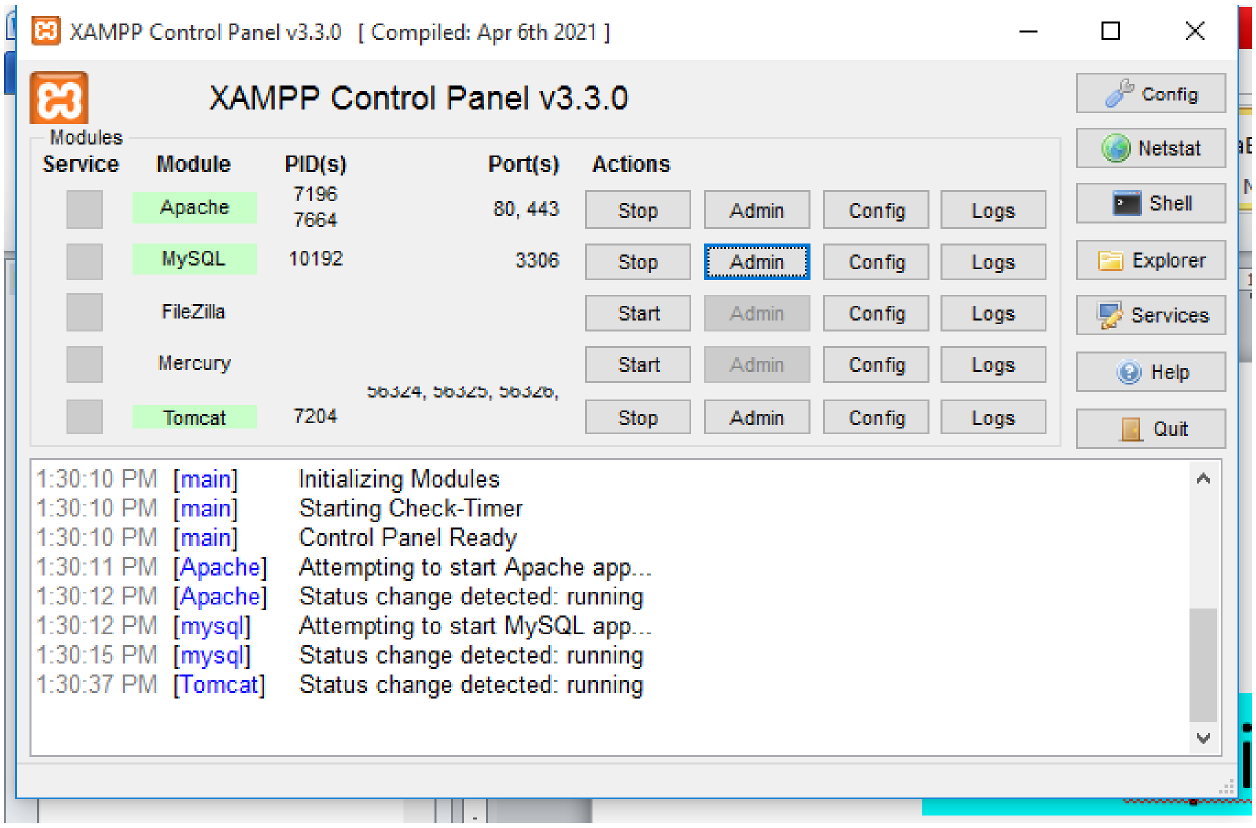
Rest Api in Spring Boot :-

Install xampp and run it



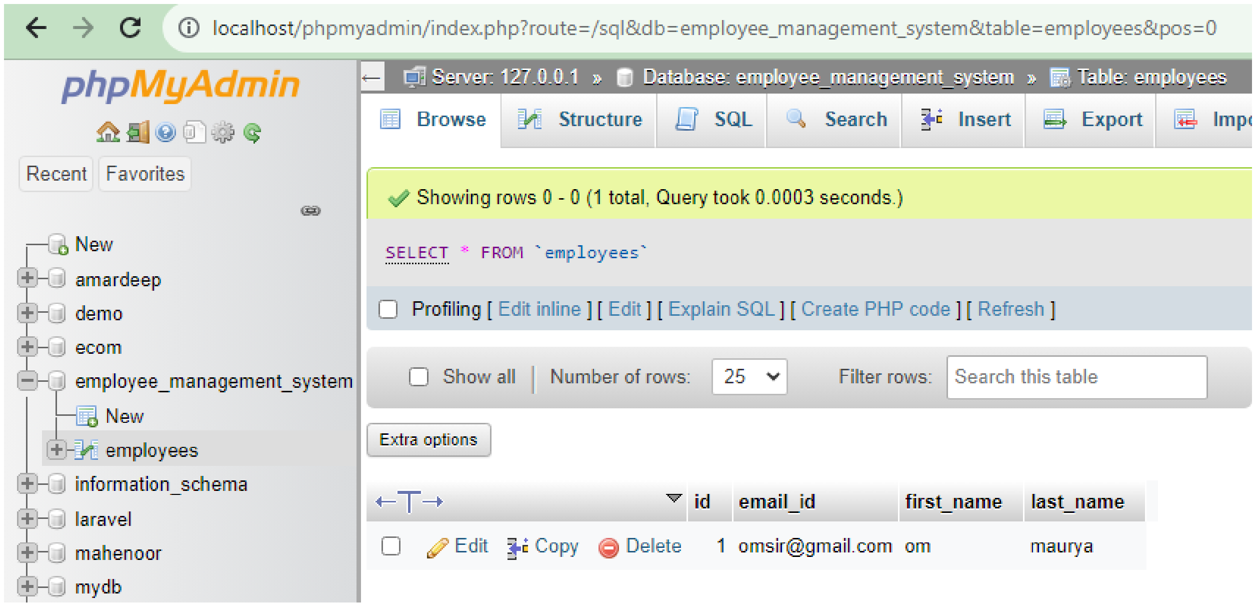
XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]

