TASK 2

Contents

[Goal 3](#_Toc204643660)

[Approach 6](#_Toc204643661)

[Implementation 7](#_Toc204643662)

[Logic 7](#_Toc204643663)

[Testing 8](#_Toc204643664)

[H2 Database after startup 8](#_Toc204643665)

[GET /role/{id} - get role by Id 8](#_Toc204643666)

[GET /allroles - get complete list of roles in nested structure. 9](#_Toc204643667)

[Negative Testing 13](#_Toc204643668)

[Pass incorrect roleId: GET /role/{id} - get role by Id 13](#_Toc204643669)

# Goal

Create Db Connection:

Host a database in local or aws (postgres or mysql or inmemory db(H2)).

Configure your microservice to connect to db.

Create Hibernate Entity Class based on the below table requirement to auto create tables in hosted db.

Populate the db from a post request with the below data or directly insert into db.

Expose endpoints to fetch requests based on Id and complete list of objects.

While getting the complete table, Modify the result set(arrays or list) to form nested object structure.

Below is the nested output in json. Associate color to each object according to table.

Create a table with fields ID, Name, Color, ParentId and populate with below data.

Don't use ORM to map or form parent child relationships. Write an algorithm to form the relationship after fetching the list from db, in an efficient way.

|  |  |  |  |
| --- | --- | --- | --- |
| id | parentid | name | color |
| 1 | 0 | Warrior | red |
| 2 | 0 | Wizard | green |
| 3 | 0 | Priest | white |
| 4 | 0 | Rogue | yellow |
| 5 | 1 | Fighter | blue |
| 6 | 1 | Paladin | lighblue |
| 7 | 1 | Ranger | lighgreen |
| 8 | 2 | Mage | grey |
| 9 | 2 | Specialist wizard | lightgrey |
| 10 | 3 | Cleric | red |
| 11 | 3 | Druid | green |
| 12 | 3 | Priest of specific mythos | white |
| 13 | 4 | Thief | yellow |
| 14 | 4 | Bard | blue |
| 15 | 13 | Assassin | lighblue |

Nested Json response. Ignore Dangling Branches Scenario.

[

{

"Name": "Wizard",

"Sub Classes": [

{

"Name": "Mage"

},

{

"Name": "Specialist wizard"

}

]

},

{

"Name": "Priest",

"Sub Classes": [

{

"Name": "Cleric"

},

{

"Name": "Druid"

},

{

"Name": "Priest of specific mythos"

}

]

},

{

"Name": "Warrior",

"Sub Classes": [

{

"Name": "Fighter"

},

{

"Name": "Paladin"

},

{

"Name": "Ranger"

}

]

},

{

"Name": "Rogue",

"Sub Classes": [

{

"Name": "Thief",

"Sub Classes": [

{

"Name": "Assassin"

}

]

},

{

"Name": "Bard"

}

]

}

]

Create a method level Annotation @LogMethodParam which logs parameters passed to method.

# Approach

**Db Connection:** in-memory h2 db

**Data inserted through**: data.sql

Entire table converted into insert statements in data.sql

**Expose endpoints**

* GET /role/{id} - get role by Id
* GET /allroles - get complete list of roles in nested structure.

**Create a method level Annotation @LogMethodParam which logs parameters passed to method.**

Log before each method call 🡺 Use AOP Interception

* Create annotation LogMethodParam
* Wherever @LogMethodParam annotation is used, before method call, logs will be printed.

**Note**

* Associate color to each object according to table. 🡺 Need to keep color also in return object.
* Don't use ORM to map or form parent child relationships. 🡺 No ORM mapping to be done. Handle through java logic.

# Implementation

* **Role**: Basic Java Entity Equivalent to roles table in DB
* **RoleDto**: transfer only relevant information via API
* **RoleMapper**: To convert Role Entity to RoleDto
* **RoleService**: Logic for getRoleById(Long id) & getRoles()
* Others are helper classes.

