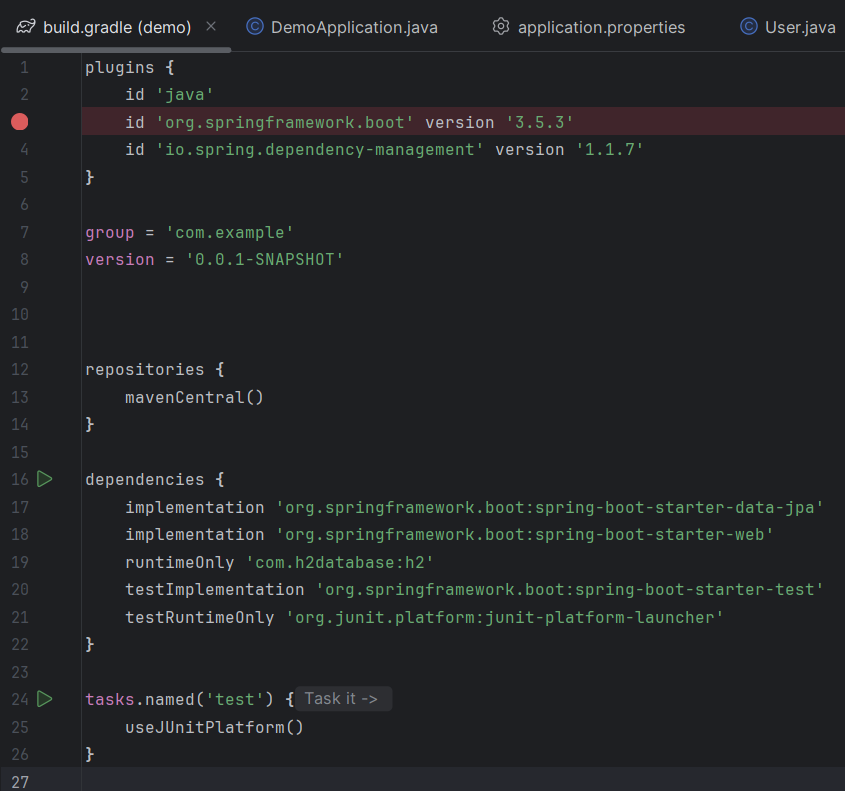
**Cognizant Deep Skilling Mandatory Hands-On Questions**

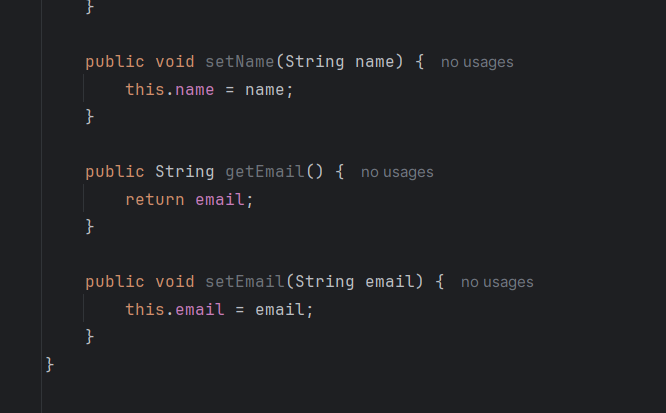
**TOPIC- Spring Data JPA with Spring Boot, Hibernate Exercises**

