CS 513 Software Systems(ESD)

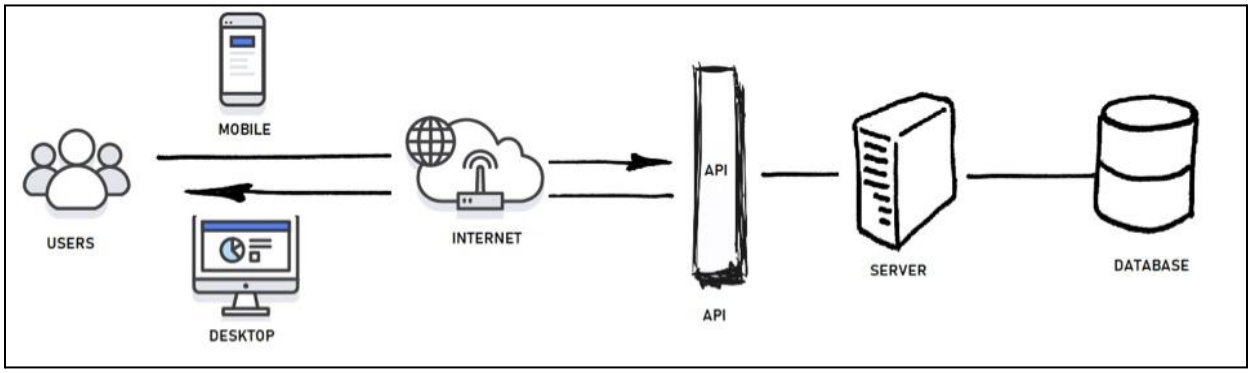
Lab 4: REST Services

Introduction:

All of you have learnt about the client-server architecture where the client sends the requests to the server and the server responds to the client accordingly;

But How???

We need some predefined functionalities(similar to functions) at the back-end that actively listens to the requests coming from the client side. Formally, these functionalities are called the API endpoints where API stands for Application Programming Interface. **API** is a software interface that allows two applications to interact with each other without any user intervention. API is a collection of software functions and procedures. In simple terms, API means a software code that can be accessed or executed. API is defined as a code that helps two different software's to communicate and exchange data with each other.



In this session, we are going to focus on one of the architectural styles(click here for more details) to design the API which is:

Web API...!!

This design uses HTTP protocol to access the API over the web. **RESTFUL services** developed are based on HTTP using technologies such as java and ASP.NET. While the API

is ideal for web browsers or web servers, it is not suitable for mobile applications. The Web API is used to provide services across multiple devices and on distributed systems.

RESTful API

A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. That data can be used to GET, PUT, POST and DELETE data types, which refers to the reading, updating, creating and deleting of operations concerning resources.

A RESTful API also referred to as a RESTful web service or REST API is based on representational state transfer (REST). REST technology is generally preferred over other similar technologies. This tends to be the case because REST uses less bandwidth, making it more suitable for efficient internet usage. RESTful APIs can also be built with programming languages such as JavaScript or Python.

A RESTful API uses existing HTTP methodologies defined by the RFC 2616 protocol, such as:

- GET to retrieve a resource;
- PUT to change the state of or update a resource, which can be an object, file or block;
- POST to create that resource; and
- DELETE to remove it.