## Logic

**getRoleById**(Long id): get role by passed id. Find through repository.

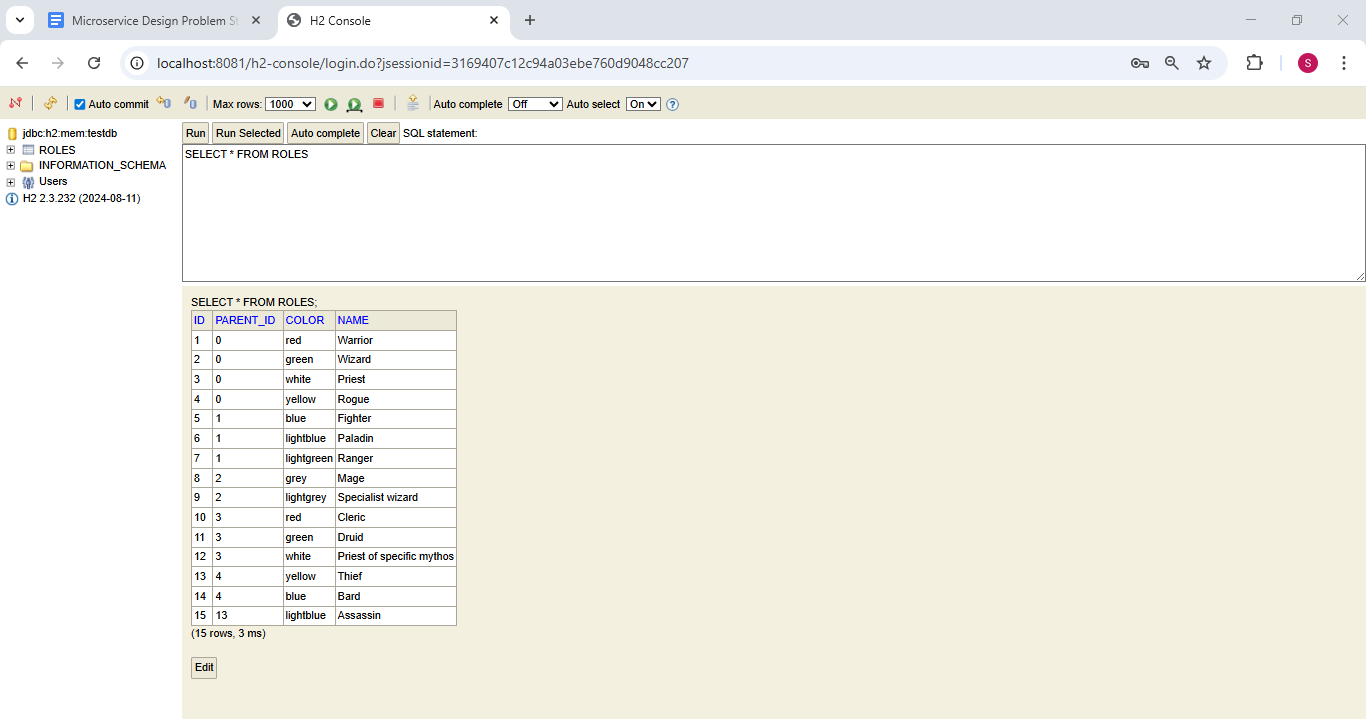
**getRoles**(): get all roles and form nested structure.

* Fetch all roles
* Map roles by ID
* Build the role hierarchy based on parentId and Id. Add into subClasses of parent.