XAMPP Control Panel v3.3.0

Service	Module	PID(s)	Port(s)	Actions			
	Apache	7196 7664	80, 443	Stop	Admin	Config	Logs
	MySQL	10192	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat	7204	56324, 56325, 56326,	Stop	Admin	Config	Logs

```
1:30:10 PM  [main]     Initializing Modules
1:30:10 PM  [main]     Starting Check-Timer
1:30:10 PM  [main]     Control Panel Ready
1:30:11 PM  [Apache]   Attempting to start Apache app...
1:30:12 PM  [Apache]   Status change detected: running
1:30:12 PM  [mysql]    Attempting to start MySQL app...
1:30:15 PM  [mysql]    Status change detected: running
1:30:37 PM  [Tomcat]   Status change detected: running
```

And Create database and table inside your localhost/phpmyadmin



localhost/phpmyadmin/index.php?route=/sql&db=employee_management_system&table=employees&pos=0

Server: 127.0.0.1 » Database: employee_management_system » Table: employees

Showing rows 0 - 0 (1 total, Query took 0.0003 seconds.)

```
SELECT * FROM `employees`
```

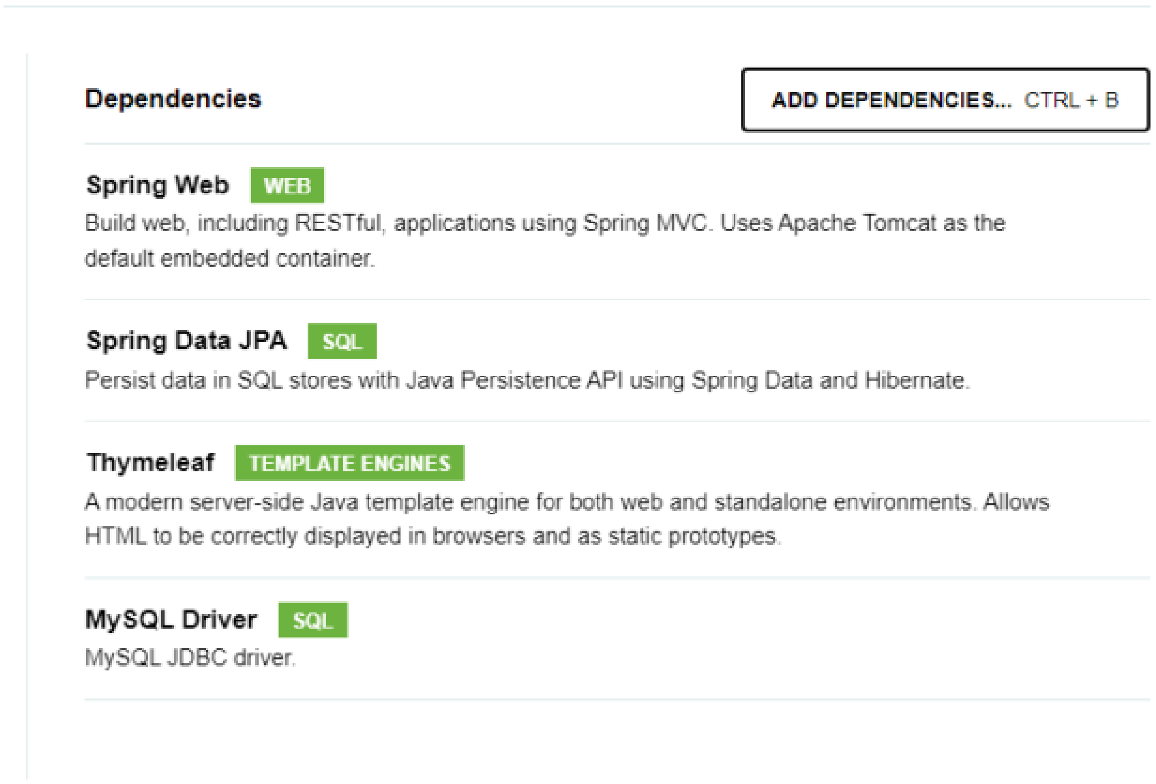
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