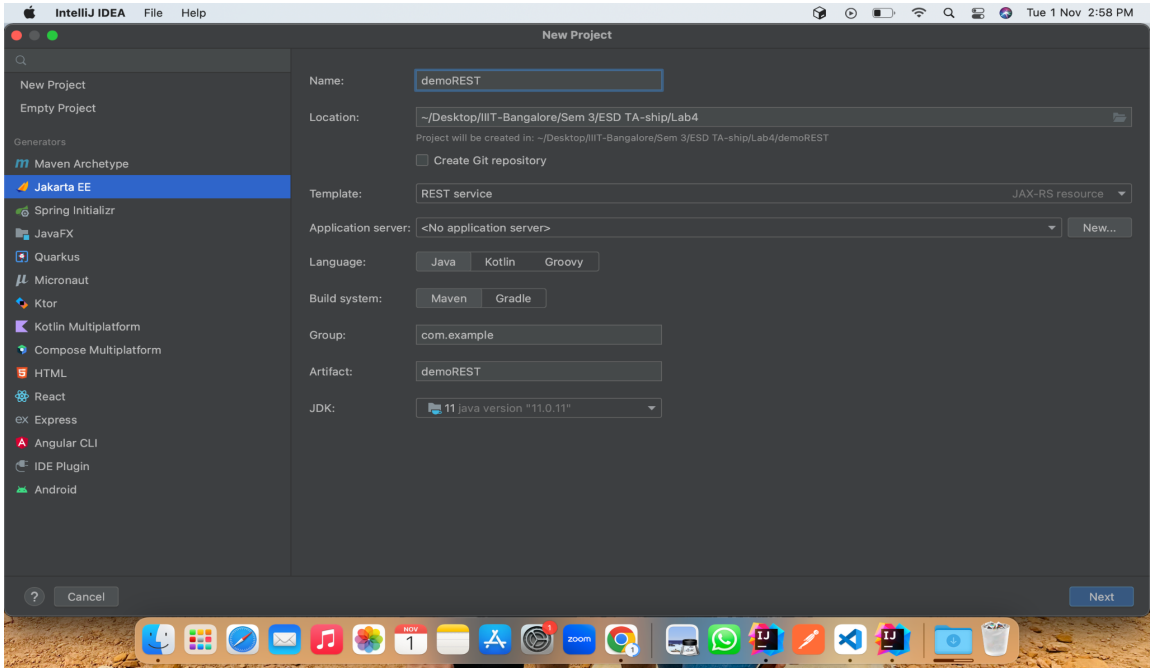
Prerequisite

- Environment setup lab handson
- Database setup(MySQL) - Lab 2
- Hibernate and JPA setup - Lab 3
- Postman
  Download link: https://www.postman.com/downloads/
- Glassfish server:
  Download link: https://projects.eclipse.org/projects/ee4j.glassfish/downloads

