Learning Consolidation

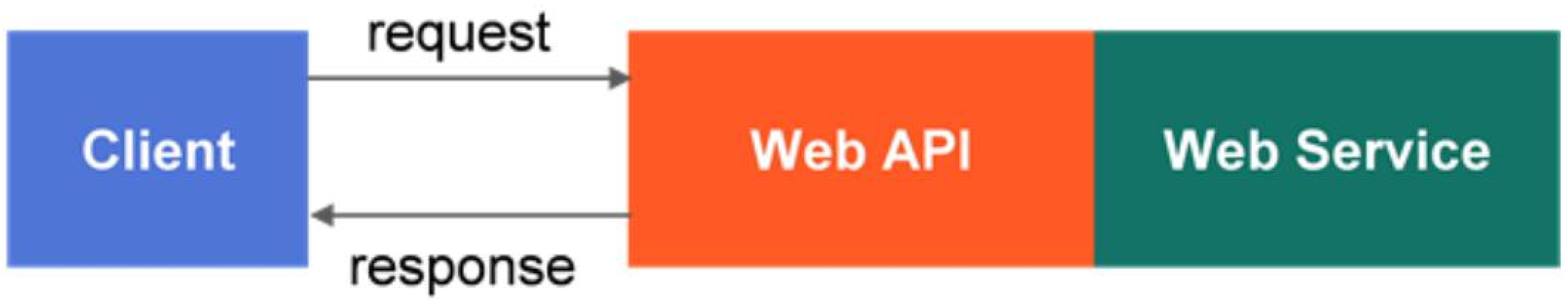
Develop RESTful Services by Using Spring Boot by Using JPA

Learning Objectives

- Describe REST API
- Explore applications of RESTful services
- Define components of RESTful services
- Implement layers of the RESTful API

Web Applications and Services

- Nowadays, a client-server model is used to make web applications.
- The web application is software that runs on a web server to fulfill requests from the clients.
- Clients use application programming interfaces (APIs) to communicate with web services.
- APIs expose a set of data and functions to facilitate interactions between computer programs and allow them to exchange information.
- A Web API is the face of a web service, directly listening to and responding to client requests.



REST – Representational State Transfer

- REST is an acronym for **Re**presentational **S**tate **T**ransfer
- It is a design pattern or architectural style for designing web APIs.
- REST relies on HTTP, and therefore it is stateless.
- The client makes an HTTP request to RESTful API for requested information referred to as a **resource**.
  - For e.g., products, orders, and shipping addresses are examples of resources on an online shopping website.
- At a given instance, the values of the resource determine its state.
  - For e.g., if a user has successfully logged in, their state of login property would be logged in, or else it is logged out.
- When RESTful API is called, the server sends the request or data to the client usually in JSON (language independent) text-based format.
  - For e.g., when a client requests an order delivery status, the server transfers the current state of the delivery with track details.

Need of REST API

- Unlike other architectures, REST clearly defines the role of a client and a server.
- The user interface is separated from database services.
- Developers can focus on only one aspect when developing client-server applications.
- REST is independent of the underlying platform.
- Any platform such as PHP, Python, and Node.js can be used to implement a REST API.
- REST allows communication between the server and the client by using HTTP, regardless of the platform used..
- REST APIs are highly scalable and flexible.