	id	email_id	first_name	last_name
Edit Copy Delete	1	omsir@gmail.com	om	maurya

go to website https://start.spring.io/

Note:- Following dependencies are important



Dependencies

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL

Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Thymeleaf TEMPLATE ENGINES

A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

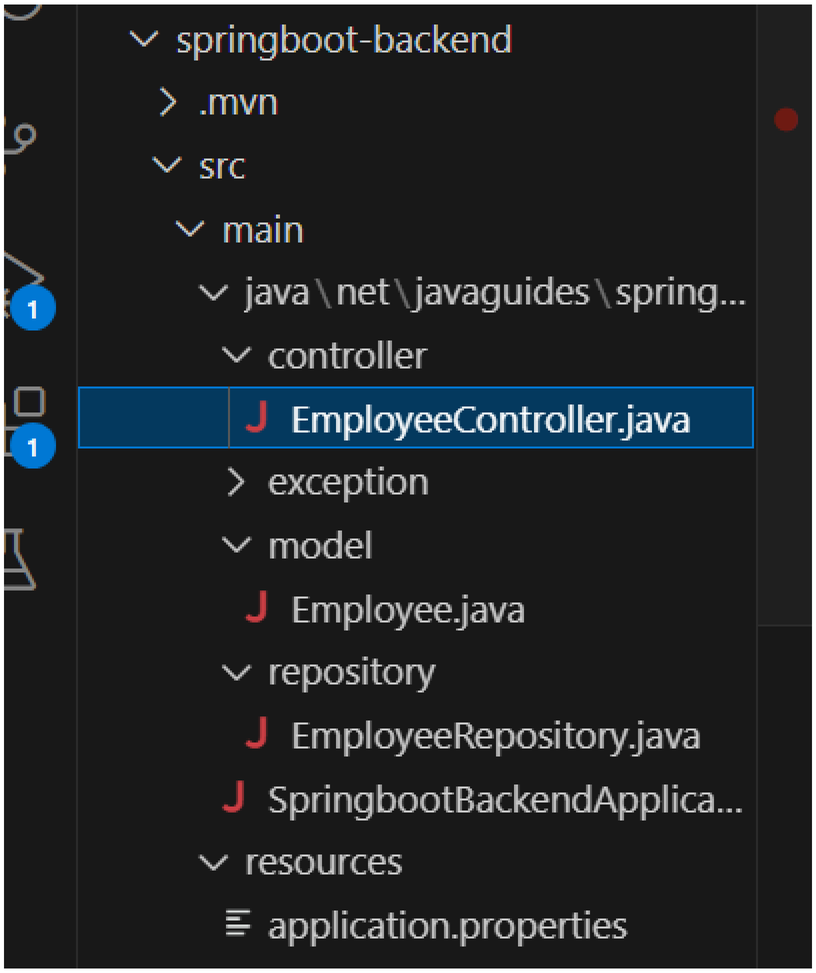
MySQL Driver SQL

MySQL JDBC driver.

Finally click on Generate and you will get zip file downloaded

And extract it open with visual code here you will create following package and files as shown below :-

Project structure :-



```
springboot-backend
  .mvn
  src
    main
      java\net\javaguides\spring...
        controller
          EmployeeController.java
        exception
        model
          Employee.java
        repository
          EmployeeRepository.java
        SpringbootBackendApplica...
      resources
        application.properties
```

Step 1:-

Write code in application.properties file :-

```
spring.datasource.url=jdbc:mysql://localhost:3306/employee_management_system?useS
SL=false
spring.datasource.username=root
spring.datasource.password=

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQLDialect

spring.jpa.hibernate.ddl-auto = update
```

step 2:-

create model folder and inside it `Employee.java` file code:-

```
package net.javaguides.springboot.model;
```

```java
import jakarta.persistence.*;

@Entity
@Table(name = "employees")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private long id;

    @Column(name = "first_name")
    private String firstName;

    @Column(name = "last_name")
    private String lastName;

    @Column(name = "email_id")
    private String emailId;

    public Employee() {

    }

    public Employee(String firstName, String lastName, String emailId) {
        super();
        this.firstName = firstName;
        this.lastName = lastName;
        this.emailId = emailId;
    }
    public long getId() {
        return id;
    }
    public void setId(long id) {
        this.id = id;
    }
    public String getFirstName() {
        return firstName;
    }
    public void setFirstName(String firstName) {
        this.firstName = firstName;
    }
    public String getLastName() {
        return lastName;
    }
    public void setLastName(String lastName) {
        this.lastName = lastName;
    }
    public String getEmailId() {
        return emailId;
    }
    public void setEmailId(String emailId) {
        this.emailId = emailId;
    }
}
```

