**Scenario 1:**

Make a DB call to get the required employee details using jpa(Spring boot)

**Steps to Implemention**

**Step 1 :**

Initially created a spring boot application by using the eclipse

**Step 2 :**

And then created on created a interface called EmpRepo.java that was extends the JpaRepository

**Step 3:**

After that created one controller named as EmpController.java that should autowired  
EmpRepo.java

**Step 4 :**

Created one method to get required emp id by using findById method which is in default   
method of JpaRepository

**Step 5 :**

H2 internal database created by using application.property file by enabling h2

Database ,

**Step 6 :**

After that created a model called Employee classs which is annotated with @Entity

Then added fields emp\_id,emp\_name,emp\_mailid , once spring boot application

Started , it will create a table in H2 database with mentioned fields

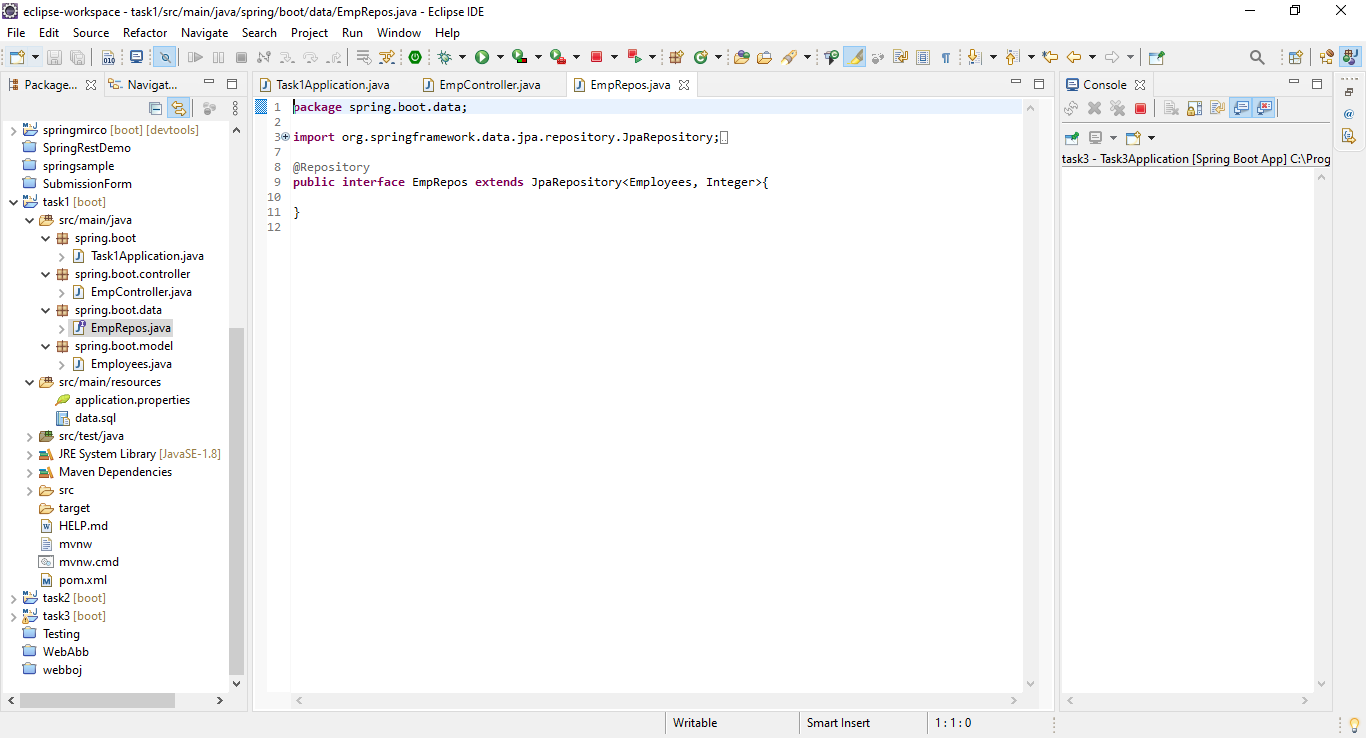
**Step 7 :**

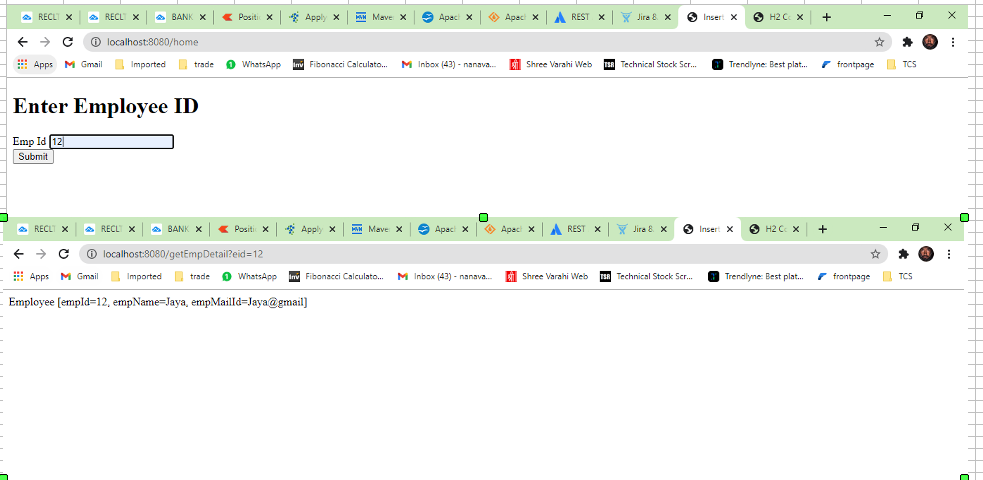
While calling the getEmpDetail method from controller ,spring boot call the corresponding

Method from JpaRepository , and it will hit the H2 table and return response to controller

**Step 8 :**

After getting successful response , that is showed in corresponding view jsp page





**Scenario 2 :**

Rest call to jira api to get the issues created in a particular date for a project

**Steps to Implemention**

**Steps 1:**

Initially created a spring boot application by using the eclipse ,and then

Created the controller class file for set the path api/RestJiraController

**Steps 2:**

Inside that controller, getJiraIssue method is created for executing the above

Scenario

**Steps 3:**

Then a Service class is autowired and invoked from controller class to get

The business logic.

**Step 4 :**

Here JiraRestClient which is used to invoke the a rest service , and then

getSearchClient method is used to get corresponding issues for particular

project in particular date

**Step 5:**

And then basic authentication that is username and password of jira

Has been given in AuthenticationHandler which is used for authentication for jira