Components of RESTful Services

Resources

Request Verbs

Request Headers

Request Body

Response Status Code

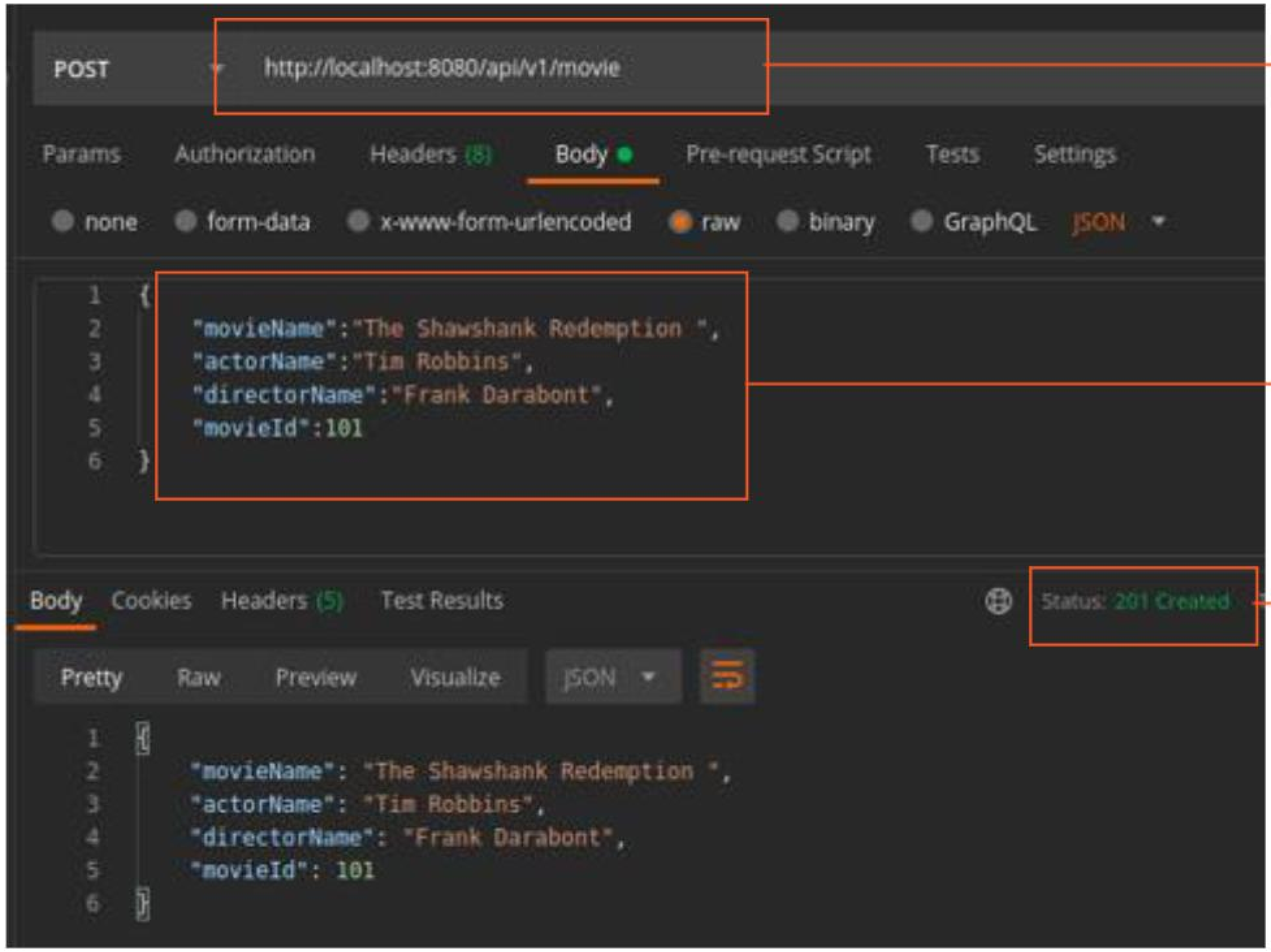
HTTP Status Code

- HTTP response status codes indicate the status of the completion of the HTTP request.
- Responses are grouped into five classes:
  - 100 – 199: Informational responses
    - For e.x., the status code 102 indicates the server has received and is processing the request. The response has not yet been generated.
  - 200 – 299: Successful responses
    - For e.g., the status code 200 for a Get request indicates the request was successful in fetching the requested data.

HTTP Status Code (cont'd)

- 300 – 399: Redirection responses
  - For e.x., the status code 301 indicates the request has been changed permanently, and a new URL is given in the response.
- 400 – 499: Client errors
  - For e.x., the status code 404 indicates the server could not find the requested resource.
- 500 – 599: Server errors
  - For e.x., the status code 500 indicates an internal server error that the server does not know how to handle. For instance, a resource with a duplicate ID is sent to the server, and the server is not allowed to store a resource with a duplicate ID.