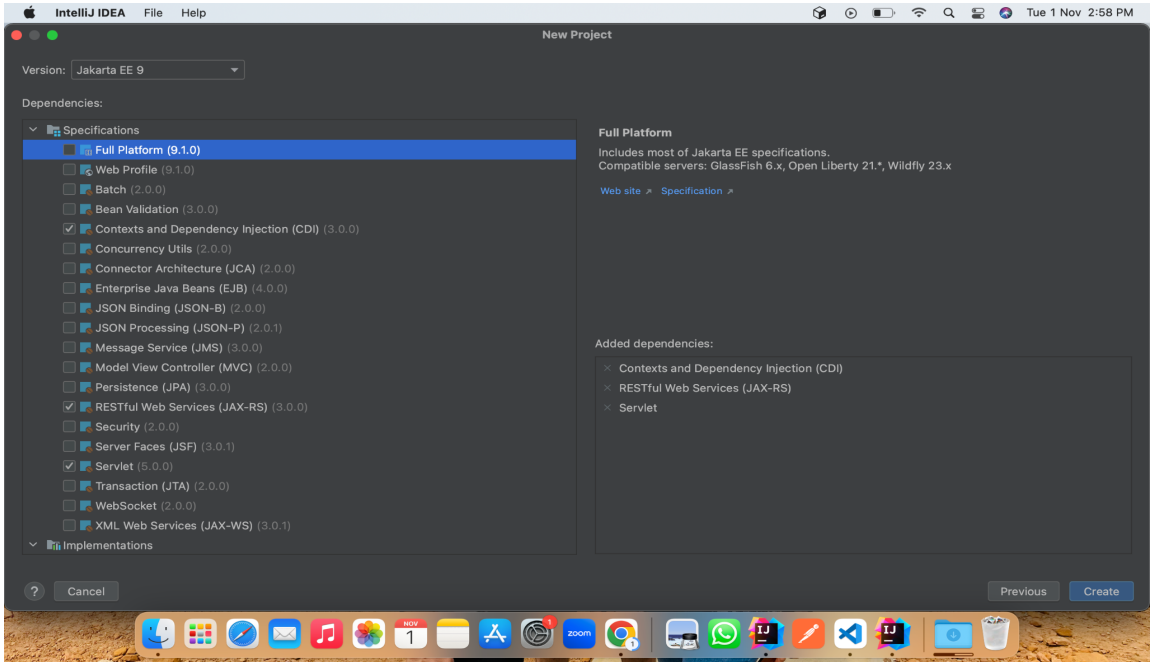
Lab Activity

Github link: https://github.com/Ashu-Soni/demoREST

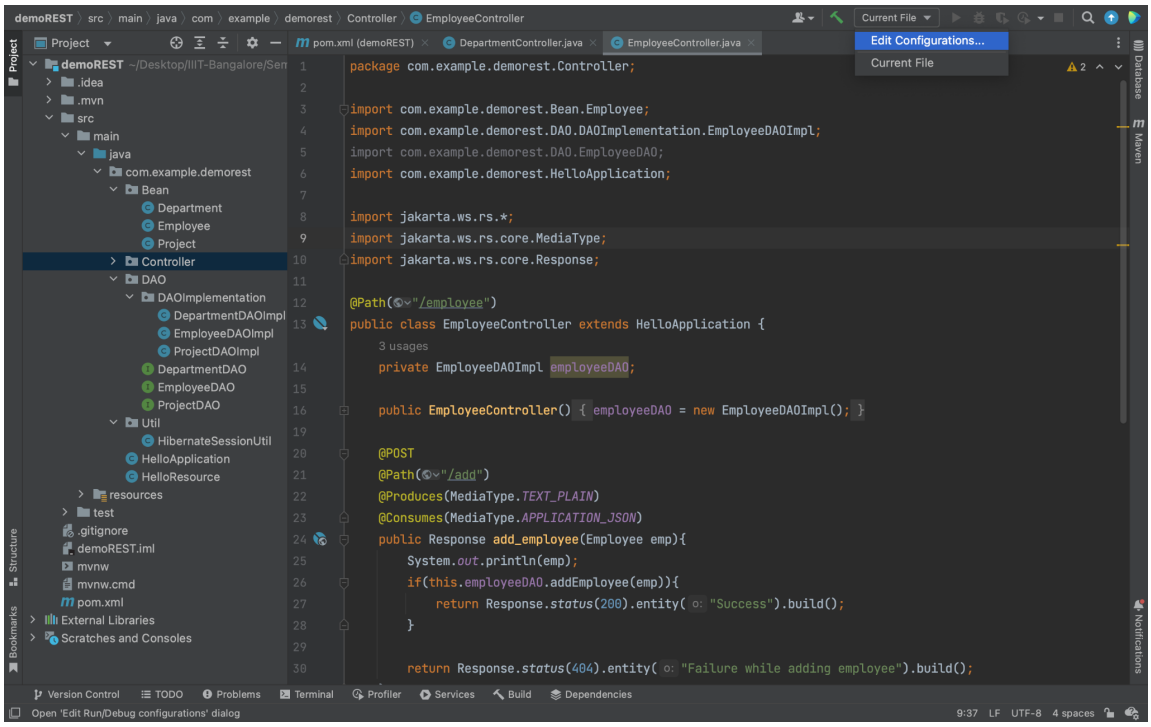
1. Initiate the project

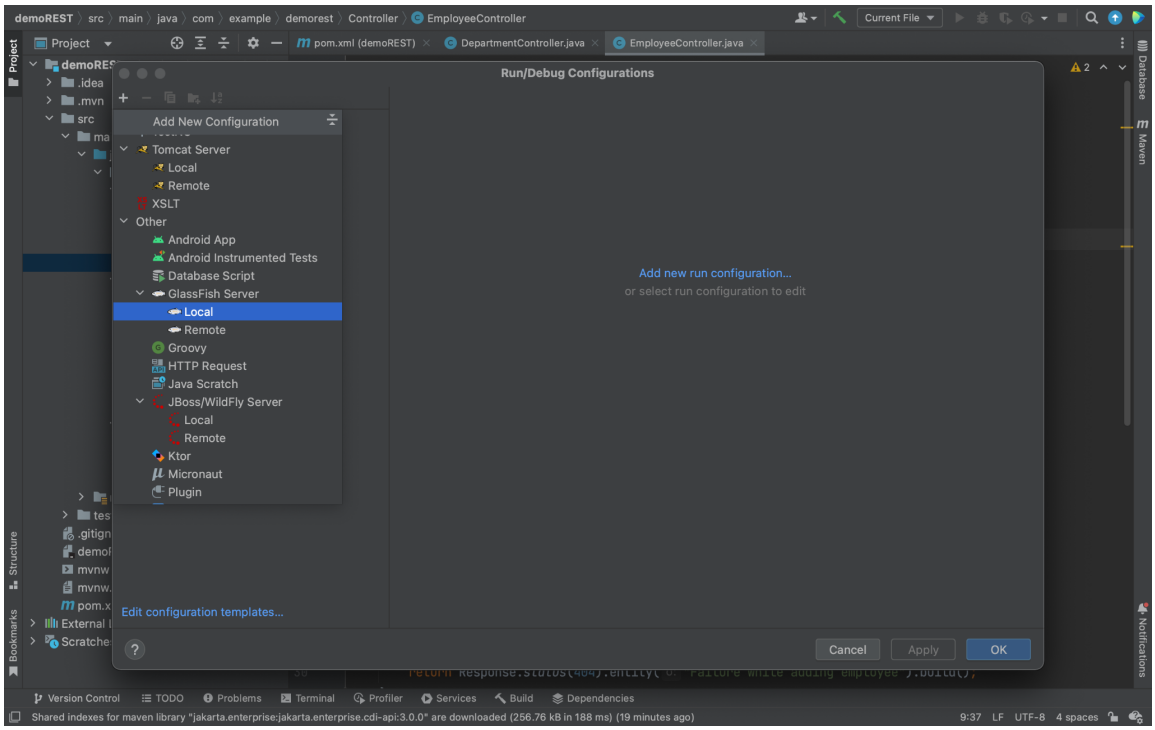


2. Add necessary dependencies

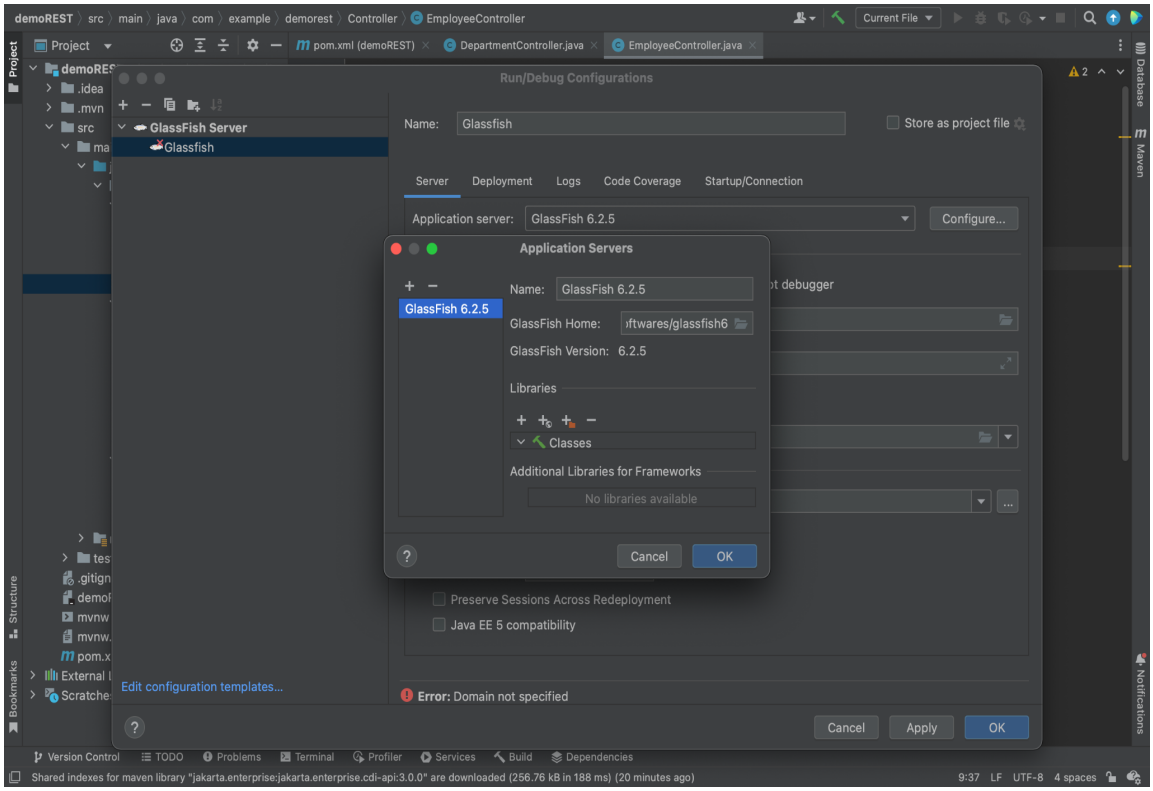


3. Add web server configuration

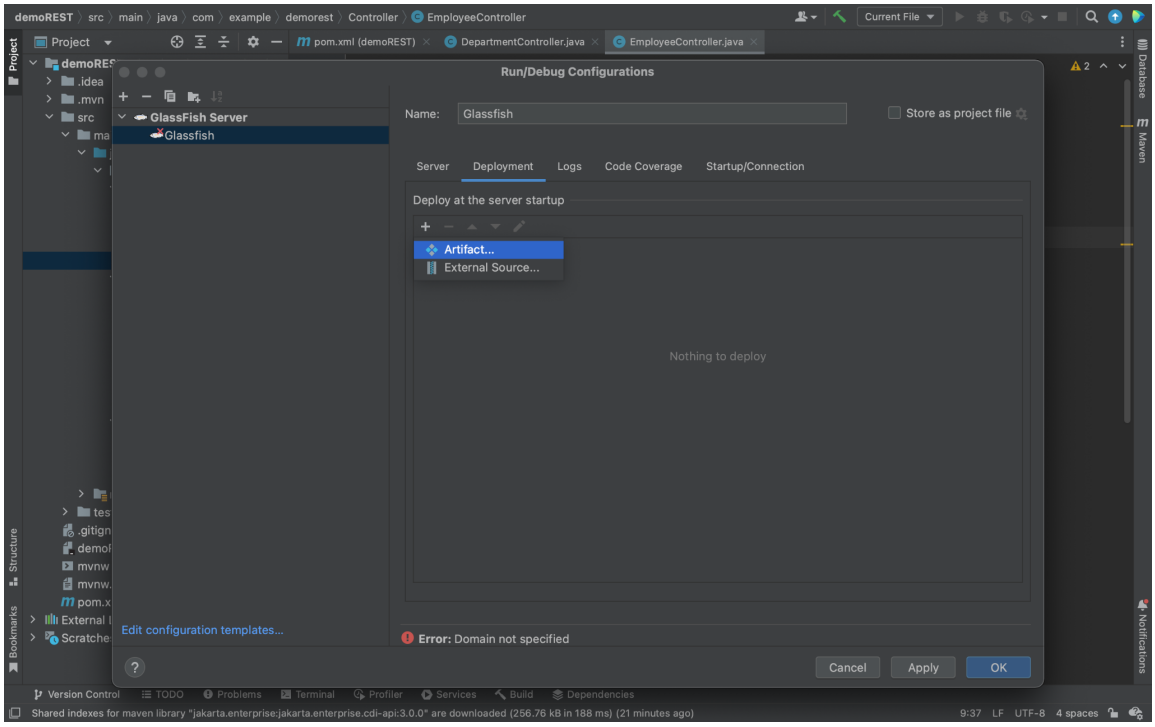


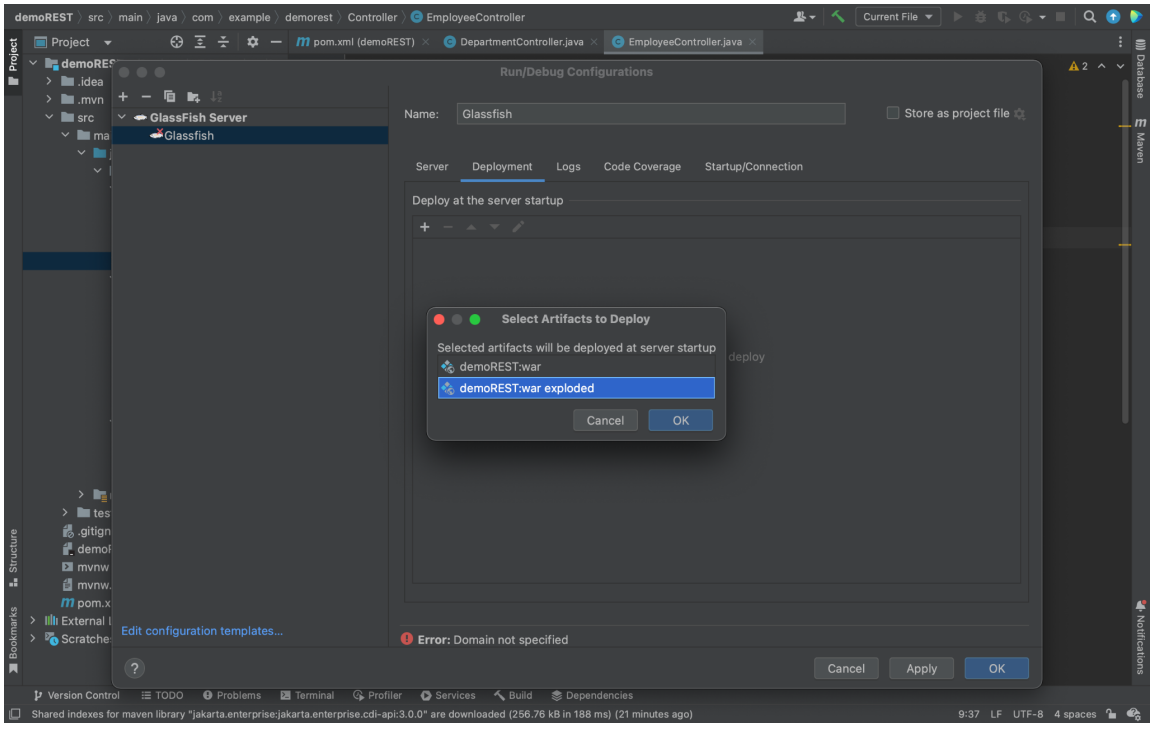


Provide the glassfish home directory path in the popup at “GlassFish Home” tab. Then *ok -> apply* , goto the server tab and fix the errors if any.



Add Deployment configuration: switch the tab to “deployment” and add the deployment artifact and make sure to select the artifact whose name ends with **war exploded**. Check out the following screenshots for your reference.


demoREST
src
main
java
com
example
demorest
Controller
EmployeeController
Current File
Project
pom.xml (demoREST)
DepartmentController.java
EmployeeController.java
Run/Debug Configurations
GlassFish Server
Glassfish
Name:
Glassfish
Store as project file
Server
Deployment
Logs
Code Coverage
Startup/Connection
Deploy at the server startup
Artifact...
External Source...
Nothing to deploy
Edit configuration templates...
Error: Domain not specified
Cancel
Apply
OK
Version Control
TODO
Problems
Terminal
Profiler
Services
Build
Dependencies
Shared indexes for maven library "jakarta.enterprise:jakarta.enterprise.cdi-api:3.0.0" are downloaded (256.76 kB in 188 ms) (21 minutes ago)
9:37
LF
UTF-8
4 spaces


