Practical 1

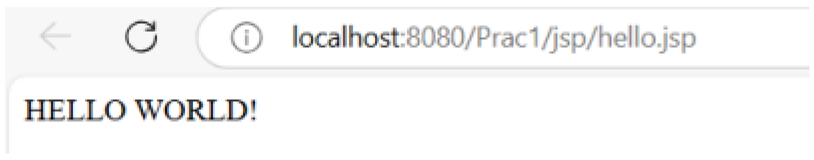
Aim: Connecting HTML & Java Server Pages (JSP) files

CODE :

Q1. Write a jsp program to print Hello World

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
        <head>
                <meta charset="ISO-8859-1">
        </head>
        <body>
                <%
                String f = "HELLO";
                String s = "WORLD!";
                out.println(f);
                %>
                <%=s %>
        </body>
</html>
```

OUTPUT -



```
localhost:8080/Prac1/jsp/hello.jsp

HELLO WORLD!
```

Q2. Write a jsp program to print the sum of the digits of a number.
Q3. sum of squares of digits.
Q4.to find the max of 3 numbers entered by the user.
Q5.write a jsp program which reverses the word entered by the user.

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="ISO-8859-1">
   <title>Number Operations</title>
</head>
<body>
   <!-- Sum of Digits -->
   <form method="post" action="sum.jsp">
      <label>Enter the number:</label>
      <input type="number" name="num1" required><br>
      <input type="submit" value="Calculate Sum">
   </form>
```

```
    <!-- Sum of Squares of Digits -->
    <form method="post" action="square.jsp">
        <label>Enter the number:</label>
        <input type="number" name="n" required><br>
        <input type="submit" value="Calculate Sum of Squares">
    </form>
    <!-- Find Maximum of Three Numbers -->
    <form method="post" action="max.jsp">
        <label>Enter num1:</label>
        <input type="number" name="n1" required><br>
        <label>Enter num2:</label>
        <input type="number" name="n2" required><br>
        <label>Enter num3:</label>
        <input type="number" name="n3" required><br>
        <input type="submit" value="Find Max">
    </form>
    <!-- Reverse a String -->
    <form method="post" action="reversetest.jsp">
        <label>Enter the string:</label>
        <input type="text" name="mystring" required><br>
        <input type="submit" value="Reverse String">
    </form>
</body>
</html>
```

<!-- reversetest.jsp -->

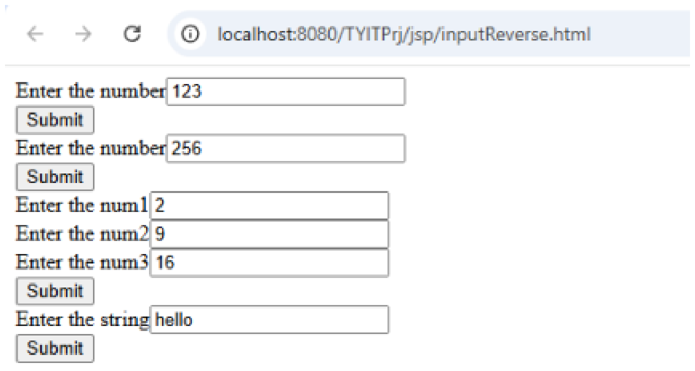
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Reversed String</title>
</head>
<body>
    <h2>Reversed String:</h2>
    <%
        String mystr = request.getParameter("mystring");
        StringBuffer sb = new StringBuffer(mystr);
        String rev = sb.reverse().toString();
        out.print("<strong>" + rev + "</strong>");
    %>
</body>
</html>
```

<!-- square.jsp -->

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
  <meta charset="ISO-8859-1">
  <title>Sum of Squares</title>
</head>
<body>
  <h2>Sum of Squares of Digits:</h2>
  <%
    int num = Integer.parseInt(request.getParameter("n"));
    int d, sum = 0, square;
    while (num != 0) {
      d = num % 10;
      square = d * d;
      sum += square;
      num /= 10;
    }
    out.print("<strong>" + sum + "</strong>");
  %>
</body>
</html>
<!-- max.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
  <meta charset="ISO-8859-1">
  <title>Maximum Number</title>
</head>
<body>
  <h2>Maximum of Three Numbers:</h2>
  <%
    int num1 = Integer.parseInt(request.getParameter("n1"));
    int num2 = Integer.parseInt(request.getParameter("n2"));
    int num3 = Integer.parseInt(request.getParameter("n3"));
    int max = Math.max(num1, Math.max(num2, num3));
    out.print("<strong>" + max + "</strong>");
  %>
</body>
</html>
<!-- sum.jsp -->
```

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="ISO-8859-1">
   <title>Sum of Digits</title>
</head>
<body>
   <h2>Sum of Digits:</h2>
   <%
      int num = Integer.parseInt(request.getParameter("num1"));
      int d, sum = 0;
      while (num != 0) {
         d = num % 10;
         sum += d;
         num /= 10;
      }
      out.print("<strong>" + sum + "</strong>");
   %>
</body>
</html>
```

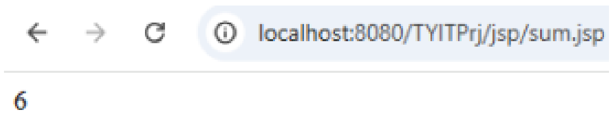
OUTPUT-



localhost:8080/TYITPrj/jsp/inputReverse.html

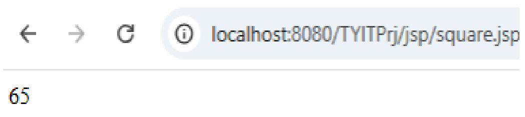
```
Enter the number 123
Submit
Enter the number 256
Submit
Enter the num1 2
Enter the num2 9
Enter the num3 16
Submit
Enter the string hello
Submit
```

Q2.   Q3.



localhost:8080/TYITPrj/jsp/sum.jsp

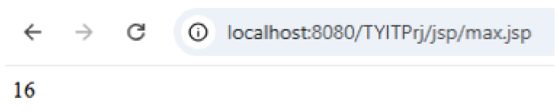
```
6
```



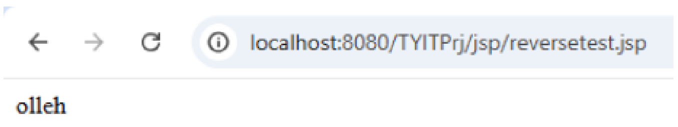
localhost:8080/TYITPrj/jsp/square.jsp

```
65
```

Q4.   Q5.