**Step 6 :**

searchJql method is used for searching the issue for particular project for

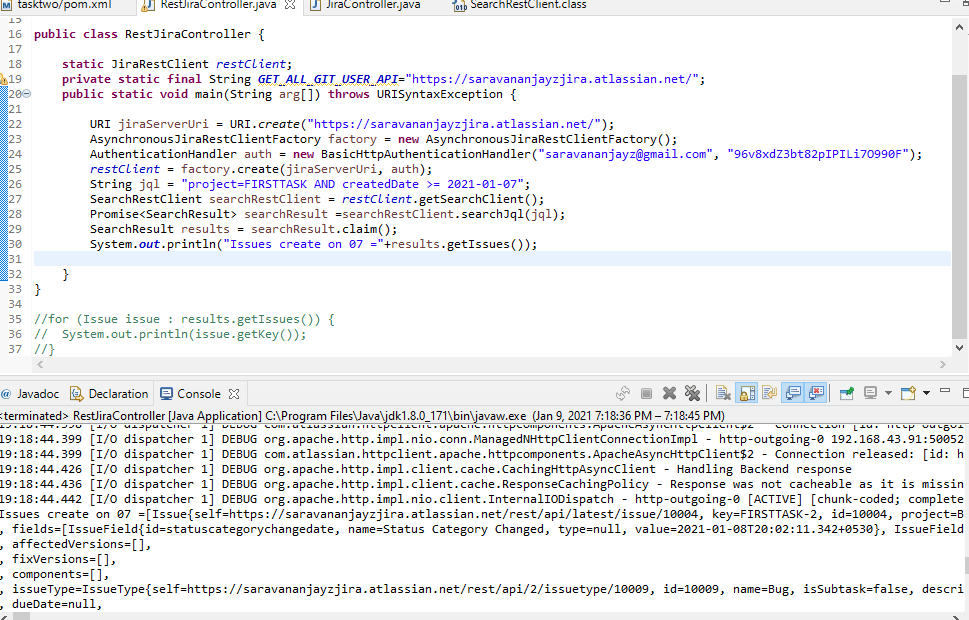
mentioned date like below

JiraRestApi / atlassian.net /rest/api/latest/search?jql=project={projectName}

And createdDate ={date}

**Step 7:**

After Succuessfull completion from restapi , response object is return



**Scenario 3:**

Exception handling using Aop using spring boot application

**Steps to Implemention**

**Step 1 :**

Initially created a spring boot application by using the eclipse

**Step 2 :**

Inside the src folder created one controller package after that inside that package

EmpController.java class file is created which is used for handle the exception

**Step 3 :**

After that created one method which method is used for getting employee details

From emp id , if emp id is not present in corresponding table it should throw

The Resource not found exeception

**Step 4 :**

Parallelly created the Exception package , inside that created exception response

Model class file which have errorMessage,errorCode,timestamp ,and then create getter

Setter and constructor for this fields

**Step 5 :**

After that created one class called Global exception handler which class is annotated

With @ControllerAdvice annotation , which allows to handle the exceptions across

While application in one global handling component

**Step 6 :**

And then create Resource not exception class file in same package it will

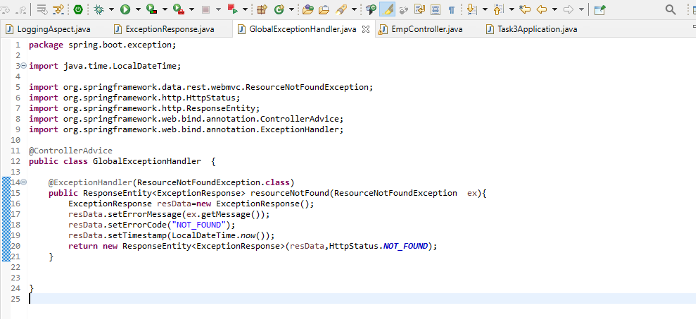
Extends the RunTimeException class

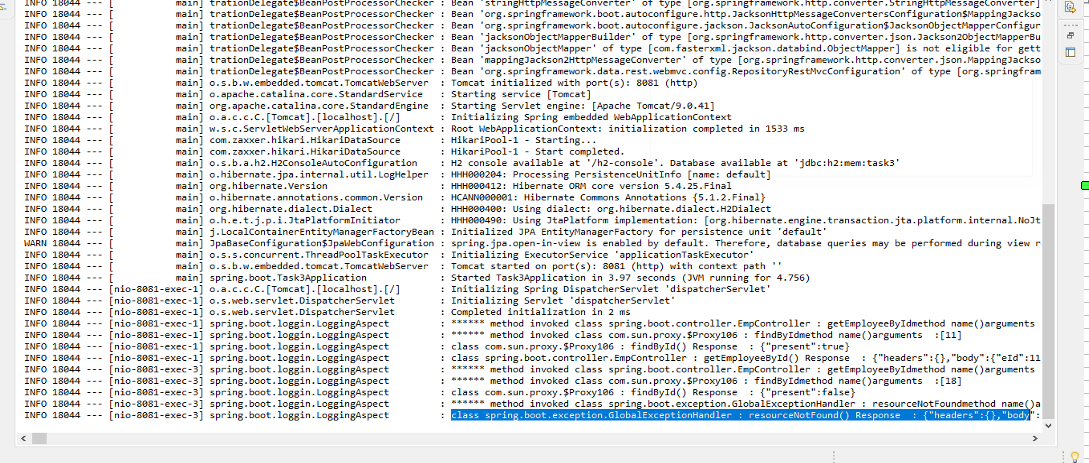
**Step 7 :**

While corresponding emp id not present in table , it will throw the resource not found

Exception through the global exception handler to mapping the error message and error

Code and timestamp and it will throws the exception successfully.