Request Headers and Body



http://localhost:8080/api/v1/movie

```
{
   "movieName":"The Shawshank Redemption ",
   "actorName":"Tim Robbins",
   "directorName":"Frank Darabont",
   "movieId":101
}
```

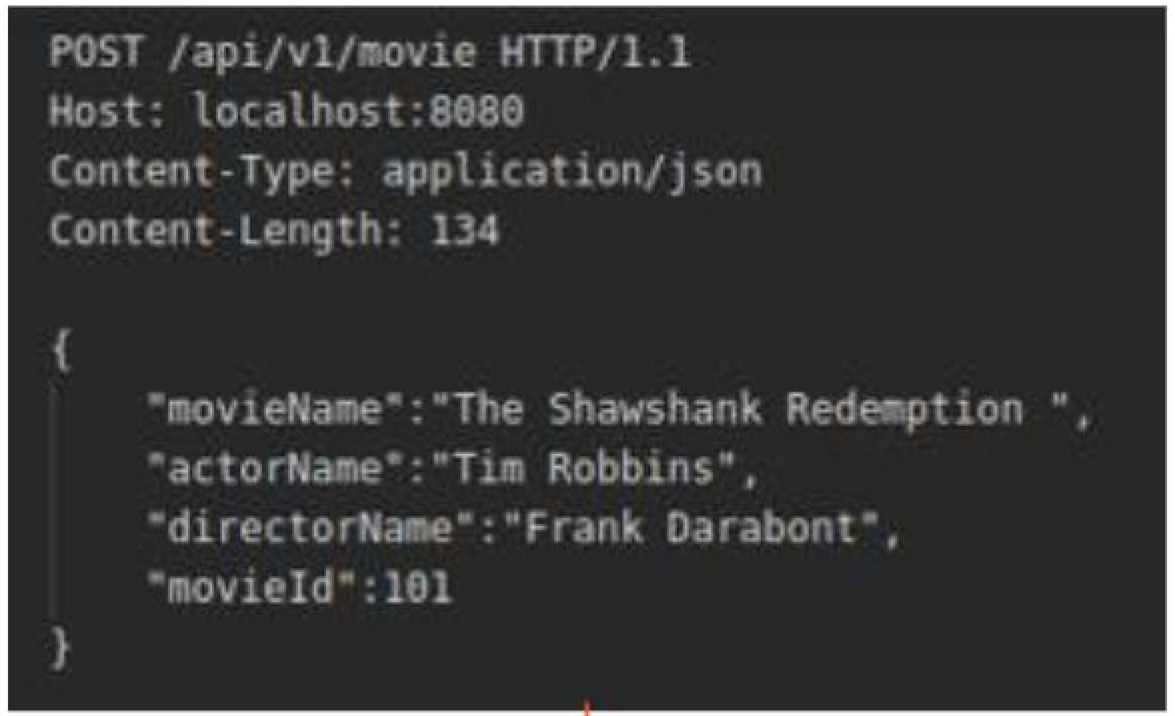
Status: 201 Created

```
{
   "movieName": "The Shawshank Redemption ",
   "actorName": "Tim Robbins",
   "directorName": "Frank Darabont",
   "movieId": 101
}
```

