Java Persistent API

Spring Boot JPA

Spring Boot JPA is a Java specification for managing relational data in Java applications. It allows us to access and persist data between Java object class and relational database. JPA follows Object-Relation Mapping (ORM). It is a set of interfaces. It also provides a runtime EntityManager API for processing queries and transactions on the objects against the database. It uses a platform- independent object-oriented query language JPQL

JPA is not a framework. It defines a concept that can be implemented by any framework.

Object-Relation Mapping (ORM)

In ORM, the mapping of Java objects to database tables, and vice-versa is called Object-Relational Mapping. The ORM mapping works as a bridge between a relational database (tables and records) and Java application (classes and objects).

Add the following Dependency:-

***Spring Web**

***Spring data JPA**

***MYSQL Driver**

MainController.java

```
package com.example.demo;


import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.GetMapping;
```

```
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
@RequestMapping(path="/save")
public class MainController {
  @Autowired
  private UserRepository userRepository;

  @PostMapping(path="/add")
  public @ResponseBody String addNewUser (@RequestParam String name
      , @RequestParam String email) {

    User n = new User();
    n.setname(name);
    n.setemail(name);
    userRepository.save(n);
    return "Saved";
  }

  @GetMapping(path="/all")
  public @ResponseBody Iterable<User> getAllUsers() {
    return userRepository.findAll();
  }
}
```

Mysql1Application.java

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class Mysql1Application {

	public static void main(String[] args) {
		SpringApplication.run(Mysql1Application.class, args);
	}

}
```

User.java

```
package com.example.demo;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;

@Entity
public class User{
	@Id
```

```
@GeneratedValue(strategy=GenerationType.AUTO)
private Integer id;

private String name;

private String email;

public Integer getId() {
        return id;
}

public void setId(Integer id) {
        this.id=id;
}

public String getname() {
        return name;
}
public void setname(String name) {
        this.name=name;
}
public String getEmail() {
        return email;
}
public void setemail(String email) {
        this.email=email;
}
```

```
    }
```

UserRepository.java

```
package com.example.demo;

import org.springframework.data.repository.CrudRepository;

public interface UserRepository extends CrudRepository<User, Integer> {

}
```

application.properties

```
spring.application.name=mysql-1
server.port = 4040
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3305/sonya
spring.datasource.username=root
spring.datasource.password=kishan
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.jpa.show-sql: true
```

eclipse-workspace - mysql-1/src/main/java/com/example/demo/MainController.java - Eclipse IDE

```java
package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;

@Controller
@RequestMapping(path="/save")
public class MainController {
  @Autowired
  private UserRepository userRepository;

  @PostMapping(path="/add")
  public @ResponseBody String addNewUser (@RequestParam String name
      , @RequestParam String email) {

    User n = new User();
    n.setname(name);
    n.setemail(name);
    userRepository.save(n);
    return "Saved";
  }