```
16
```



```
olleh
```

Learnings:

1. Download Apache Tomcat.
2. Install Eclipse IDE for Java EE.
3. Go to **Help > Eclipse Marketplace**, search for **Web Tools Platform**, and install it. Restart Eclipse after the installation.
4. Switch to the **Servers** view:
   - Go to **Window > Show View > Other**.
   - In the dialog, search for and select **Servers**, then click **OK**.
   - In the Servers tab, click **No servers are available. Click this link to create a new server...**
5. In the **Define a New Server** dialog:
   - Expand the **Apache** category.
   - Select the version of Tomcat you downloaded (e.g., **Apache Tomcat v10.0**) and click **Next**.
6. In the **Tomcat Installation Directory** field:
   - Browse to the location where you extracted the Tomcat files (e.g., **/path/to/tomcat**).
7. Click **Finish**.

—---

Practical 2

Aim: Connecting JSP page to Maria DB-1

CODE:
Home.jsp

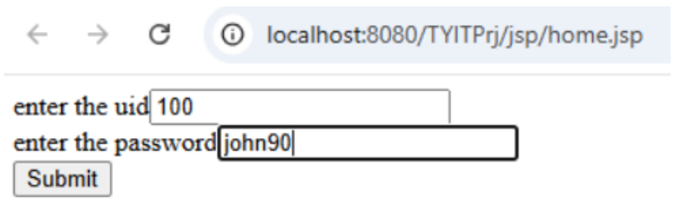
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="ISO-8859-1">
   <title>Insert title here</title>
</head>
<body>
   <form method="post" action="backend.jsp">
      <label>Username:</label>
```

```
        <input type="text" name="uname"><br><br>
        <label>Password:</label>
        <input type="text" name="pwd"><br><br>
        <input type="submit">
    </form>
</body>
</html>
```

Process.jsp

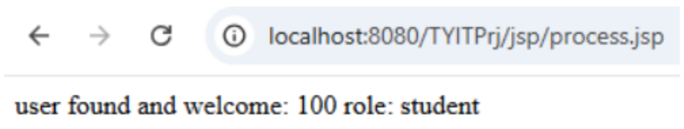
```
<%@ page language="java" import="java.sql.*, java.util.*" contentType="text/html;
charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Login Processing</title>
</head>
<body>
    <%
        boolean found = false;
        String role = null;
        String uid = request.getParameter("uid");
        int u = Integer.parseInt(uid);
        String passwd = request.getParameter("pwd")
        try {
            Class.forName("org.mariadb.jdbc.Driver");
            Connection cn =
DriverManager.getConnection("jdbc:mariadb://localhost:3306/mydb", "root",
"maria");
            PreparedStatement ps = cn.prepareStatement("SELECT * FROM
authentication WHERE uid=? AND password=?");
            ps.setInt(1, u);
            ps.setString(2, passwd);
            ResultSet rs = ps.executeQuery();
            while (rs.next()) {
                role = rs.getString("role");
                found = true;
            }
            if (found) {
                out.println("User found and welcome: " + uid + " Role: " + role);
            } else {
                out.println("You are not an authenticated user.");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    %>
</body>
</html>
```

OUTPUT:



localhost:8080/TYITPrj/jsp/home.jsp

```
enter the uid 100
enter the password john90
Submit
```



localhost:8080/TYITPrj/jsp/process.jsp

```
user found and welcome: 100 role: student
```

Learnings:

1. Download `mariadb-java-client-3.5.1.jar`.
2. Place it in the `lib` folder inside `WEB-INF` of the `webapp`.
3. Configure the project:
    - Go to **Project** → **Properties** → **Java Build Path** → **Libraries** → **Classpath**.
    - Click **Add External JARs** and select the downloaded `.jar` file.

—---

Practical 3

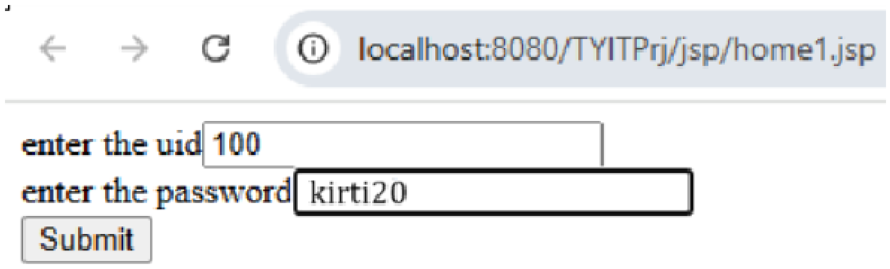
Aim: Connecting JSP page to Maria DB-2

CODE:

```
<!-- home1.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Insert title here</title>
</head>
<body>
    <form method="post" action="result.jsp">
```

