Penjelasan :**

Pada class ini terdapat method main() yang akan memanggil SpringApplication.run() untuk menjalankan keseluruhan aplikasi.

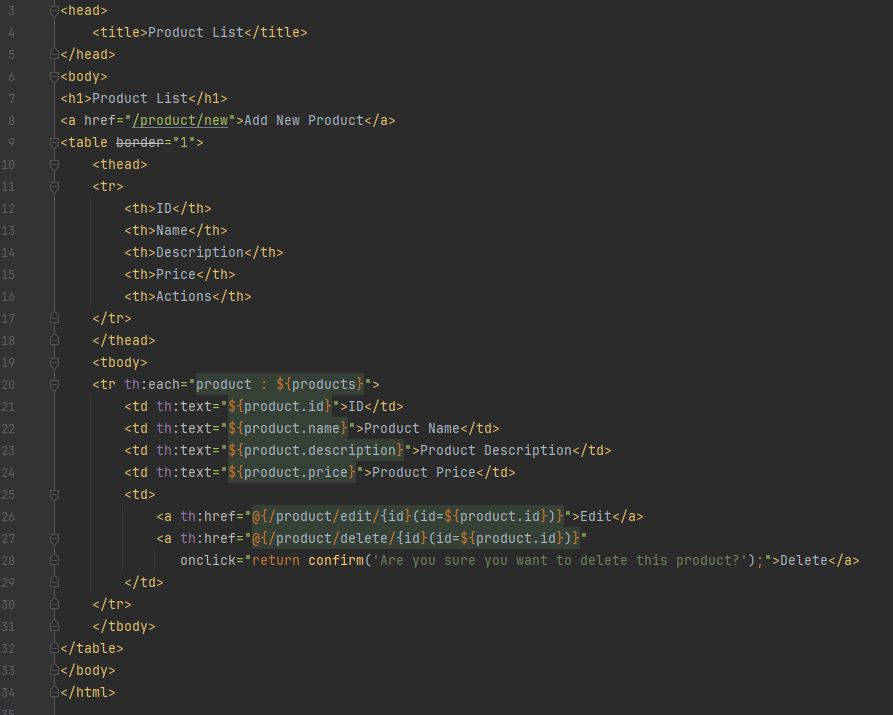
* 1. **product\_form.html**

****

**Penjelasan :**

Form ini menggunakan atribut th:field dari Thymeleaf. Form ini berfungsi untuk meminta input pada product name, product description, dan product price kepada user. Inputan tersebut akan disimpan dengan menggunakn method “post” dengan type “hidden” yang memiliki field “product.id”. Product name dan product description memiliki type “text”, sedangkan price memiliki type “number”. Terdapat button dengan nama “submit” untuk mengirim dan menyimpan data.

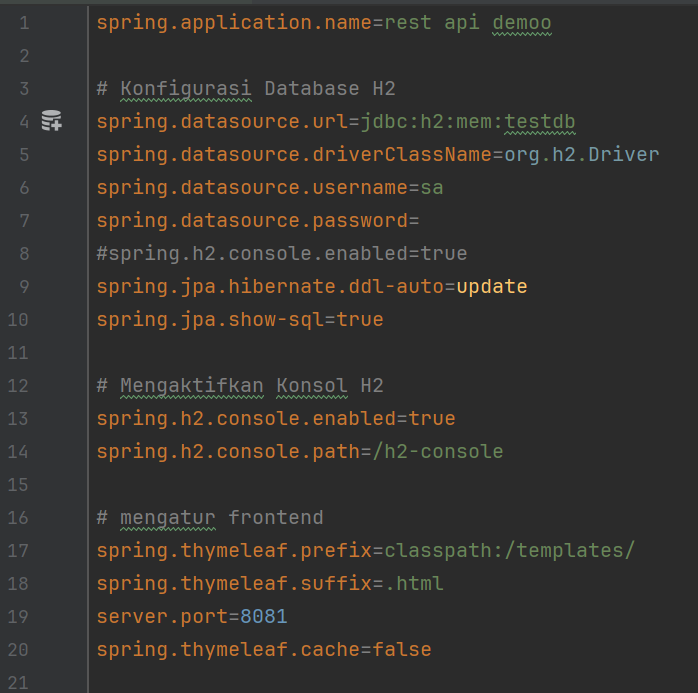
* 1. **products.html**

****

**Penjelasan :**

From ini menggunakan th:each dari Thymeleaf. From ini berfungsi untuk menampilkan daftar produk. Terdiri dari 5 kolom yang bernama “ID”, “Name”, “Description”, “Price”, dan “Actions”. Membuat pilihan untuk untuk mengedit produk berdasarkan id nya dengan menggunakan “@{/product/edit/{id}(id=${product.id})}” yang kemudian diberikan nama “Edit”. Kemudian, terdapat pilihan untuk menghapus dengan menggunakan "@{/product/delete/{id}(id=${product.id})}" yang memberikan teks konfirmasi penghapusan untuk memastikan ingin menghapus data atau tidak dan diberikan nama “Delete”.