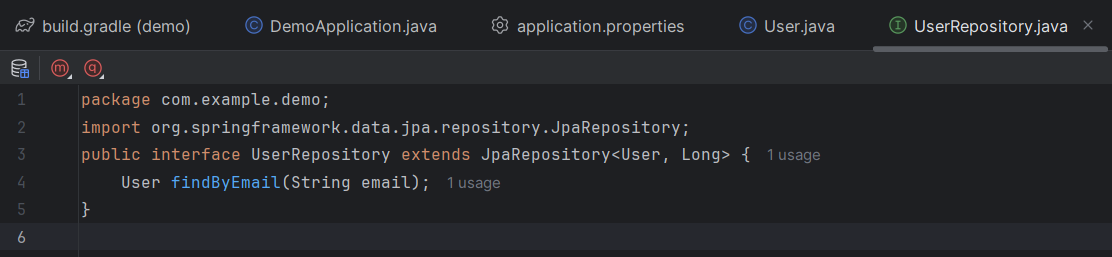
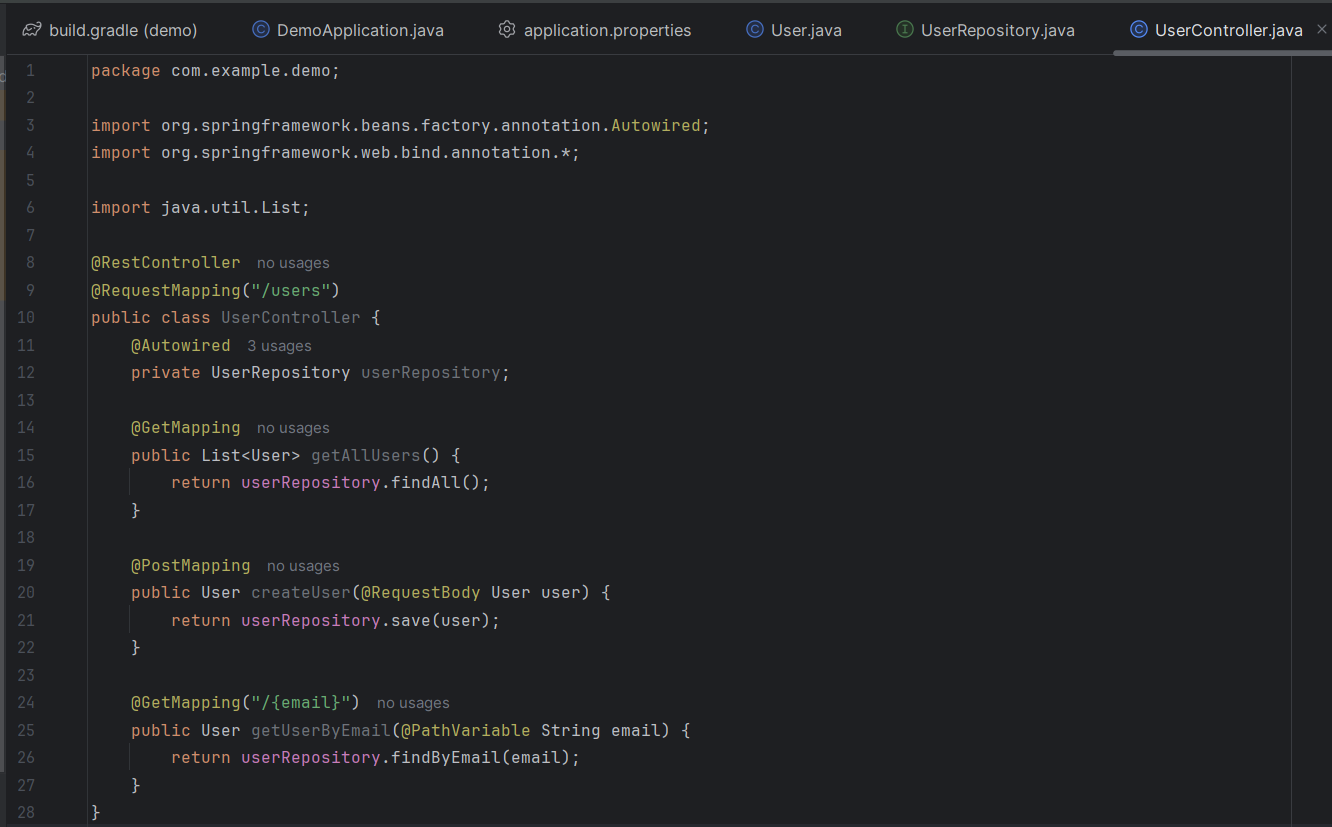
**Q.1 Explain Spring Data JPA - Quick Example?**