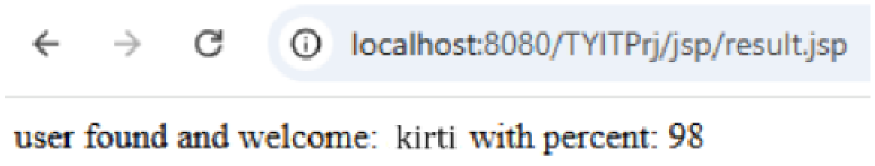
```
        Enter the UID: <input type="text" name="uid"><br><br>
        Enter the Password: <input type="text" name="pwd"><br><br>
        <input type="submit"><br>
    </form>
</body>
</html>
<!-- result.jsp -->
<%@ page language="java" import="java.sql.*, java.util.*" contentType="text/html;
charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Insert title here</title>
</head>
<body>
<%
    boolean found = false;
    String name = null;
    int percent = 0;
    String uid = request.getParameter("uid");
    int u = Integer.parseInt(uid);
    String passwd = request.getParameter("pwd");
    try {
        Class.forName("org.mariadb.jdbc.Driver");
        Connection cn =
DriverManager.getConnection("jdbc:mariadb://localhost:3306/mydb", "root",
"maria");
        PreparedStatement ps = cn.prepareStatement("SELECT * FROM marks WHERE
uid=? AND password=?");
        ps.setInt(1, u);
        ps.setString(2, passwd);
        ResultSet rs = ps.executeQuery();
        while (rs.next()) {
            name = rs.getString("name");
            percent = rs.getInt("percent");
            found = true;
        }
        if (found) {
            out.println("User found and welcome: " + name + " with percent: " + percent);
        } else {
            out.println("You are not an authenticated user");
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
%>
</body>
</html>
```

OUTPUT :



localhost:8080/TYITPrj/jsp/home1.jsp

```
enter the uid 100
enter the password kirti20
Submit
```



localhost:8080/TYITPrj/jsp/result.jsp

```
user found and welcome:  kirti with percent: 98
```

Learnings:

1. Data Retrieval: Fetch user details from the database using SQL queries.
2. Result Display: Dynamically show user-specific data with JSP output.

—--

Practical 4

Aim: Retrieving information from jsp registration page

CODE:

<!-- home.jsp -->

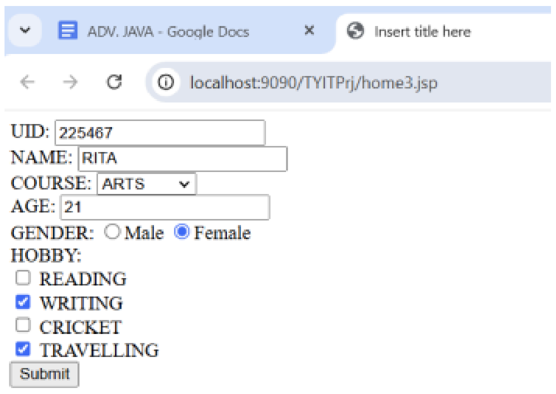
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Insert title here</title>
</head>
<body>
    <form action="processregistration3.jsp" method="post">
        UID: <input type="text" name="uid"><br>
        NAME: <input type="text" name="name"><br>
        COURSE:
        <select name="course">
```

```
            <option>ARTS</option>
            <option>BCOM</option>
            <option>SCIENCE</option>
        </select><br>
        AGE: <input type="text" name="age"><br>
        GENDER:
        <input type="radio" name="gender" value="male">Male
        <input type="radio" name="gender" value="female">Female<br>
        HOBBY: <br>
        <input type="checkbox" id="hobby1" name="hobby" value="reading">
        <label for="hobby1">READING</label><br>
        <input type="checkbox" id="hobby2" name="hobby" value="writing">
        <label for="hobby2">WRITING</label><br>
        <input type="checkbox" id="hobby3" name="hobby" value="cricket">
        <label for="hobby3">CRICKET</label><br>
        <input type="checkbox" id="hobby4" name="hobby" value="travelling">
        <label for="hobby4">TRAVELLING</label><br>
        <input type="submit">
    </form>
</body>
</html>
```

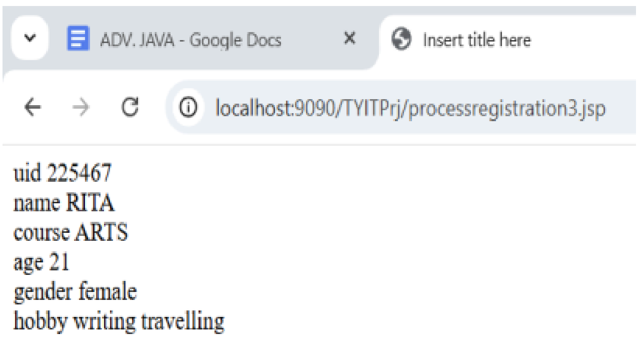
<!-- processregistration.jsp -->

```
<%@ page language="java" import="java.util.*" contentType="text/html;
charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Insert title here</title>
</head>
<body>
    <%
        Enumeration e = request.getParameterNames();
        while (e.hasMoreElements()) {
            String output = null;
            String p = (String) e.nextElement();
            // out.print(p+" ");
            if (p.equals("hobby")) {
                String[] h = request.getParameterValues(p);
                out.print(p);
                for (String s : h) {
                    out.print(" " + s);
                }
            } else {
                output = request.getParameter(p);
                out.println(p + " " + output + "<br>");
            }
        }
    %>
</body>
</html>
```

OUTPUT:



```
UID: 225467
NAME: RITA
COURSE: ARTS
AGE: 21
GENDER: ○ Male ● Female
HOBBY:
☐ READING
☑ WRITING
☐ CRICKET
☑ TRAVELLING
Submit
```