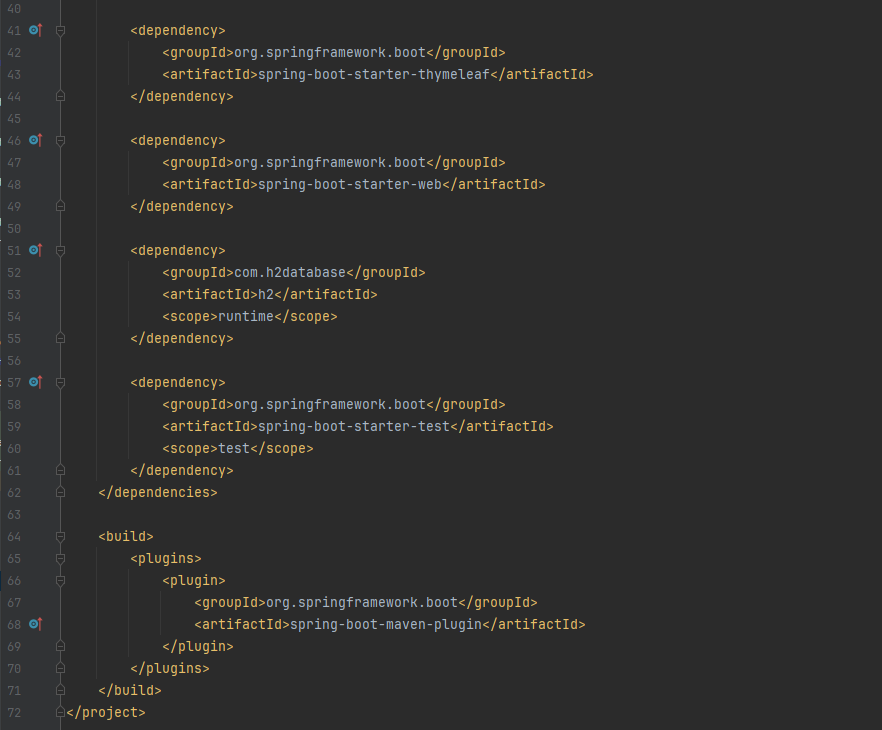
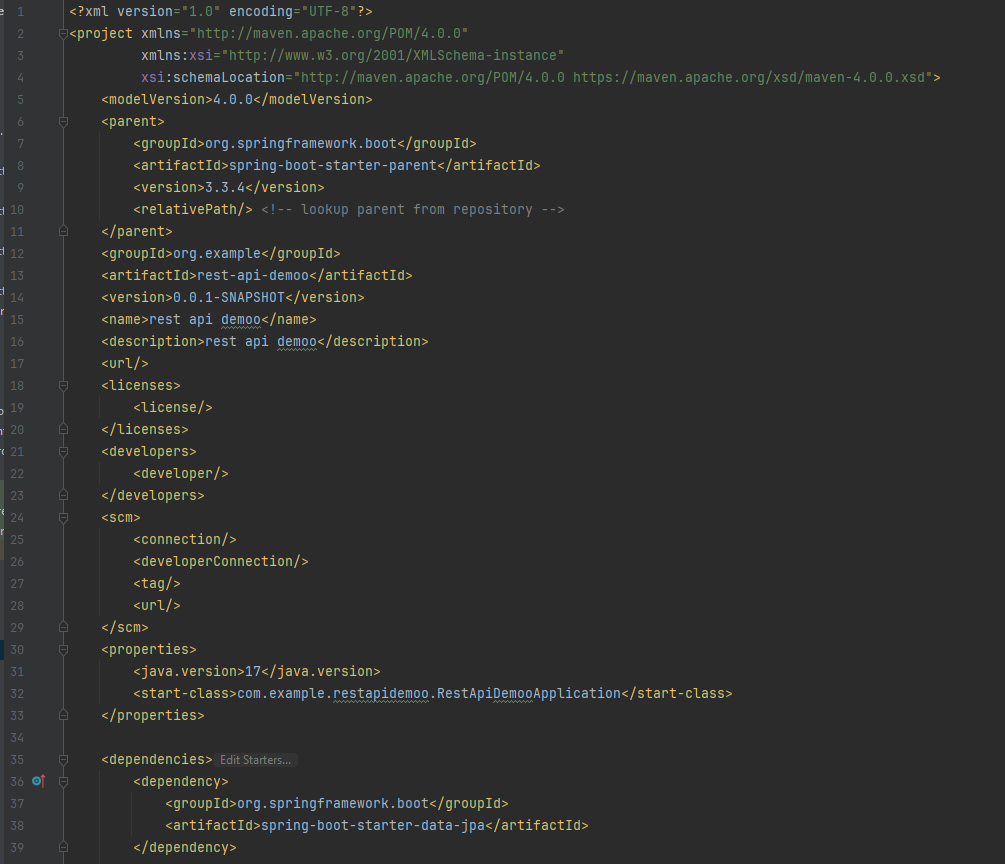
* 1. **application.properties**