<https://github.com/sudhanshu-y/microservice/tree/main/Task2/task2>

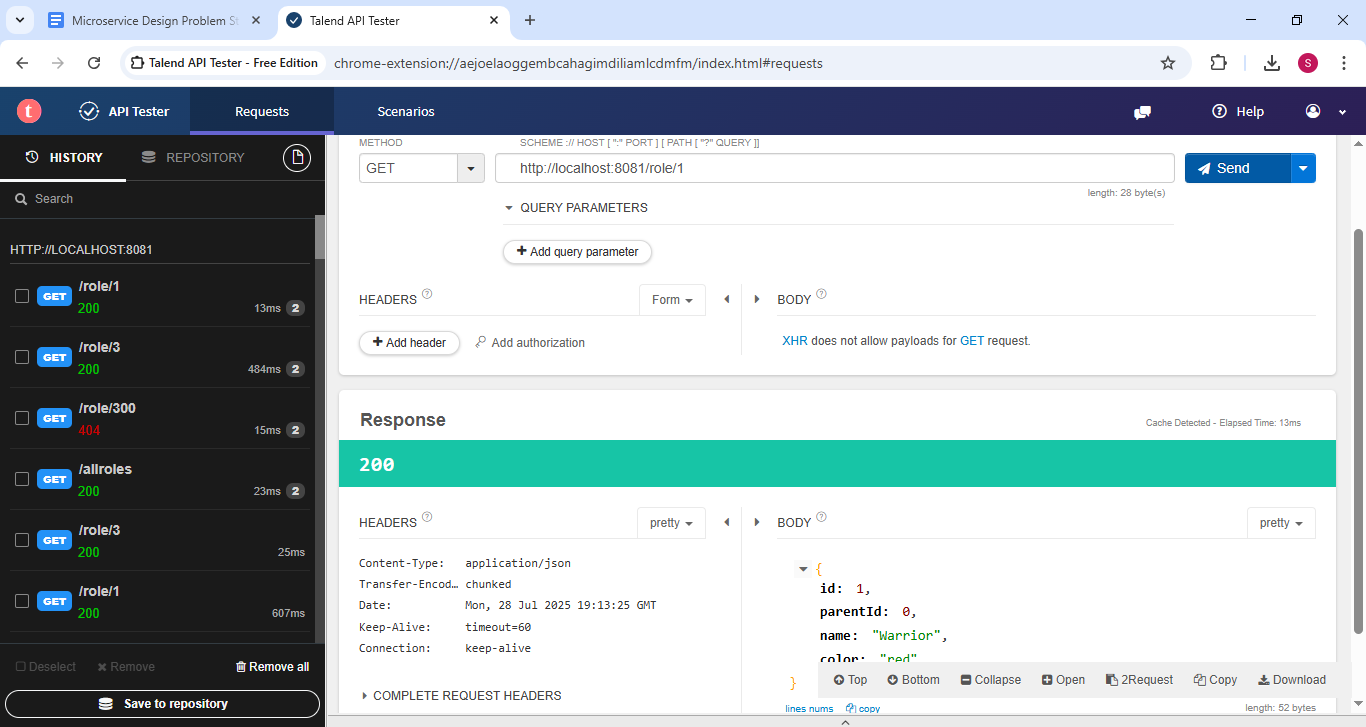
# Testing

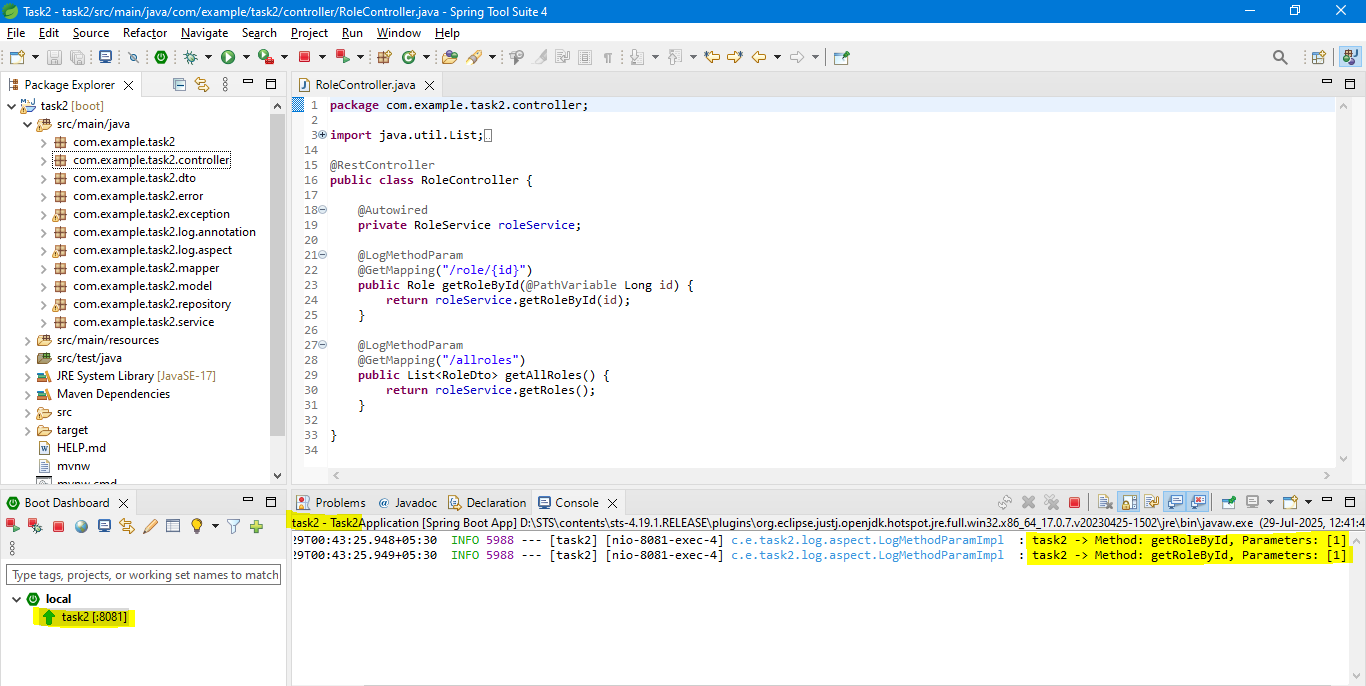