```
uid 225467
name RITA
course ARTS
age 21
gender female
hobby writing travelling
```

Learning:

1. Form Handling: Use request.getParameterNames() to process dynamic inputs.
2. Checkbox Data: Retrieve multiple values with request.getParameterValues().

—---

Practical 5

Aim: Implementing an AJAX-based communication between a client-side and a server-side JSP page

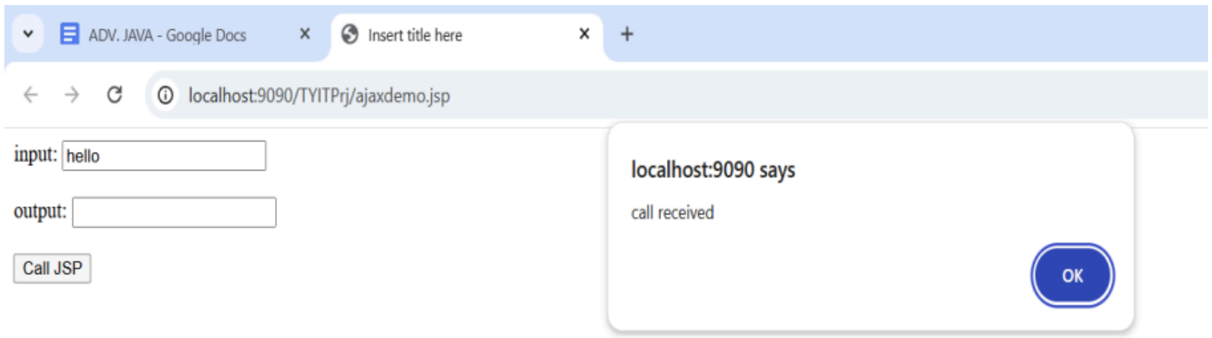
CODE:

```
<!-- ajaxdemo.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/1.4/jquery.min.js"></script>
    <script type="text/javascript">
        function check() {
            alert("call received");
            $.ajax({
```

```
            type: "post",
            url: "ajaxprocess.jsp", // here you can use servlet, jsp, etc
            data: "input=" + $('#ip').val() + "&output=" + $('#op').val(),
            success: function (msg) {
                $('#output').append(msg);
            },
            error: function () {
                alert("error in alert");
            }
        });
    }
  </script>
  <meta charset="ISO-8859-1">
  <title>Insert title here</title>
</head>
<body>
  input: <input type="text" id="ip" name="ip" value="" /><br><br>
  output: <input type="text" id="op" name="op" value=""><br><br>
  <input type="button" onclick="check()" value="Call JSP" name="Call JSP"
id="call">
  <div id="output"></div>
</body>
</html>
<!-- ajaxprocess.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
  pageEncoding="ISO-8859-1" import="org.json.simple.JSONObject"%>
<!DOCTYPE html>
<html>
<head>
  <meta charset="ISO-8859-1">
  <title>Insert title here</title>
</head>
<body>
  <%
    System.out.print("call received");
    String flag = "working";
    String s = request.getParameter("input");
    out.println(flag);
    JSONObject jsonResponse = new JSONObject();
    jsonResponse.put("name", flag + s);
    System.out.print(jsonResponse);
    response.getWriter().write(jsonResponse.toString());
    out.flush();
  %>
```

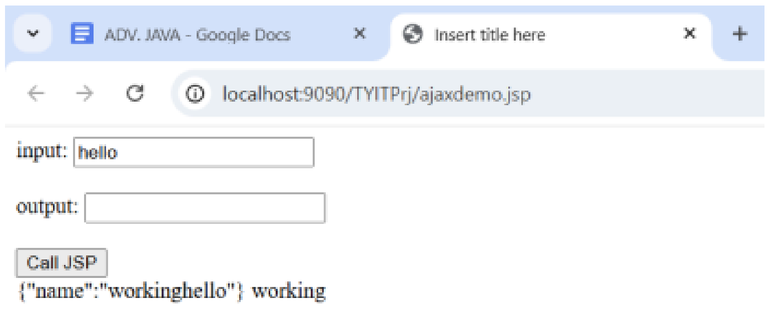
```
</body>
</html>
```

OUTPUT:



```
localhost:9090/TYITPrj/ajaxdemo.jsp
input: hello
output:
Call JSP

localhost:9090 says
call received
OK
```



```
localhost:9090/TYITPrj/ajaxdemo.jsp
input: hello
output:
Call JSP
{"name":"workinghello"} working
```

Learning:

1. AJAX Integration: Use jQuery $.ajax() for asynchronous server communication.
2. JSON Handling: Send JSON responses from JSP using JSONObject.

—---

Practical 6

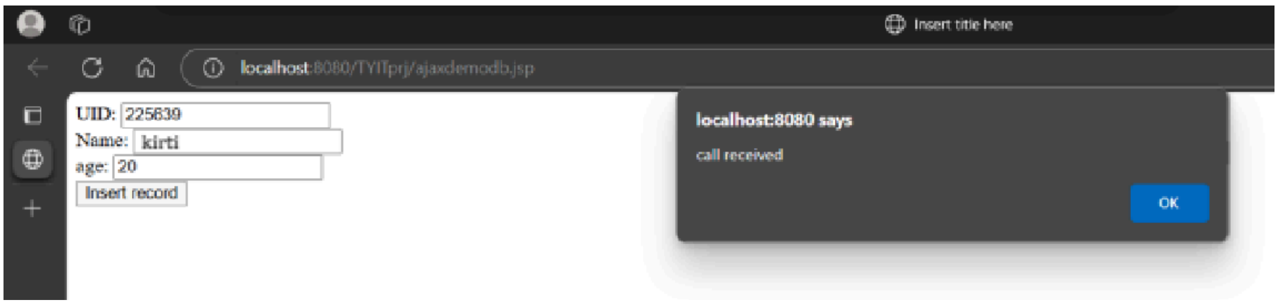
Aim: Implementation of ajax and database

CODE:

<!-- ajaxdemodb.jsp -->

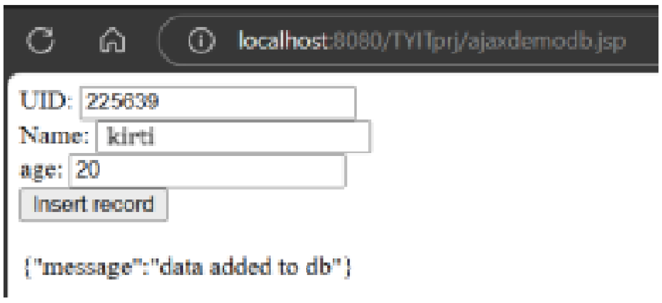
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
   <meta charset="ISO-8859-1">
   <title>Insert title here</title>
   <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
   <script type="text/javascript">
```

```
        function check() {
            alert("call received");
            $.ajax({
                type: "post",
                url: "ajaxprocessdb.jsp",
                data: "uid=" + $("#uid").val() + "&name=" + $("#name").val() + "&age=" +
$("#age").val(),
                success: function(result) {
                    $("#output").append(result);
                },
                error: function() {
                    alert("error in alert");
                }
            });
        }
    </script>
</head>
<body>
    UID: <input id="uid" type="text" name="uid" value="" /> <br />
    Name: <input id="name" type="text" name="name" value="" /> <br />
    Age: <input id="age" type="text" name="age" value="" /> <br />
    <input type="button" onClick="check()" value="Insert record" />
    <br /><br />
    <div id="output"></div>
</body>
</html>
<!-- ajaxprocessdb.jsp -->
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1" import="java.sql.*, org.json.simple.JSONObject"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Insert title here</title>
</head>
<body>
    <%
        int uid = Integer.parseInt(request.getParameter("uid"));
        String name = request.getParameter("name");
        int age = Integer.parseInt(request.getParameter("age"));
        JSONObject jsonResponse = new JSONObject();
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/db", "root", "root");
            PreparedStatement ps = connection.prepareStatement("INSERT INTO
students VALUES (?, ?, ?)");
            ps.setInt(1, uid);
            ps.setString(2, name);
            ps.setInt(3, age);
            int rows = ps.executeUpdate();
            if (rows > 0) {
                jsonResponse.put("message", "Data added to DB");
            } else {
                jsonResponse.put("message", "Data not added to DB");
            }
        } catch (Exception e) {
            System.out.print("Error occurred due to " + e.toString());
            e.printStackTrace();
            jsonResponse.put("message", "Error occurred while inserting data");
        }
        response.getWriter().write(jsonResponse.toString());
        out.flush(); // returns to first page
    %>
</body>
</html>
```
OUTPUT:


localhost:8080/TYITprj/ajaxdemodb.jsp

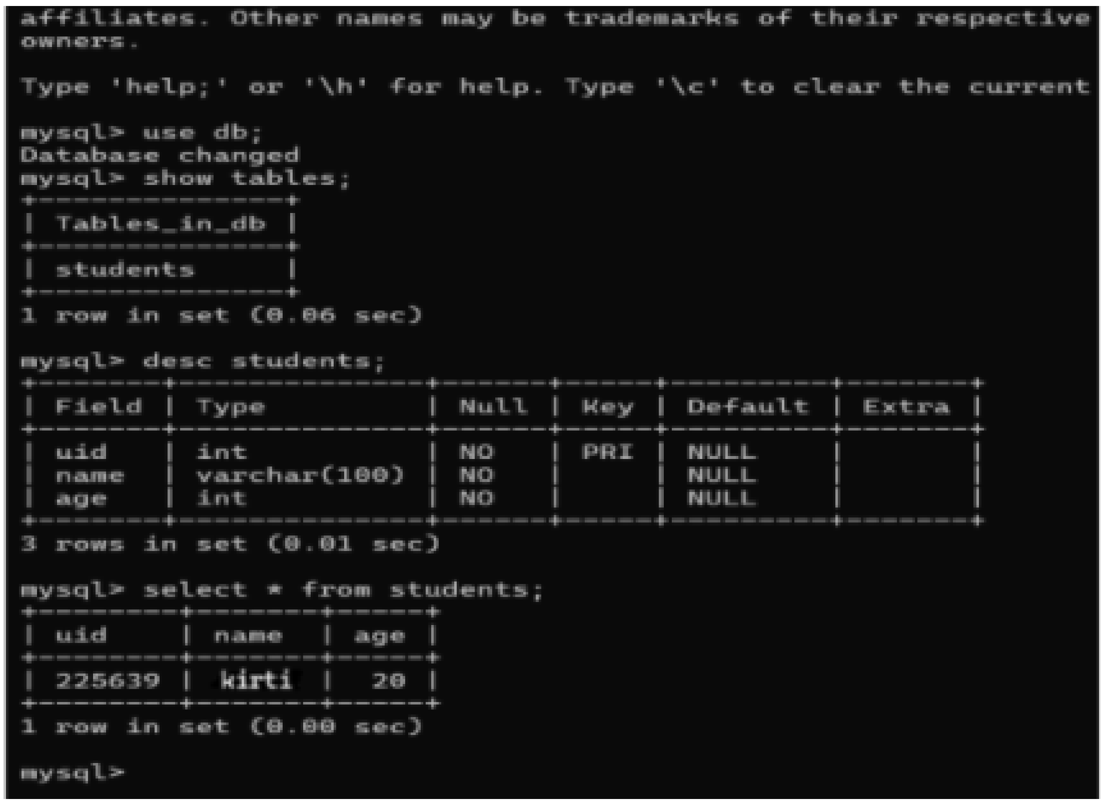
UID: 225639
Name: kirti
age: 20
Insert record