demoREST
src
main
java
com
example
demorest
Controller
EmployeeController
Current File
Project
pom.xml (demoREST)
DepartmentController.java
EmployeeController.java
Run/Debug Configurations
GlassFish Server
Glassfish
Name:
Glassfish
Store as project file
Server
Deployment
Logs
Code Coverage
Startup/Connection
Deploy at the server startup
Select Artifacts to Deploy
Selected artifacts will be deployed at server startup
demoREST:war
demoREST:war exploded
Cancel
OK
deploy
Edit configuration templates...
Error: Domain not specified
Cancel
Apply
OK
Version Control
TODO
Problems
Terminal
Profiler
Services
Build
Dependencies
Shared indexes for maven library "jakarta.enterprise:jakarta.enterprise.cdi-api:3.0.0" are downloaded (256.76 kB in 188 ms) (21 minutes ago)
9:37
LF
UTF-8
4 spaces

4. Check the working of application

Click the run button appeared on the top to start the application:

Note: In case of failure like "port already in use", you have to change the configuration server port of the glassfish server. Search for "8080" in the file(path/to/glassfish6/glassfish/domains/domain1/config/domain.xml) and replace it with another port number like "9090". And run the application again.

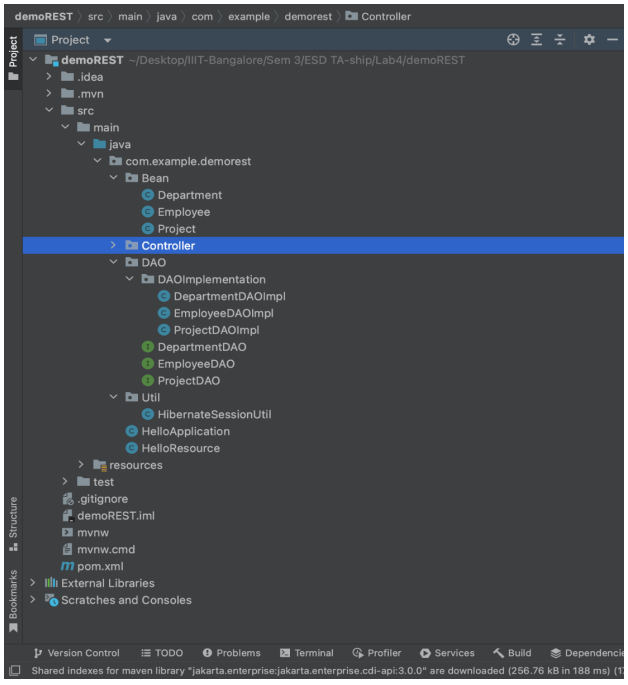
After successful run, hit the following URL on the browser:

http://localhost:<port_number>/api/hello-world

This must show the hello world message on the browser page and it means that the server is up and running.

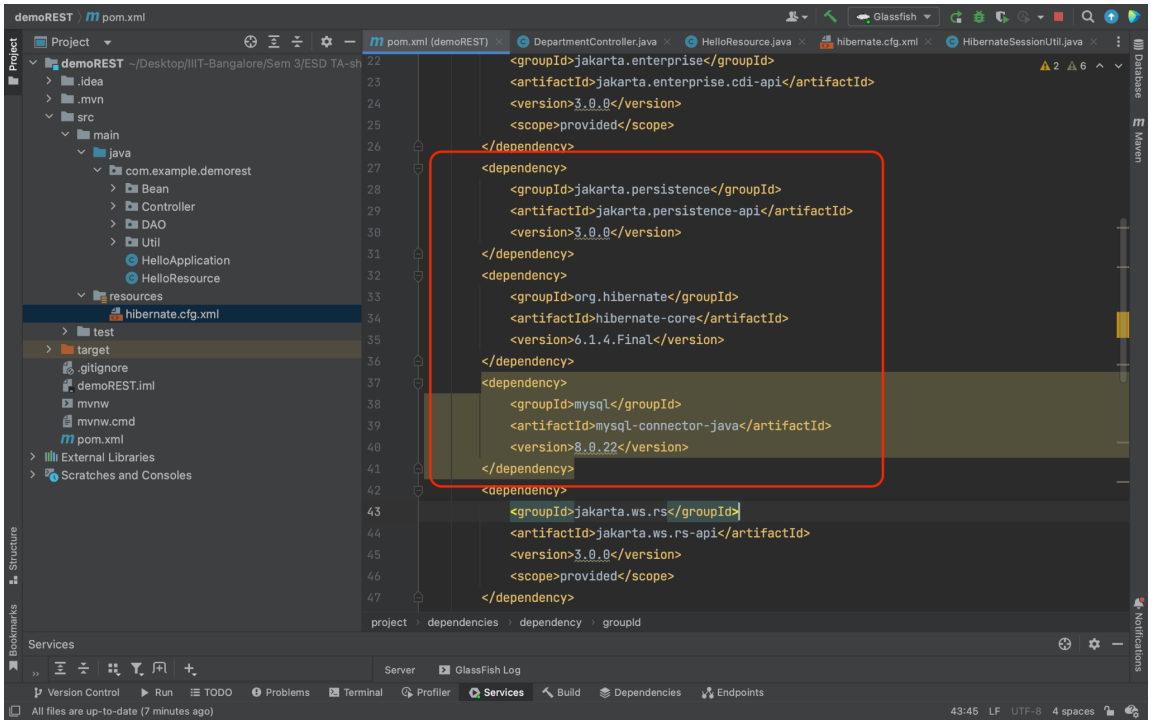
5. Paste the existing code(i.e. Till hibernate lab)