****

**Penjelasan :**

Database h2 disetting dengan menggunakan “jdbc:h2:mem:testdb” dengan driverClassName “org.h2.Driver”. Kemudian, memberikan data username, yaitu “sa” tanpa password. Mengaktifkan konsol h2 dengan menggunakan “spring.h2.console.enabled=true” dan memiliki path “path=/h2-console”. Lalu, mengatur frontend dengan menggunakan “spring.thymeleaf” yang di mana “.prefix=classpath:/templates/” untuk dan “.suffix=.html” untuk , menggunakan port 8081 dan “spring.thymeleaf.cache=false” untuk

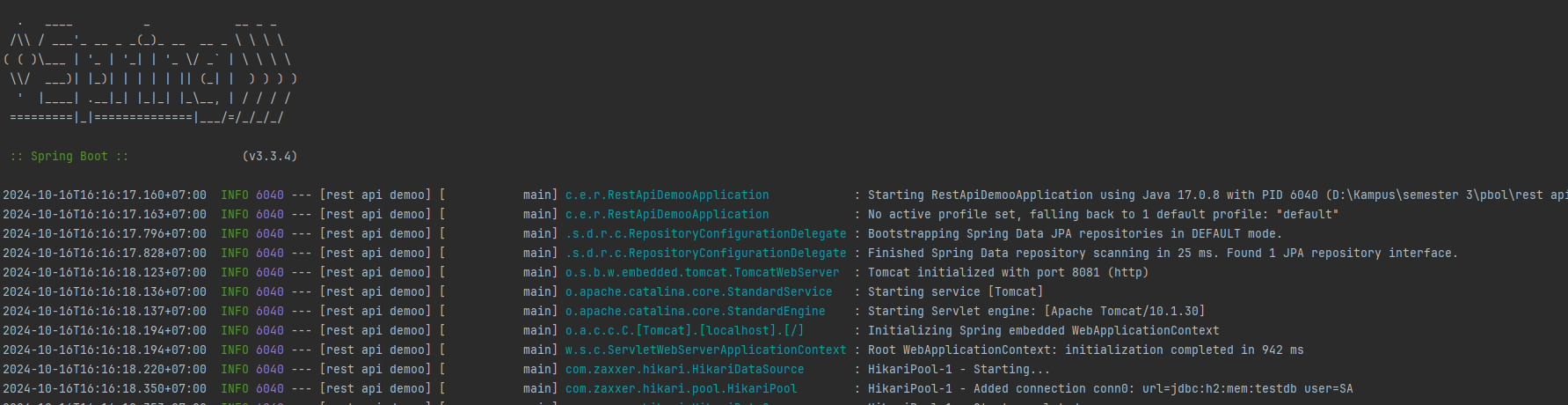
* 1. **Pom.xml**

****

**Penjelasan :**

Memiliki groupId org.example dan artifactId rest-api-demoo, Menggunakan java dengan versi 17. Pada dependencied terdapat spring-boot-starter-data-jpa yang berfungsi untuk mengimplementasikan JPA repositories, spring-boot-starter-thymeleaf yang berfungsi untuk mengonfigurasi template Thymeleaf secara otomatis dalam aplikasi web, spring-boot-starter-web yang berfungsi untuk menyediakan aplikasi web Spring Boot yang siap produksi, dan spring-boot-starter-test yang berfungsi untuk menyediakan alat untuk menguji. Pada build plugind terdapat spring-boot-maven-plugin yang berfungsi untuk menyediakan dukungan Spring Boot di Apache Maven

* 1. **Output**

****