localhost:8080 says
call received
OK


localhost:8080/TYITprj/ajaxdemodb.jsp

UID: 225639
Name: kirti
age: 20
Insert record

{"message":"data added to db"}



```
affiliates. Other names may be trademarks of their respective
owners.

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

mysql> use db;
Database changed
mysql> show tables;
+---------------+
| Tables_in_db |
+---------------+
| students      |
+---------------+
1 row in set (0.06 sec)

mysql> desc students;
+-------+--------------+------+-----+---------+-------+
| Field | Type         | Null | Key | Default | Extra |
+-------+--------------+------+-----+---------+-------+
| uid   | int          | NO   | PRI | NULL    |       |
| name  | varchar(100) | NO   |     | NULL    |       |
| age   | int          | NO   |     | NULL    |       |
+-------+--------------+------+-----+---------+-------+
3 rows in set (0.01 sec)

mysql> select * from students;
+--------+-------+-----+
| uid    | name  | age |
+--------+-------+-----+
| 225639 | kirti |  20 |
+--------+-------+-----+
1 row in set (0.00 sec)

mysql>
```

Learning:

1. AJAX simplifies data submission without reloading the page, improving user experience.
2. Proper error handling is crucial to debug database connectivity and insertion issues.

—---

Practical 7

Aim: REST (user)

CODE:

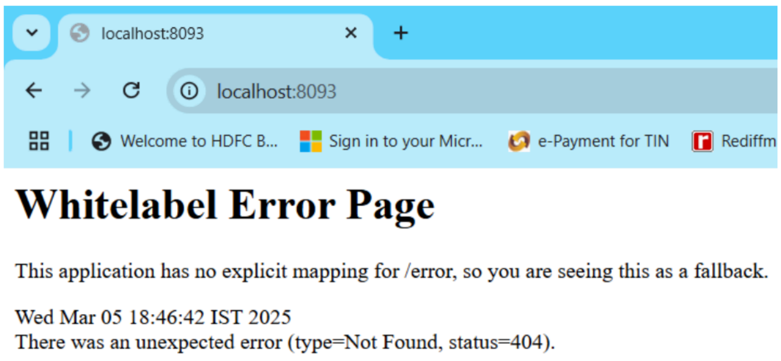
```
// User.java
package com.example.demoUser;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.Data;
@Entity
@Data
public class User {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
    private Long id;
    @Column(name = "firstName", nullable = true)
    private String firstName;
    @Column(name = "lastName", nullable = true)
    private String lastName;
    @Column(name = "email", nullable = true)
    private String email;
    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 getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
}
// UserController.java
package com.example.demoUser;
import java.util.List;
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.RequestBody;
import org.springframework.web.bind.annotation.RestController;
@RestController
@Controller
public class UserController {
    @Autowired
    private UserService service;
    @PostMapping("/user")
    public User createUser(@RequestBody User user) {
        return service.createUser(user);
    }
    @GetMapping("/users/{userId}")
    public User getUserById(Long userId) {
        return service.getUserById(userId);
    }
    @GetMapping("/users")
    public List<User> getAllUsers() {
        return service.getAllUsers();
    }
}
// UserRepository.java
package com.example.demoUser;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
@Repository
public interface UserRepository extends JpaRepository<User, Long> {
}
// UserService.java
package com.example.demoUser;
import java.util.List;
import org.springframework.stereotype.Service;
@Service
public interface UserService {
    User createUser(User user);
    User getUserById(Long userId);
    List<User> getAllUsers();
    User saveUser(User user);
    public List<User> saveUser(List<User> user);
}
// UserServiceImpl.java
package com.example.demoUser;
import java.util.List;
import java.util.Optional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import lombok.AllArgsConstructor;
@Service
@AllArgsConstructor
public class UserServiceImpl implements UserService {
    @Autowired
    private UserRepository userRepository;
    @Override
    public User createUser(User user) {
        // TODO Auto-generated method stub
        return userRepository.save(user);
    }
    @Override
    public User getUserById(Long userId) {
        // TODO Auto-generated method stub
        Optional<User> optionalUser = userRepository.findById(userId);
        return optionalUser.get();
    }
    @Override
    public List<User> getAllUsers() {
        // TODO Auto-generated method stub
        return userRepository.findAll();
    }
    @Override
    public User saveUser(User user) {
        // TODO Auto-generated method stub
        return userRepository.save(user);
    }
    @Override
    public List<User> saveUser(List<User> user) {
        // TODO Auto-generated method stub
        return userRepository.saveAll(user);
    }
}
```
# application.properties
```
spring.application.name=demoUser
server.port=8093
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/db2
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.hibernate.naming.implicit-strategy=org.hibernate.boot.model.naming.Imp
licitNamingStrategyLegacyJpaImpl
spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.Phy
sicalNamingStrategyStandardImpl
```

// DemoUserApplication.java

```java
package com.example.demoUser;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class DemoUserApplication {
   public static void main(String[] args) {
      SpringApplication.run(DemoUserApplication.class, args);
   }
}
```

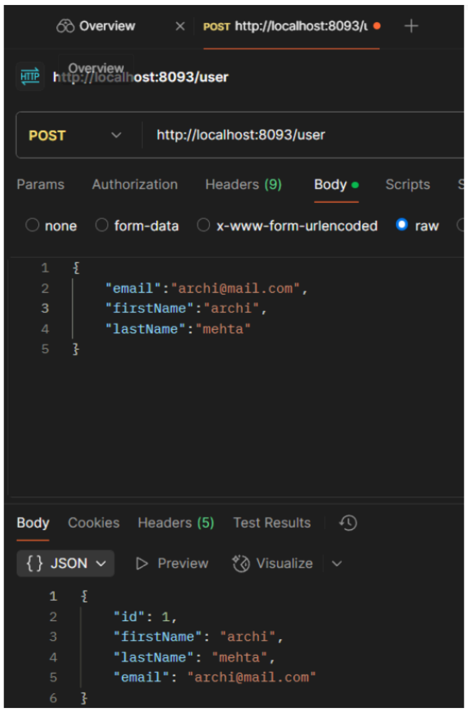
OUTPUT:



Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Wed Mar 05 18:46:42 IST 2025
There was an unexpected error (type=Not Found, status=404).