Step 3:-

Create repository folder inside it `EmployeeRepository.java` `file code`

```
package net.javaguides.springboot.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import net.javaguides.springboot.model.Employee;

@Repository
public interface EmployeeRepository extends JpaRepository<Employee, Long>{

}
```

Step 4:- create controller folder and inside it

Write code for `EmployeeController.java` `file code`

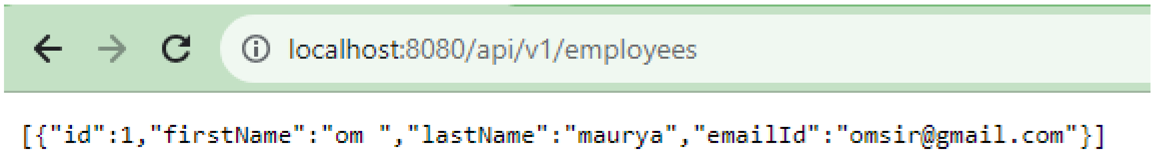
```
package net.javaguides.springboot.controller;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import net.javaguides.springboot.exception.ResourceNotFoundException;
import net.javaguides.springboot.model.Employee;
import net.javaguides.springboot.repository.EmployeeRepository;

@CrossOrigin(origins = "http://localhost:4200")
@RestController
@RequestMapping("/api/v1/")
public class EmployeeController {

    @Autowired
    private EmployeeRepository employeeRepository;

    // get all employees
    @GetMapping("/employees")
    public List<Employee> getAllEmployees(){
        return employeeRepository.findAll();
    }

    // create employee rest api
    @PostMapping("/employees")
    public Employee createEmployee(@RequestBody Employee employee) {
        return employeeRepository.save(employee);
    }

    // get employee by id rest api
    @GetMapping("/employees/{id}")
    public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {
        Employee employee = employeeRepository.findById(id)
                .orElseThrow(() -> new ResourceNotFoundException("Employee not
exist with id :" + id));
        return ResponseEntity.ok(employee);
    }

    // update employee rest api
    @PutMapping("/employees/{id}")
    public ResponseEntity<Employee> updateEmployee(@PathVariable Long id,
@RequestBody Employee employeeDetails){
        Employee employee = employeeRepository.findById(id)
                .orElseThrow(() -> new ResourceNotFoundException("Employee not
exist with id :" + id));

        employee.setFirstName(employeeDetails.getFirstName());
        employee.setLastName(employeeDetails.getLastName());
        employee.setEmailId(employeeDetails.getEmailId());

        Employee updatedEmployee = employeeRepository.save(employee);
        return ResponseEntity.ok(updatedEmployee);
    }

    // delete employee rest api
    @DeleteMapping("/employees/{id}")
    public ResponseEntity<Map<String, Boolean>> deleteEmployee(@PathVariable Long
id){
        Employee employee = employeeRepository.findById(id)
                .orElseThrow(() -> new ResourceNotFoundException("Employee not
exist with id :" + id));

        employeeRepository.delete(employee);
        Map<String, Boolean> response = new HashMap<>();
        response.put("deleted", Boolean.TRUE);
        return ResponseEntity.ok(response);
    }


}
```

And your SpringbootBackendApplication.java application file code

```
package net.javaguides.springboot;

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

@SpringBootApplication
public class SpringbootBackendApplication {
```

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

}
```

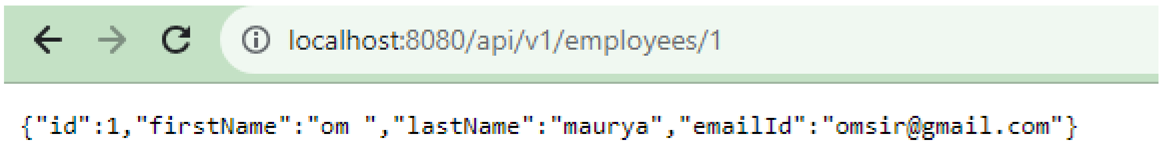
Output:-



localhost:8080/api/v1/employees

```
[{"id":1,"firstName":"om ","lastName":"maurya","emailId":"omsir@gmail.com"}]
```

And for particular id



localhost:8080/api/v1/employees/1

```
{"id":1,"firstName":"om ","lastName":"maurya","emailId":"omsir@gmail.com"}
```