URL

Request Body

HTTP Status code

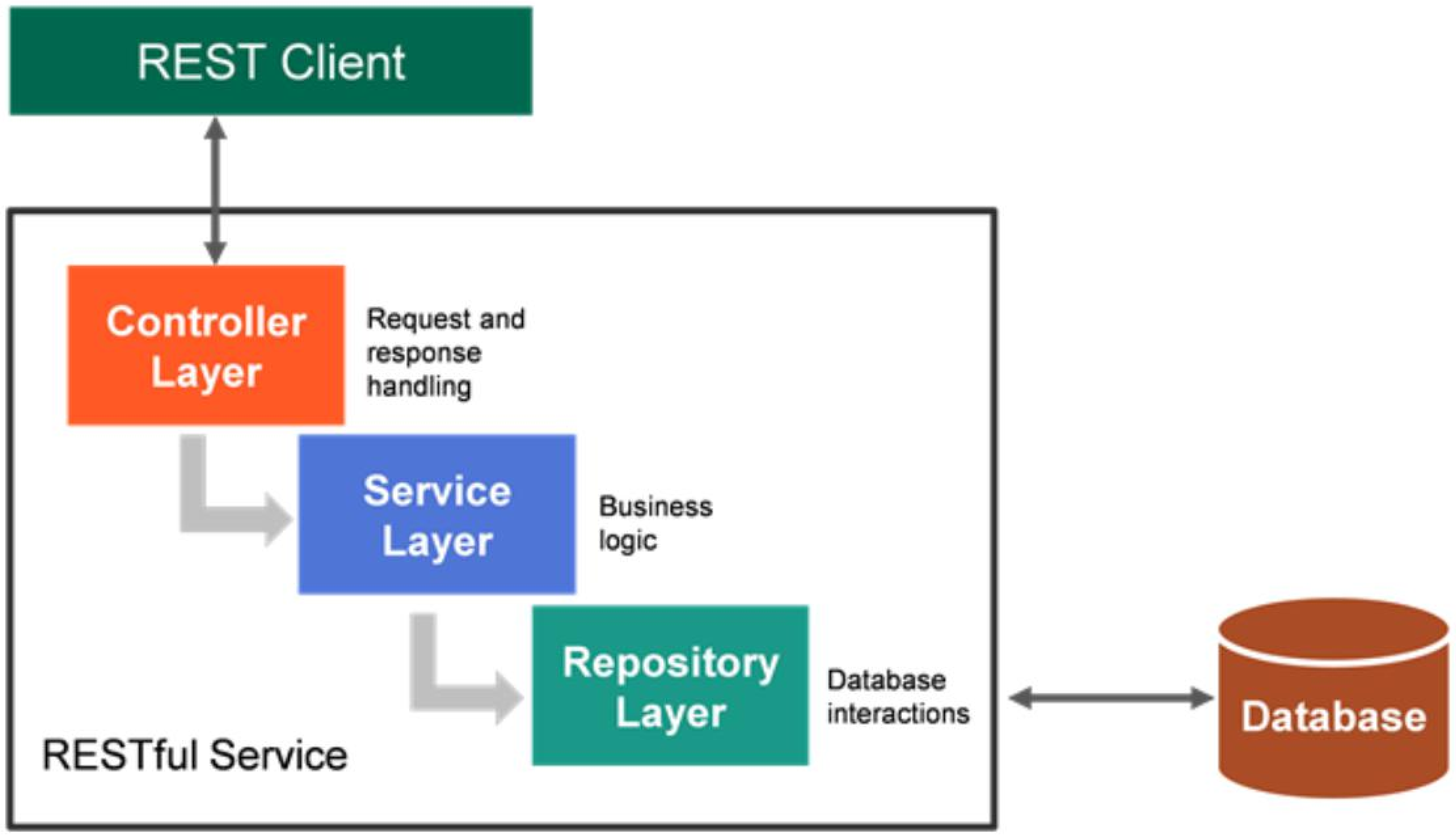


```
POST /api/v1/movie HTTP/1.1
Host: localhost:8080
Content-Type: application/json
Content-Length: 134

{
    "movieName":"The Shawshank Redemption ",
    "actorName":"Tim Robbins",
    "directorName":"Frank Darabont",
    "movieId":101
}
```