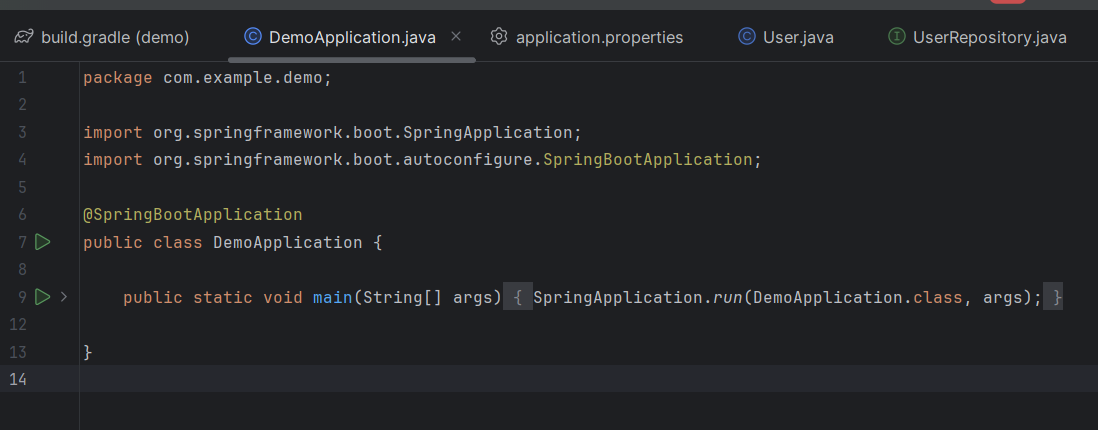
**Ans)**

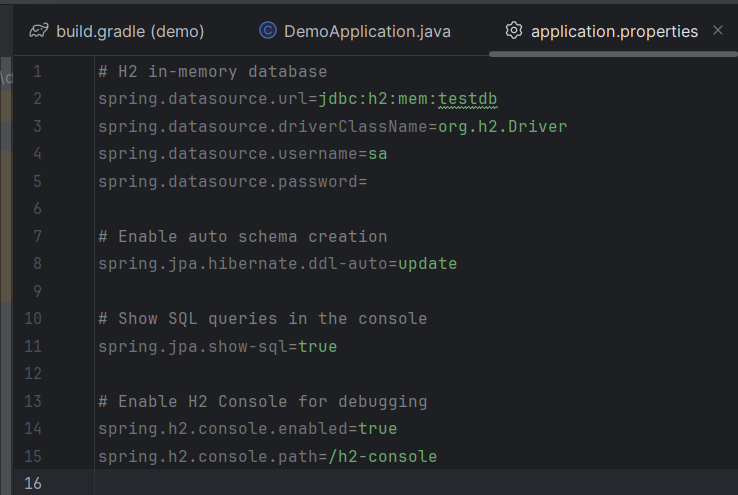
1. build.gradle(demo):
2. User.java:

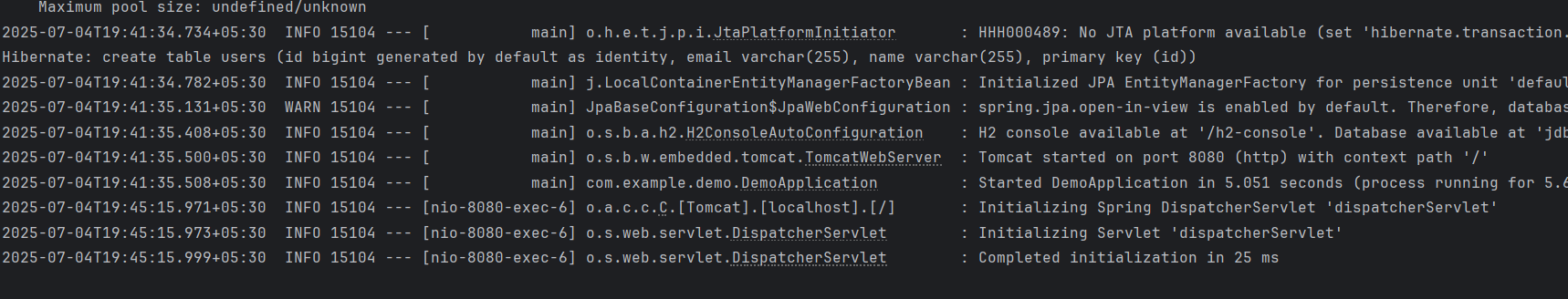


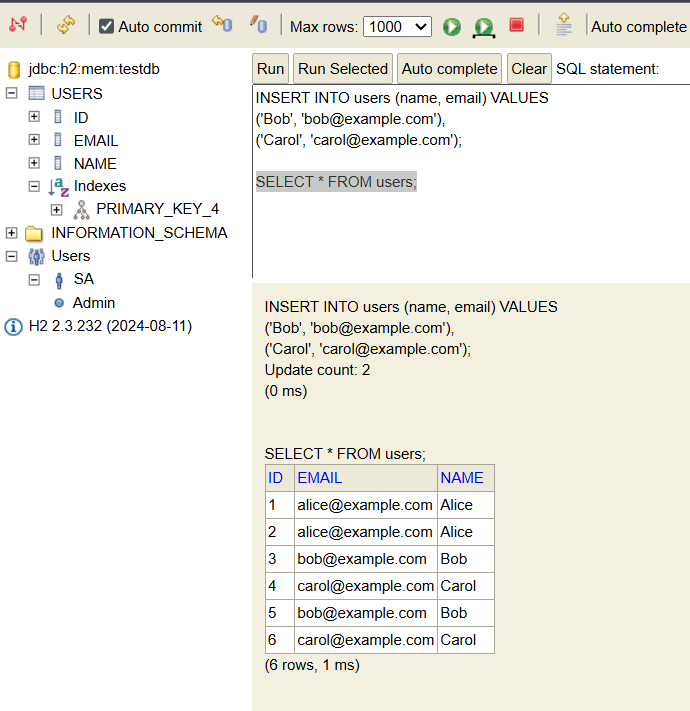
****

1. ****UserRepository:
2. ****UserController:
3. DemoApplication

****

1. application.properties:
2. Output in terminal:



1. Output in DBMC:

**Q.2 Explain the difference between Java Persistence API, Hibernate and Spring Data JPA?**

Ans)

1. **Java Persistence API (JPA)**:

JPA is a standard specification (part of Java EE) for object-relational mapping (ORM) in java. It defines a set of interfaces and annotations to map Java objects (entities) to relational database tables.

Examples of Annotations: @Entity, @Table, @Id, @OneToMany, etc.

1. **Hibernate:**

Hibernate is an ORM framework and JPA implementation.

It implements the JPA specification and provide additional features beyond JPA, such as: Caching, Lazy loading, Better fetching strategies.

You can use Hibernate with or without JPA. When you use JPA with Hibernate, you’re using Hibernate as the JPA provider.

1. **Spring Data JPA:**

Spring Data JPA is a part of the larger Spring Data project. It’s built on top of JPA. It provides a higher-level abstraction to simplify JPA-based repository implementation. It uses interfaces such as CrudRepository, JpaRepository interfaces. It uses auto-generated query method names such as findByEmail(). It uses less boilerplate code for data access layers.

| **Feature** | **JPA** | **Hibernate** | **Spring Data JPA** |
| --- | --- | --- | --- |
| Type | Specification (API) | Framework/Implementation | Abstraction layer on top of JPA |
| Provided by | Java EE / Jakarta EE | Hibernate ORM project | Spring Framework |
| Can work standalone? | No (needs a provider) | Yes | No (requires Spring + JPA + provider) |
| Uses JPA annotations | Yes | Yes (plus its own extra ones) | Yes |
| Purpose | Define ORM standards | Implement ORM logic | Simplify JPA-based repository code |
| Typical use case | Abstract ORM layer | Full-featured ORM with extras | Quick and simple data access layer in Spring |