  @GetMapping(path="/all")
  public @ResponseBody Iterable<User> getAllUsers() {
    return userRepository.findAll();
  }
}
```

eclipse-workspace - mysql-1/src/main/java/com/example/demo/Mysql1Application.java - Eclipse IDE

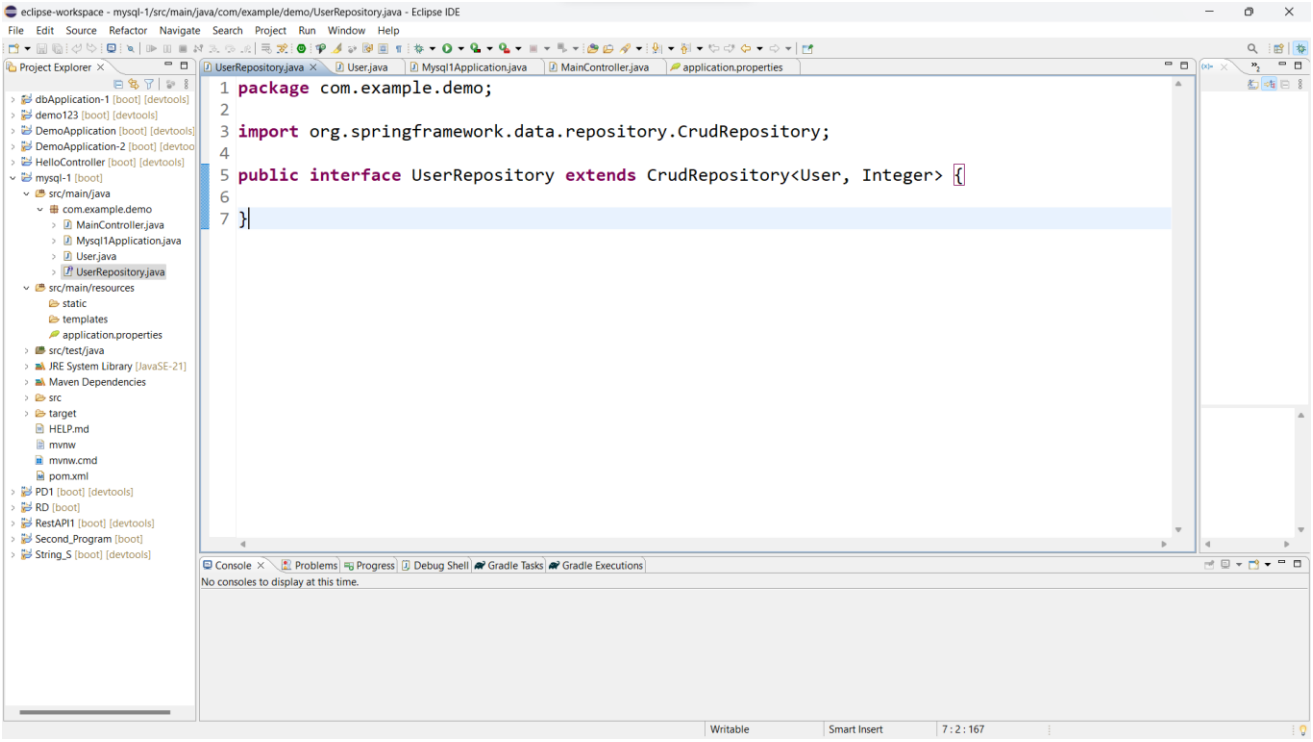
```java
package com.example.demo;

import org.springframework.boot.SpringApplication;

@SpringBootApplication
public class Mysql1Application {

    public static void main(String[] args) {
        SpringApplication.run(Mysql1Application.class, args);
    }