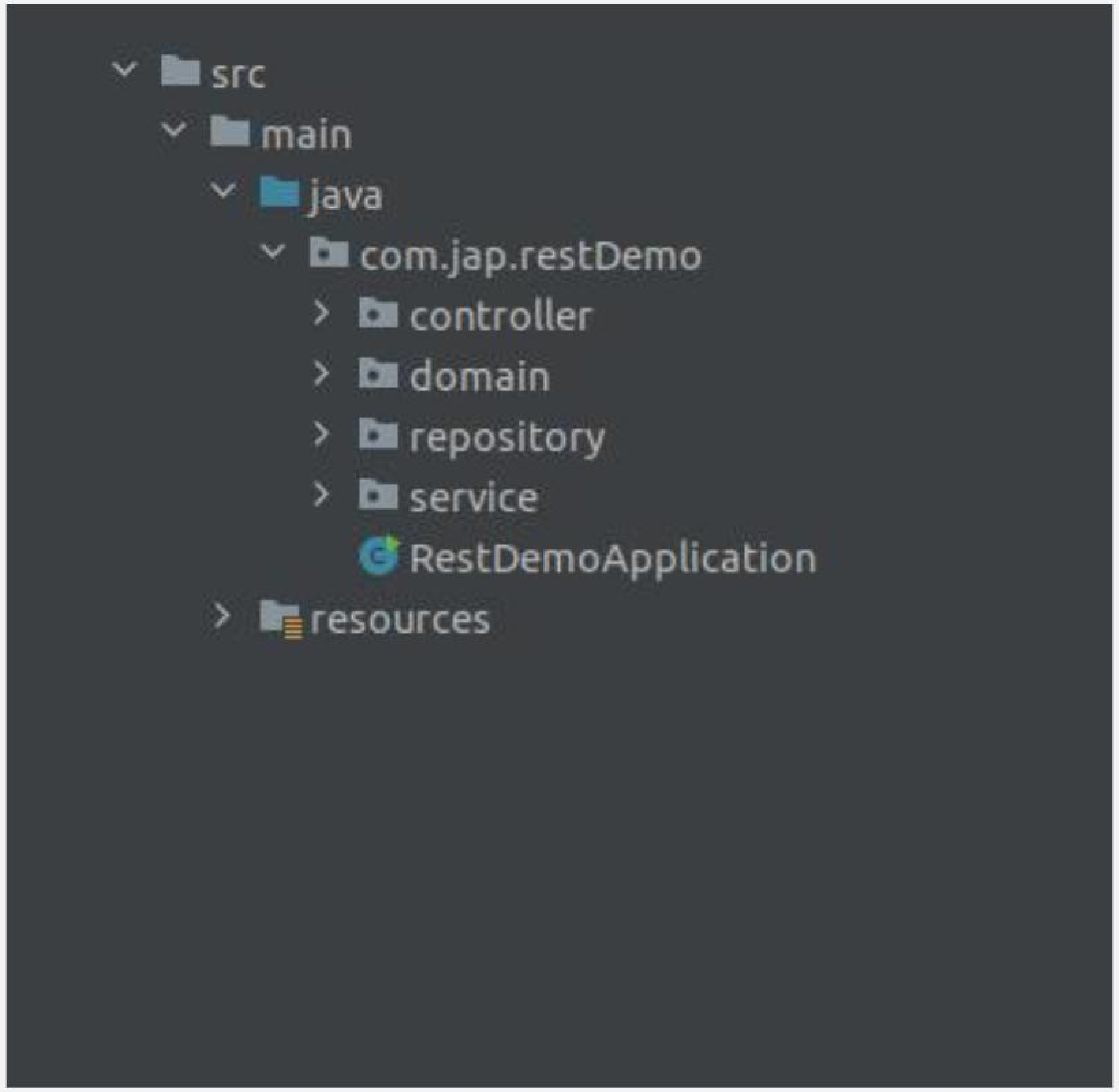
Request Header

RESTful Layers



A RESTful service is typically composed of three layers:

- **Controller layer** is responsible for handling requests and sending back a proper response.
- **Service layer** holds the business logic of the service – for example, tax calculation, bill generation, etc.
- **Repository layer**, also known as D-A-O (Data Access Object) layer, is responsible for interacting with the database and performing operations on the data.


```
src
  main
    java
      com.jap.restDemo
        controller
        domain
        repository
        service
        RestDemoApplication
    resources
```

RESTful Layers (cont'd)

- Here, `controller`, `domain`, `repository`, and `service` are user-defined packages created inside the root package, i.e., **`com.jap.restDemo`**.
- As per standard, controller, service, and repository should be the package names for the controller, service, and repository layers.
- The controller package will have the `controller` class to handle the request and send the response.
- The service package will have a `service` class containing the business logic.
- The repository package will have the `repository` class with all DAO information.
- The `Domain` classes are POJO (Plain Old Java Object) classes that contain the attributes of each object.

Postman - REST Client

- Postman is an HTTP client used for testing web services.
- It can be used to build, test, and share REST API calls.
- It's easy to use and has an intuitive interface that helps us access most of its features with just one click.
- It stores the API calls in the history, saving time and avoiding retaking API calls.
- It allows us to test our REST API without writing the code for testing or using any other additional setup.
- Install Postman.