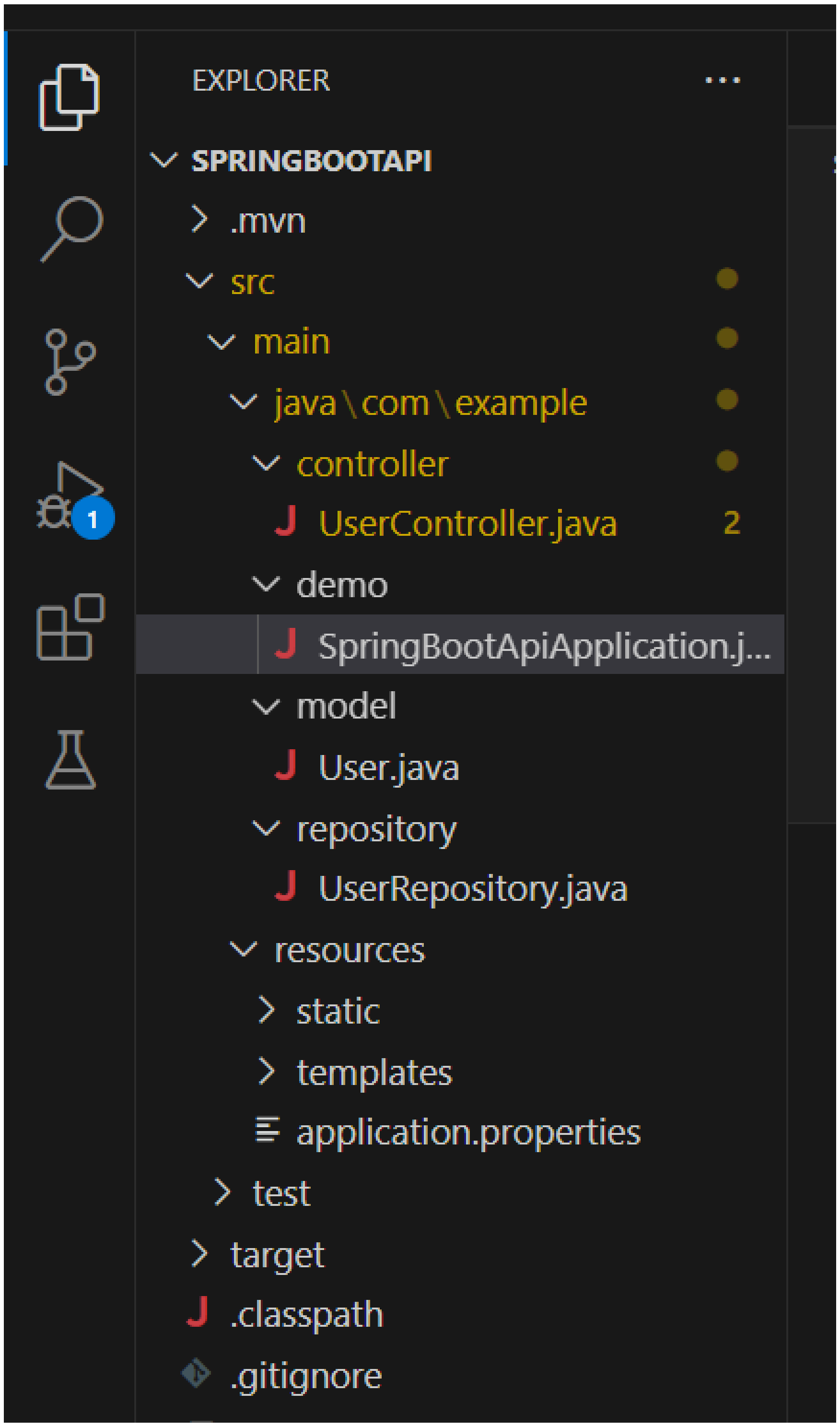
Spring boot rest Api Backend Development :-