**Scenario 4:**

Rest call to Git API to get the lock status of user and recent commit list of repo

**Approach**

* Login into GitHub site
* Create a project and repo and branch from github site
* Create some commits by adding files into repo
* Click that drop down to see the commit messages

**Steps to Implemention**

**Step 1:**

Initially created a spring boot application by using the eclipse and add necessary dependency

**Step 2:**

Created a controller class is set for the path api/RestClientController then created

callGetAllCommitsApi method in this controller for above scenario

**Step 3:**

Here used the RestTemplete which is used to invoke a rest service and then

restTemplete ‘exchange’ method is used to call corresponding api

**Step 4:**

And then parameters like api url and http method and then string class added

to restTemplete exchange method

**Step 5 :**

Rest template api url is to get latest commits for particular user below url

Rest api / repos/<users> /<repository>/commits is used to find the latest commits

in a corresponding branch

http GET call method is executed to get response passing the query parameters and

header data

**Step 6 :**

After getting successful response object ‘message’ property is used to get

all commit messages from particular repo

**Step 7:**

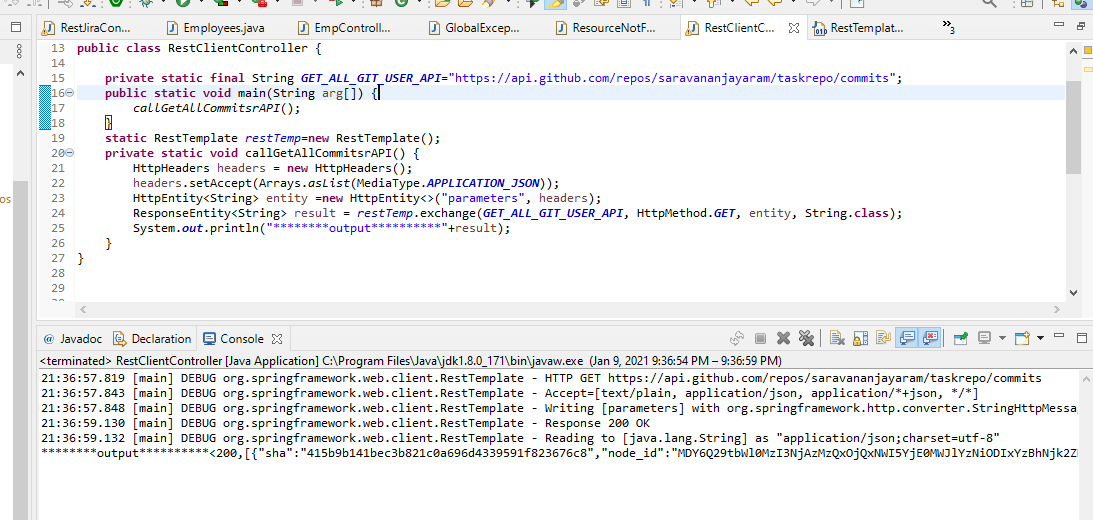
And then Second business logic is to get the lock status for particular user from

Organizations

By using Restapi /orgs/{org}/blocks/{username} is used to find blocked status of the particular user from Organizations

After Successful completion from response object ‘status’ property is used get status of user

Cant get lock status for branch due to need paid version of github



**Scenario 5:**

Spring mvc end to end flow samples like login page(jsp -> Controller -> jsp)

**Steps to Implemention**

**Step 1:**

Initially created spring mvc application by using the new -> mavan project

**Step 2 :**

After that index.jsp file and view.jsp file is created , inside index.jsp file form created

With Name, password and submit button fields , view.jsp file is used to display the

Response data which is get from controller

**Step 3:**

And then login controller.class file created inside controller package which have

the login method that return the ModelAndView Object

**Step 4:**

And Then FrontController.class file created inside same package which should

extends the AbstractAnnotationConfigDispatcherServletInitializer

this frontcontroller worked as dispatchservlet it will get the request from browser

and then send that request to corresponding controller

which have default methods to config the corresponding servlet files

**Step 5 :**

After that MvcConfig file created which should extends by WebMvcConfigurerAdapter

Which should have @EnableWebMvc annotation which is used to enable the

Spring MVC

**Step 6 :**

After entering the user name and password in index page by clicking the submit button

It will send corresponding request to dispatchservlet , and then it will send to corresponding

Controller

Step 7 :

In Controller, login method will called which have modelandview return type

Done logic in that method and return response in the format of modelandview to view jsp