}
```

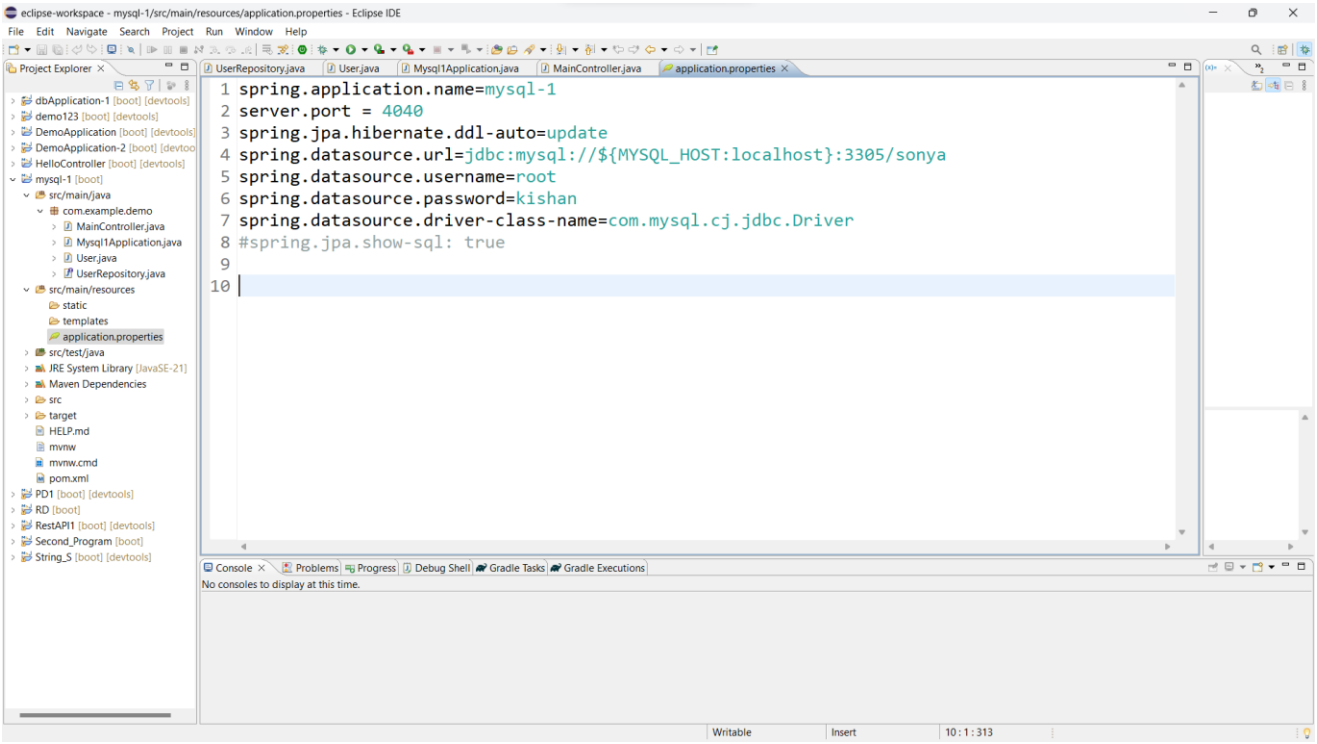
eclipse-workspace - mysql-1/src/main/java/com/example/demo/User.java - Eclipse IDE

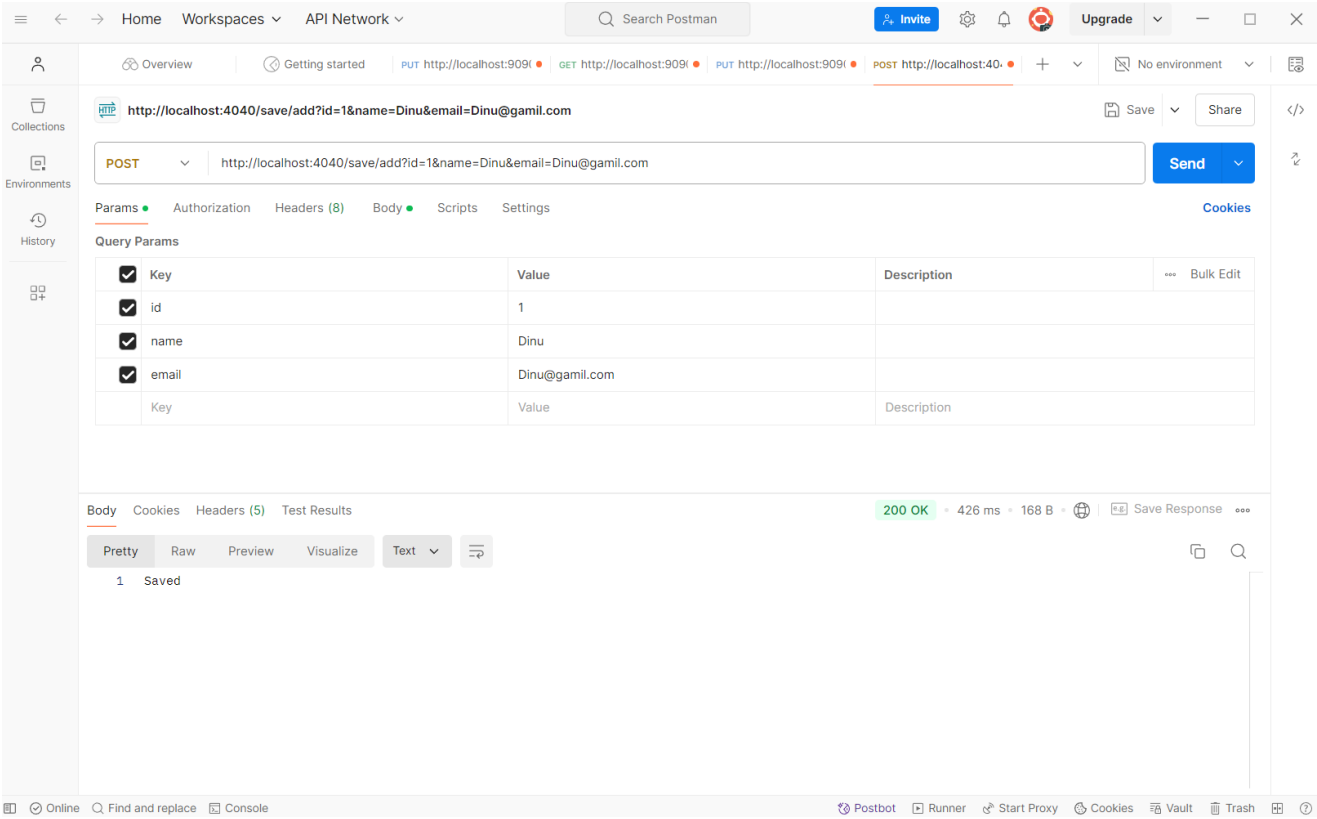
```java
package com.example.demo;