Project structure will be like this –



```
SPRINGBOOTAPI
  .mvn
  src
    main
      java\com\example
        controller
          UserController.java   2
        demo
          SpringBootApiApplication.j...
        model
          User.java
        repository
          UserRepository.java
      resources
        static
        templates
        application.properties
    test
  target
  .classpath
  .gitignore
```

Write code for model User.java file code:-

```
package com.example.model;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="users")
public class User
{
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int userid;

    @Column
    private String username;
    private String email;
    private String password;
    private String role;

    public int getUserid() {
        return userid;
    }
    public void setUserid(int userid) {
        this.userid = userid;
    }
    public String getUsername() {
        return username;
    }
    public void setUsername(String username) {
        this.username = username;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
    public String getRole() {
        return role;
    }
    public void setRole(String role) {
        this.role = role;
    }



}
```

Now write code for `UserRepository.java` file code:-

```
package com.example.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
import com.example.model.User;

@Repository
public interface UserRepository extends JpaRepository<User, Integer>
{
    public User findByUsername(String name);
    public User findByEmail(String name);
    public User findByUserid(int id);
}
```

Now write code for controller class file `UserController.java` file:-

```
package com.example.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
```

```java
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.RequestParam;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import com.example.model.User;
import com.example.repository.UserRepository;

@RestController
@RequestMapping("/user")
@CrossOrigin(value = "http://localhost:4200/")
public class UserController
{
    @Autowired
    UserRepository userRepo;

    @GetMapping("/")
    public List<User> getUsers()
    {
        return userRepo.findAll();
    }

    @GetMapping("/username/{name}")
    public User getUsers(@PathVariable("name") String name)
    {
        return userRepo.findByUsername(name);
    }

    @GetMapping("/email/{name}")
    public User getEmail(@PathVariable("name") String name)
    {
        return userRepo.findByEmail(name);
    }

    @GetMapping("/id/{id}")
    public User getEmail(@PathVariable("id") int id)
    {
        return userRepo.findByUserid(id);
    }

    @PostMapping("/add")
    public void addUser(@RequestBody User u)
    {
        userRepo.save(u);
    }

    @PutMapping("/update")
    public void updateUser(@RequestBody User u)
    {
        userRepo.save(u);
    }

    @DeleteMapping("/delete/{id}")
    public void updateUser(@PathVariable("id") int id)
    {
        User u=new User();
        u.setUserid(id);
        userRepo.delete(u);
    }
}
```

Write   following code in application.properties file which you will find under resources folder

```
spring.datasource.url=jdbc:mysql://localhost:3306/mydb
spring.datasource.username=root
spring.datasource.password=
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

server.error.whitelabel.enabled=false
spring.jpa.hibernate.ddl-auto=update

spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
```

and your application code `SpringBootApiApplication.java  file`

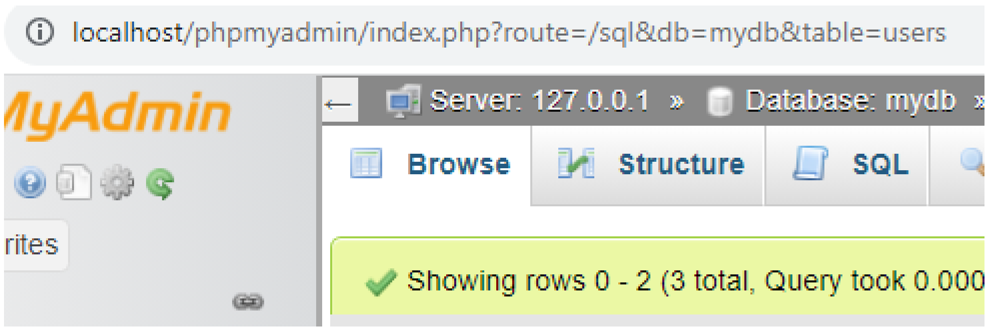
:-

```
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import org.springframework.context.annotation.ComponentScan;
```

```java
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication
@ComponentScan(basePackages = "com.example")
@EntityScan("com.example.model")
@EnableJpaRepositories("com.example.repository")
public class SpringBootApiApplication
{
    public static void main(String[] args)
    {
        SpringApplication.run(SpringBootApiApplication.class, args);
    }

}
```