Paste the existing code that was covered in the previous exercise in appropriate directory structure. Please check the following image for the structure.



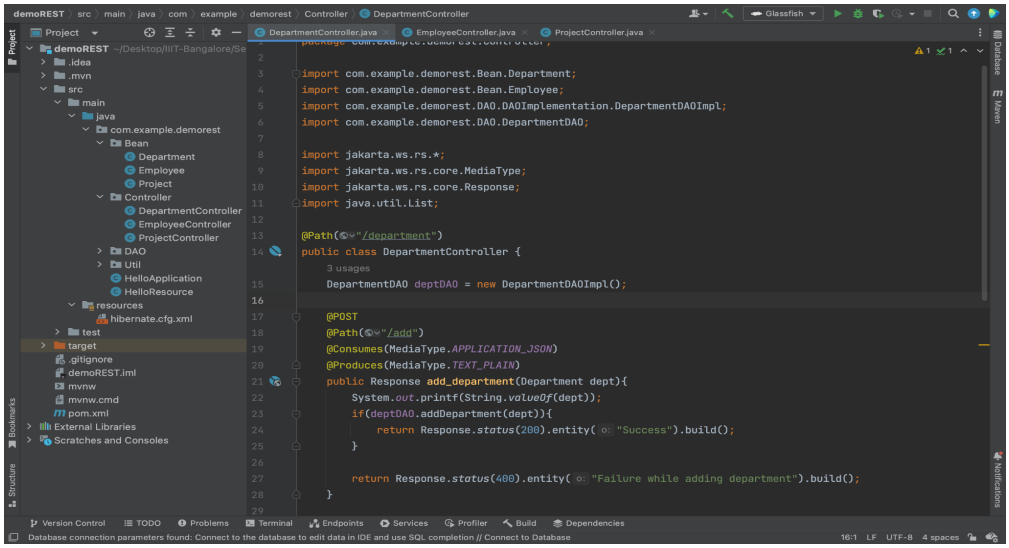
Also add the “hibernate.cfg.xml” file under the resource directory.

6. Add additional dependencies in the project



7. Start adding controller in place of driver code

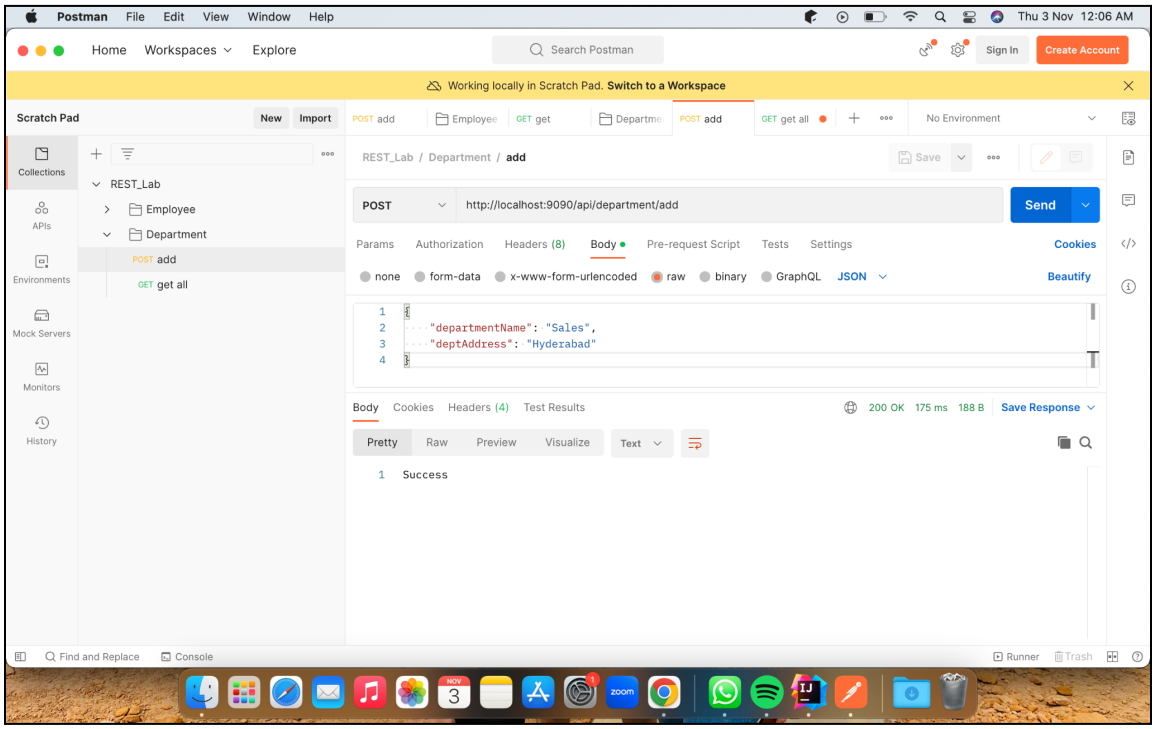
Add the package under the location “src/main/java/com/example/<project_name” with the name “Controller”. Then create the java class named “DepartmentController” under the “Controller” package and add the following code. Check the whole code from here.