import jakarta.persistence.Entity;

@Entity
public class User{
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private Integer id;

    private String name;

    private String email;

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id=id;
    }
```

eclipse-workspace - mysql-1/src/main/java/com/example/demo/User.java - Eclipse IDE

```java
        return id;
    }

    public void setId(Integer id) {
        this.id=id;
    }

    public String getname() {
        return name;
    }
    public void setname(String name) {
        this.name=name;
    }
    public String getEmail() {
        return email;
    }
    public void setemail(String email) {
        this.email=email;
    }
    }
```


eclipse-workspace - mysql-1/src/main/java/com/example/demo/UserRepository.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
UserRepository.java
User.java
Mysql1Application.java
MainController.java
application.properties
package com.example.demo;
import org.springframework.data.repository.CrudRepository;
public interface UserRepository extends CrudRepository<User, Integer> {
}
Console
Problems
Progress
Debug Shell
Gradle Tasks
Gradle Executions
No consoles to display at this time.
Writable
Smart Insert
7 : 2 : 167


eclipse-workspace - mysql-1/src/main/resources/application.properties - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Project Explorer
UserRepository.java
User.java
Mysql1Application.java
MainController.java
application.properties
spring.application.name=mysql-1
server.port = 4040
spring.jpa.hibernate.ddl-auto=update
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3305/sonya
spring.datasource.username=root
spring.datasource.password=kishan
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
#spring.jpa.show-sql: true
Console
Problems
Progress
Debug Shell
Gradle Tasks
Gradle Executions
No consoles to display at this time.
Writable
Insert
10 : 1 : 313



MySQL 8.0 Command Line Client

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use sonya;
Database changed
mysql> desc user;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| id    | int          | YES  |     | NULL    |       |
| email | varchar(255) | YES  |     | NULL    |       |
| name  | varchar(255) | YES  |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.02 sec)

mysql> select * from user;
+------+-------+------+
| id   | email | name |
+------+-------+------+
|    1 | tom   | tom  |
|    2 | Ajju  | Ajju |
|   52 | Dinu  | Dinu |
+------+-------+------+
3 rows in set (0.00 sec)

mysql>
```