Now install xampp and run it and

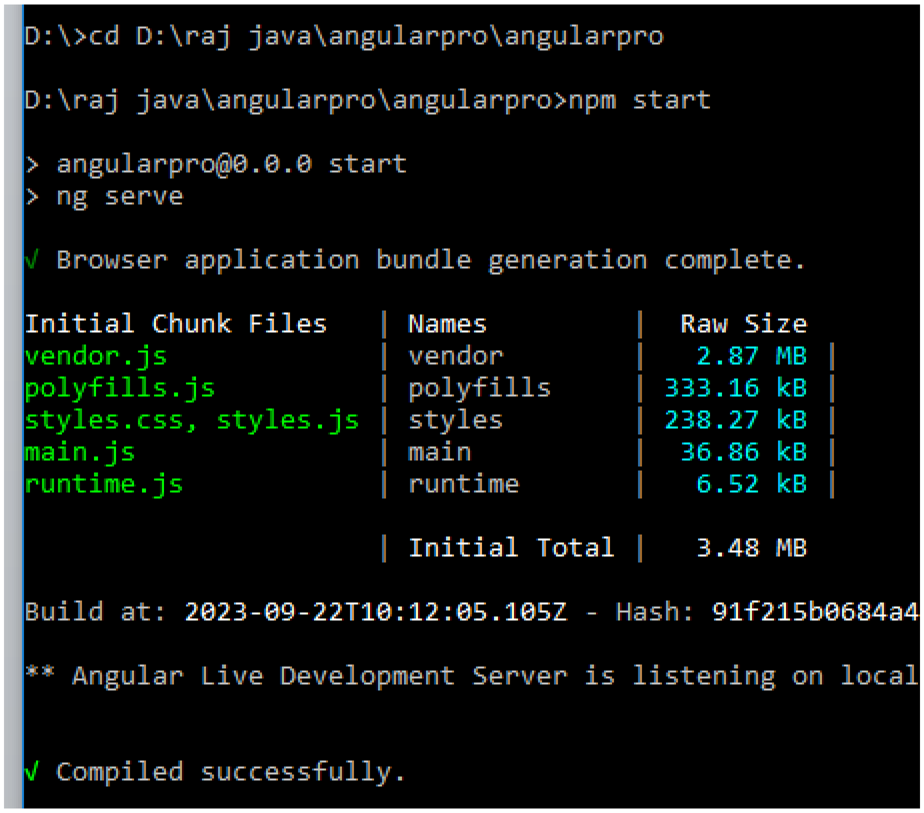
inside your "localhost/phpmyadmin" create database mydb as shown below