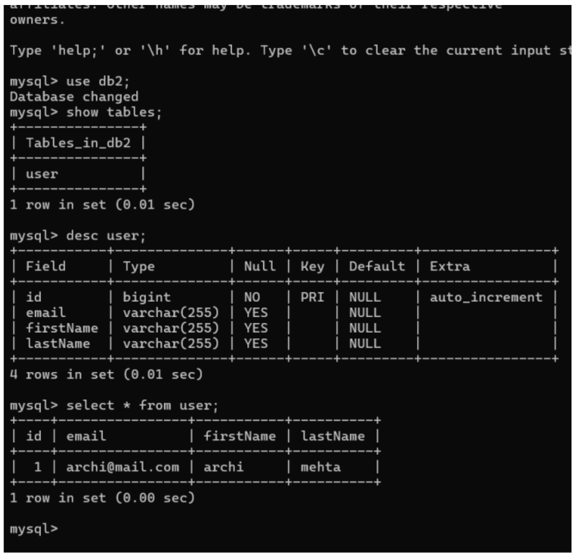
POST http://localhost:8093/user

Body raw

```json
{
    "email":"archi@mail.com",
    "firstName":"archi",
    "lastName":"mehta"
}
```

Body JSON

```json
{
    "id": 1,
    "firstName": "archi",
    "lastName": "mehta",
    "email": "archi@mail.com"
}
```



```
owners.

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

mysql> use db2;
Database changed
mysql> show tables;
+----------------+
| Tables_in_db2 |
+----------------+
| user           |
+----------------+
1 row in set (0.01 sec)

mysql> desc user;
+-----------+--------------+------+-----+---------+----------------+
| Field     | Type         | Null | Key | Default | Extra          |
+-----------+--------------+------+-----+---------+----------------+
| id        | bigint       | NO   | PRI | NULL    | auto_increment |
| email     | varchar(255) | YES  |     | NULL    |                |
| firstName | varchar(255) | YES  |     | NULL    |                |
| lastName  | varchar(255) | YES  |     | NULL    |                |
+-----------+--------------+------+-----+---------+----------------+
4 rows in set (0.01 sec)

mysql> select * from user;
+----+----------------+-----------+----------+
| id | email          | firstName | lastName |
+----+----------------+-----------+----------+
|  1 | archi@mail.com | archi     | mehta    |
+----+----------------+-----------+----------+
1 row in set (0.00 sec)

mysql>
```

Learning:

1. Spring Boot simplifies RESTful API creation with JPA, making database interactions seamless.
2. Proper dependency injection ensures efficient service-layer logic and repository management.

Practical 8

Aim: REST (customer)

CODE:

```
// Customer.java
package com.example.spring_practice;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import lombok.Data;
@Entity
@Data
public class Customer {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   private Long id;
   @Column(name = "customerName", nullable = true)
   private String customerName;
   @Column(name = "email", nullable = true)
   private String email;
   public Long getId() {
      return id;
   }
   public void setId(Long id) {
      this.id = id;
   }
   public String getCustomerName() {
      return customerName;
   }
   public void setCustomerName(String customerName) {
      this.customerName = customerName;
   }
   public String getEmail() {
      return email;
   }
   public void setEmail(String email) {
      this.email = email;
   }
}
// CustomerController.java
package com.example.spring_practice;
```

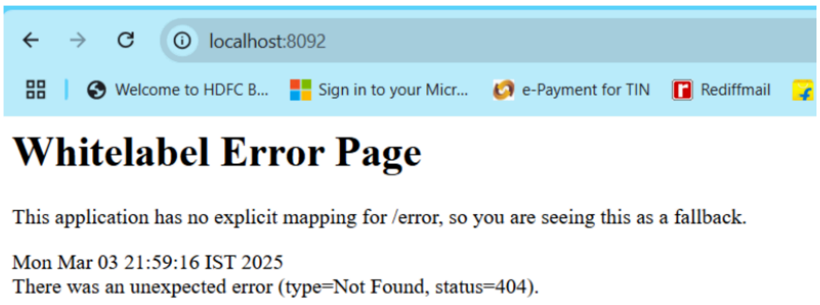
```
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
@RestController
@RequestMapping("/customers")
public class CustomerController {
   @Autowired
   private CustomerService service;
   @PostMapping
   public Customer createCustomer(@RequestBody Customer customer) {
      return service.createCustomer(customer);
   }
   @GetMapping("/{customerId}")
   public Customer getCustomerById(@PathVariable Long customerId) {
      return service.getCustomerById(customerId);
   }
   @GetMapping
   public List<Customer> getAllCustomers() {
      return service.getAllCustomers();
   }
}
// CustomerRepository.java
package com.example.spring_practice;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
@Repository
public interface CustomerRepository extends JpaRepository<Customer, Long> {
}
// CustomerService.java
package com.example.spring_practice;
import java.util.List;
import org.springframework.stereotype.Service;
@Service
public interface CustomerService {
   Customer createCustomer(Customer customer);
   Customer getCustomerById(Long customerId);
   List<Customer> getAllCustomers();
   Customer saveCustomer(Customer customer);
   List<Customer> saveCustomer(List<Customer> customers);
}
// CustomerServiceImpl.java
package com.example.spring_practice;
import java.util.List;
import java.util.Optional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import lombok.AllArgsConstructor;
@Service
@AllArgsConstructor
public class CustomerServiceImpl implements CustomerService {
   @Autowired
   private CustomerRepository customerRepository;
   @Override
   public Customer createCustomer(Customer customer) {
      return customerRepository.save(customer);
   }
   @Override
   public Customer getCustomerById(Long customerId) {
      Optional<Customer> optionalCustomer =
customerRepository.findById(customerId);
      return optionalCustomer.orElse(null);
   }
   @Override
   public List<Customer> getAllCustomers() {
      return customerRepository.findAll();
   }
   @Override
   public Customer saveCustomer(Customer customer) {
      return customerRepository.save(customer);
   }
   @Override
   public List<Customer> saveCustomer(List<Customer> customers) {
      return customerRepository.saveAll(customers);
   }
}
// application.properties
spring.application.name=spring_practice
server.port=8092
spring.datasource.url=jdbc:mysql://${MYSQL_HOST:localhost}:3306/db_practice
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.hibernate.naming.implicit-strategy=org.hibernate.boot.model.naming.Imp
licitNamingStrategyLegacyJpaImpl
```

spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

// SpringPracticeApplication.java

```
package com.example.spring_practice;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class SpringPracticeApplication {
   public static void main(String[] args) {
      SpringApplication.run(SpringPracticeApplication.class, args);
   }
}
```

OUTPUT:

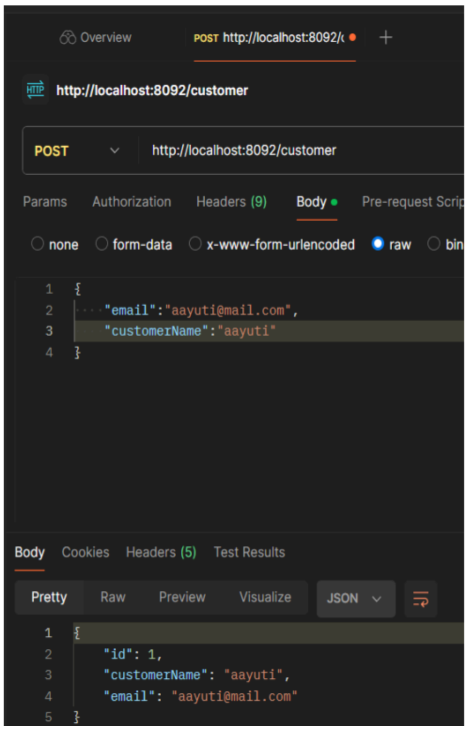


localhost:8092

Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

Mon Mar 03 21:59:16 IST 2025
There was an unexpected error (type=Not Found, status=404).

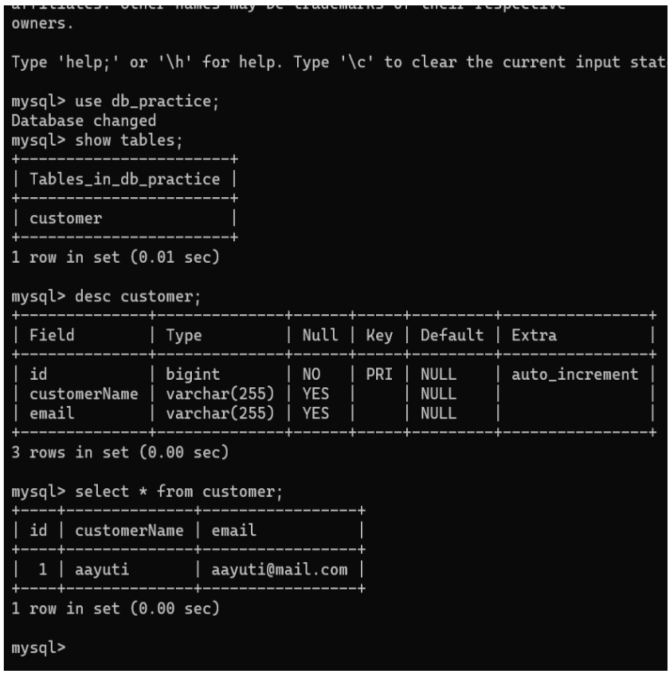


http://localhost:8092/customer

POST http://localhost:8092/customer

```
{
    "email":"aayuti@mail.com",
    "customerName":"aayuti"
}
```

```
{
    "id": 1,
    "customerName": "aayuti",
    "email": "aayuti@mail.com"
}
```



```
owners.

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

mysql> use db_practice;
Database changed
mysql> show tables;
+-----------------------+
| Tables_in_db_practice |
+-----------------------+
| customer              |
+-----------------------+
1 row in set (0.01 sec)

mysql> desc customer;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| id           | bigint       | NO   | PRI | NULL    | auto_increment |
| customerName | varchar(255) | YES  |     | NULL    |                |
| email        | varchar(255) | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
3 rows in set (0.00 sec)

mysql> select * from customer;
+----+--------------+-----------------+
| id | customerName | email           |
+----+--------------+-----------------+
|  1 | aayuti       | aayuti@mail.com |
+----+--------------+-----------------+
1 row in set (0.00 sec)

mysql>
```

Learning:

1. Proper naming conventions improve readability and avoid issues like inconsistent variable casing.
2. Using Lombok annotations (@Data) reduces boilerplate code for getters, setters, and constructors.