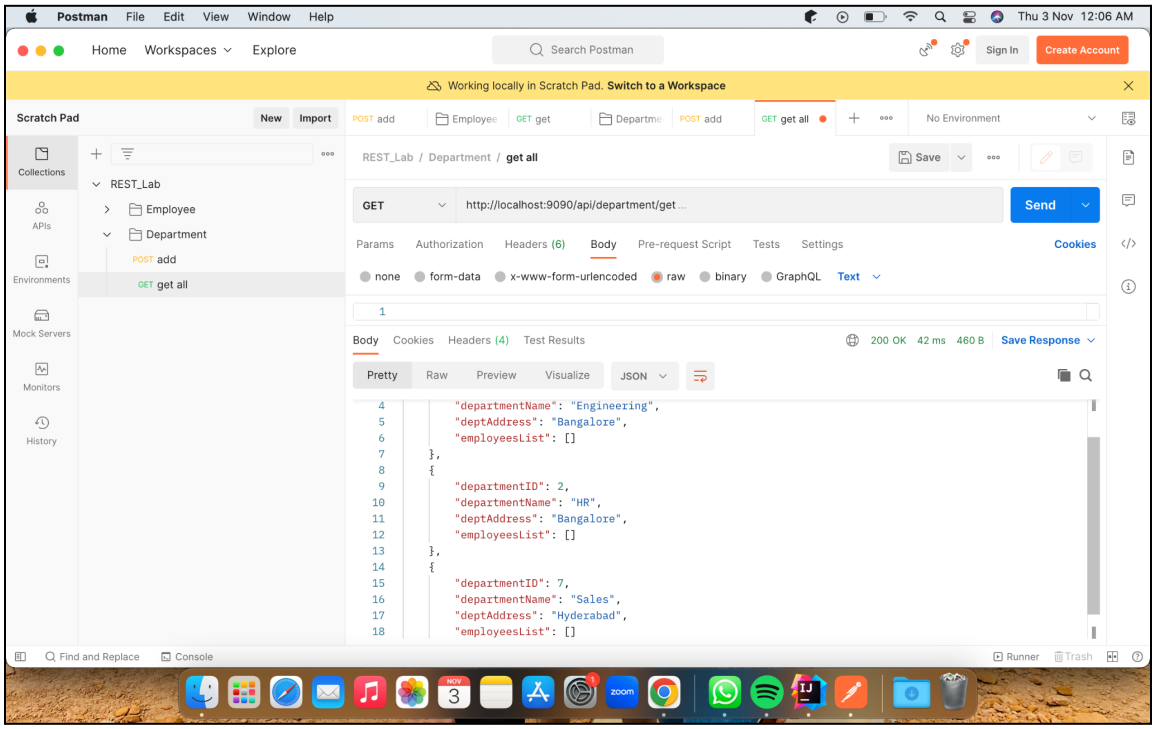
Likewise, you can generate multiple API endpoints in Department class and can create multiple classes as well(i.e. Controller for Project and Employee). Feel free to check the whole code from this repository.

8. Check the endpoints using postman

You can check the endpoints using the postman tool. Here are some screenshots from the postman UI:



POST call to add new department



GET call to extract all departments

You can download the postman collection from here. Download the file and import it into your postman, after which you can see all the requests urls/data.

Links

1. https://www.techtarget.com/searchapparchitecture/definition/RESTful-API
2. https://hackernoon.com/the-basics-of-designing-an-api-architecture
3. https://blog.bitsrc.io/not-all-microservices-need-to-be-rest-3-alternatives-to-the-classic-41cedbf1a907
4. https://www.jetbrains.com/help/idea/creating-and-running-your-first-restful-web-service.html