localhost/phpmyadmin/index.php?route=/sql&db=mydb&table=users

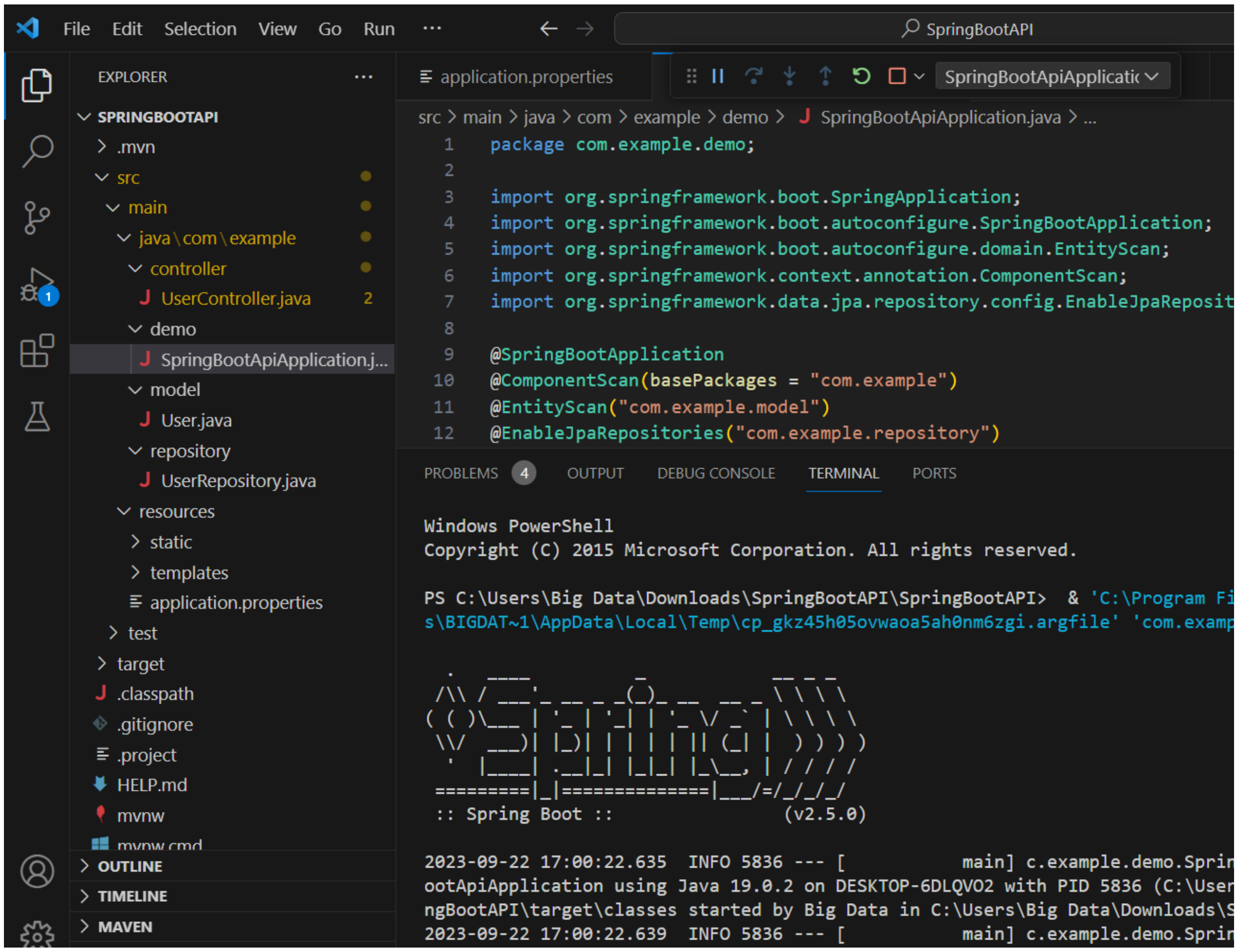
Showing rows 0 - 2 (3 total, Query took 0.000

Finally run your front end angular code



```
D:\>cd D:\raj java\angularpro\angularpro

D:\raj java\angularpro\angularpro>npm start

> angularpro@0.0.0 start
> ng serve

√ Browser application bundle generation complete.

Initial Chunk Files   | Names         |  Raw Size
vendor.js             | vendor        |   2.87 MB |
polyfills.js          | polyfills     | 333.16 kB |
styles.css, styles.js | styles        | 238.27 kB |
main.js               | main          |  36.86 kB |
runtime.js            | runtime       |   6.52 kB |

                      | Initial Total |   3.48 MB

Build at: 2023-09-22T10:12:05.105Z - Hash: 91f215b0684a4

** Angular Live Development Server is listening on local


√ Compiled successfully.
```

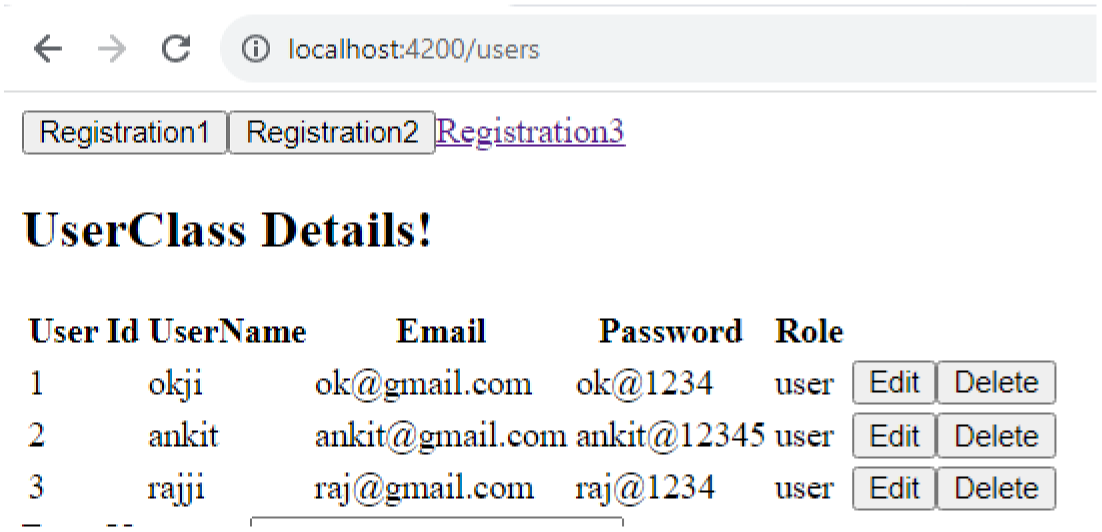
and back end spring boot code :-



```java
package com.example.demo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.data.jpa.repository.config.EnableJpaReposit

@SpringBootApplication
@ComponentScan(basePackages = "com.example")
@EntityScan("com.example.model")
@EnableJpaRepositories("com.example.repository")
```

```
Windows PowerShell
Copyright (C) 2015 Microsoft Corporation. All rights reserved.

PS C:\Users\Big Data\Downloads\SpringBootAPI\SpringBootAPI>  & 'C:\Program Fi
s\BIGDAT~1\AppData\Local\Temp\cp_gkz45h05ovwaoa5ah0nm6zgi.argfile' 'com.examp

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.5.0)

2023-09-22 17:00:22.635  INFO 5836 --- [           main] c.example.demo.Sprin
ootApiApplication using Java 19.0.2 on DESKTOP-6DLQVO2 with PID 5836 (C:\User
ngBootAPI\target\classes started by Big Data in C:\Users\Big Data\Downloads\S
2023-09-22 17:00:22.639  INFO 5836 --- [           main] c.example.demo.Sprin
```

And open your browser you will see



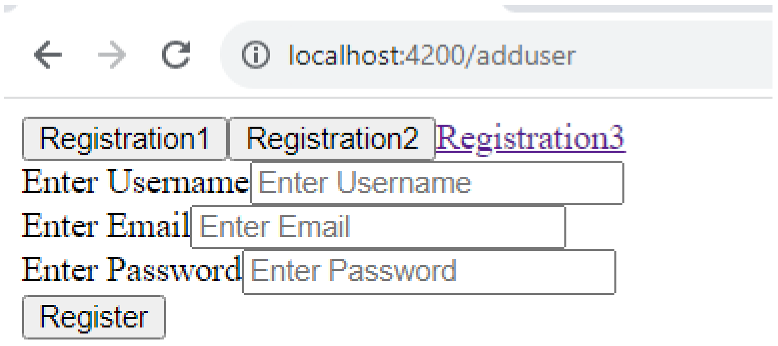
localhost:4200/users

Registration1 Registration2 Registration3

UserClass Details!

User Id	UserName	Email	Password	Role		
1	okji	ok@gmail.com	ok@1234	user	Edit	Delete
2	ankit	ankit@gmail.com	ankit@12345	user	Edit	Delete
3	rajji	raj@gmail.com	raj@1234	user	Edit	Delete

After click on Registration 1 you will see



localhost:4200/adduser

Registration1 Registration2 Registration3

Enter Username Enter Username
Enter Email Enter Email
Enter Password Enter Password
Register