## H2 Database after startup



## GET /role/{id} - get role by Id

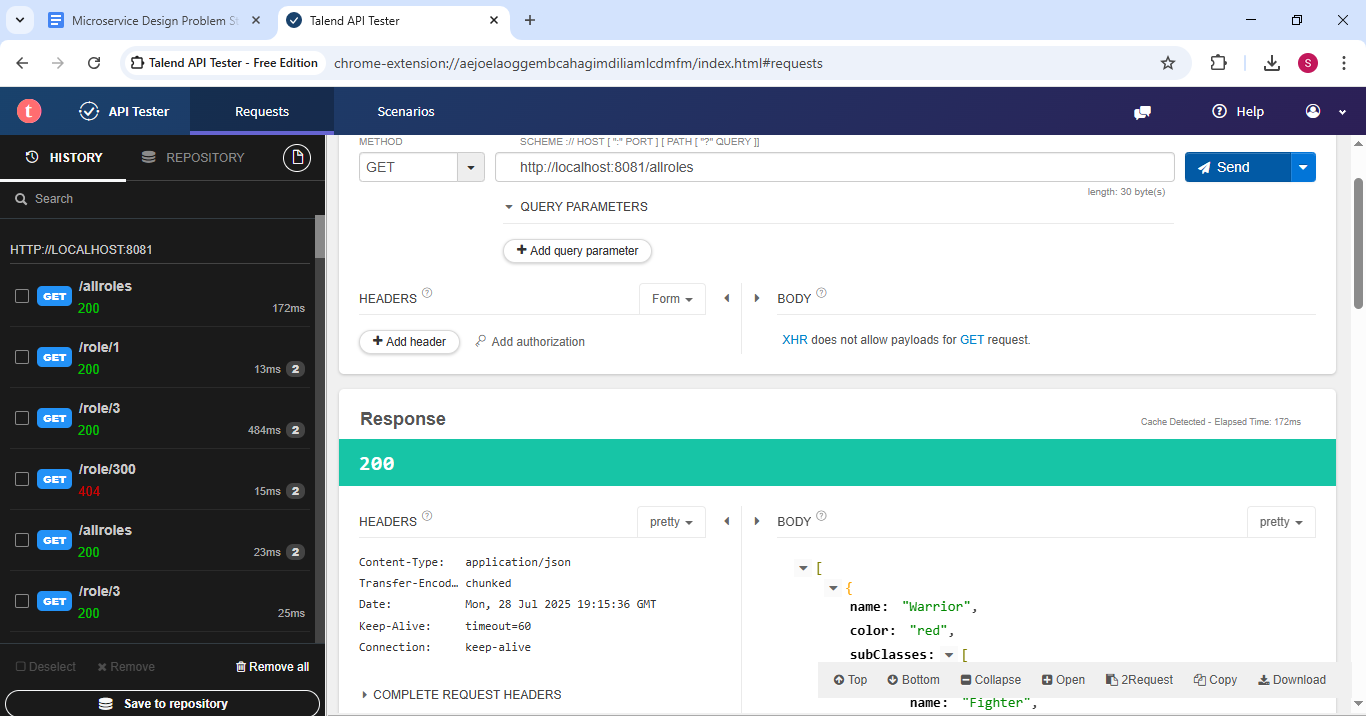
<http://localhost:8081/role/1>

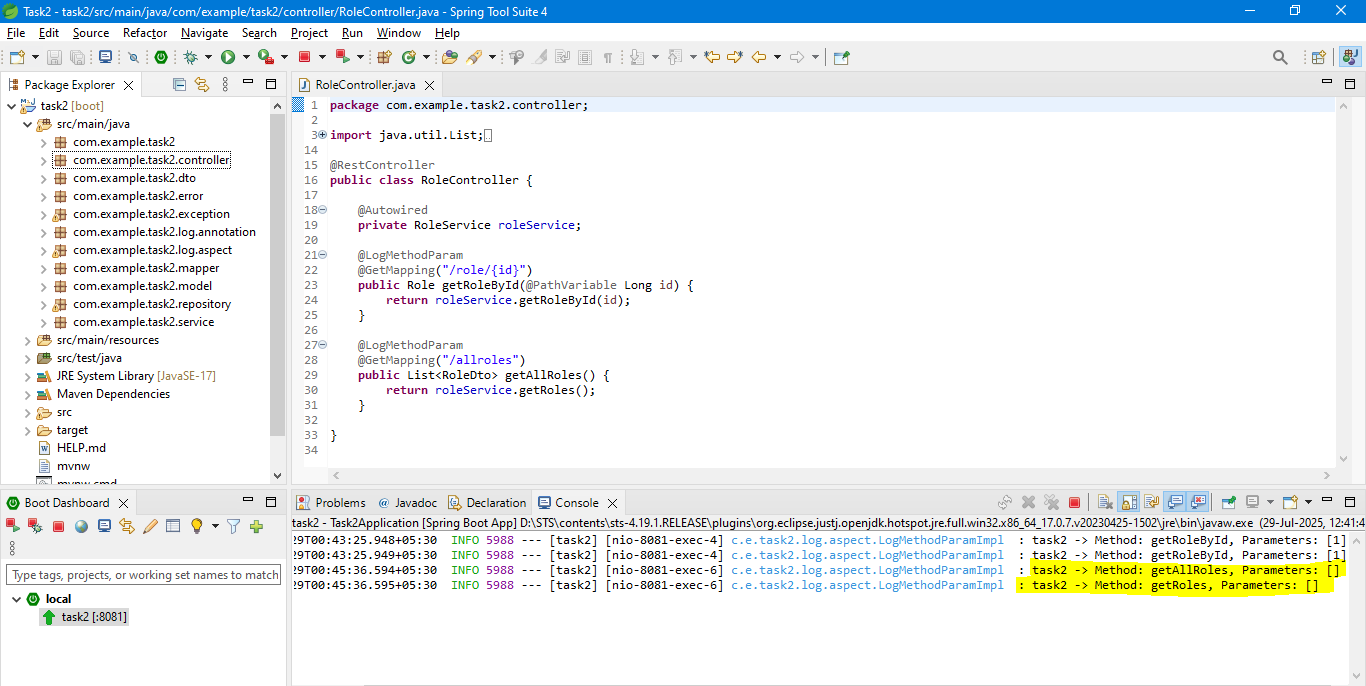




|  |
| --- |
| {  "id": 1,  "parentId": 0,  "name": "Warrior",  "color": "red"  } |

## GET /allroles - get complete list of roles in nested structure.

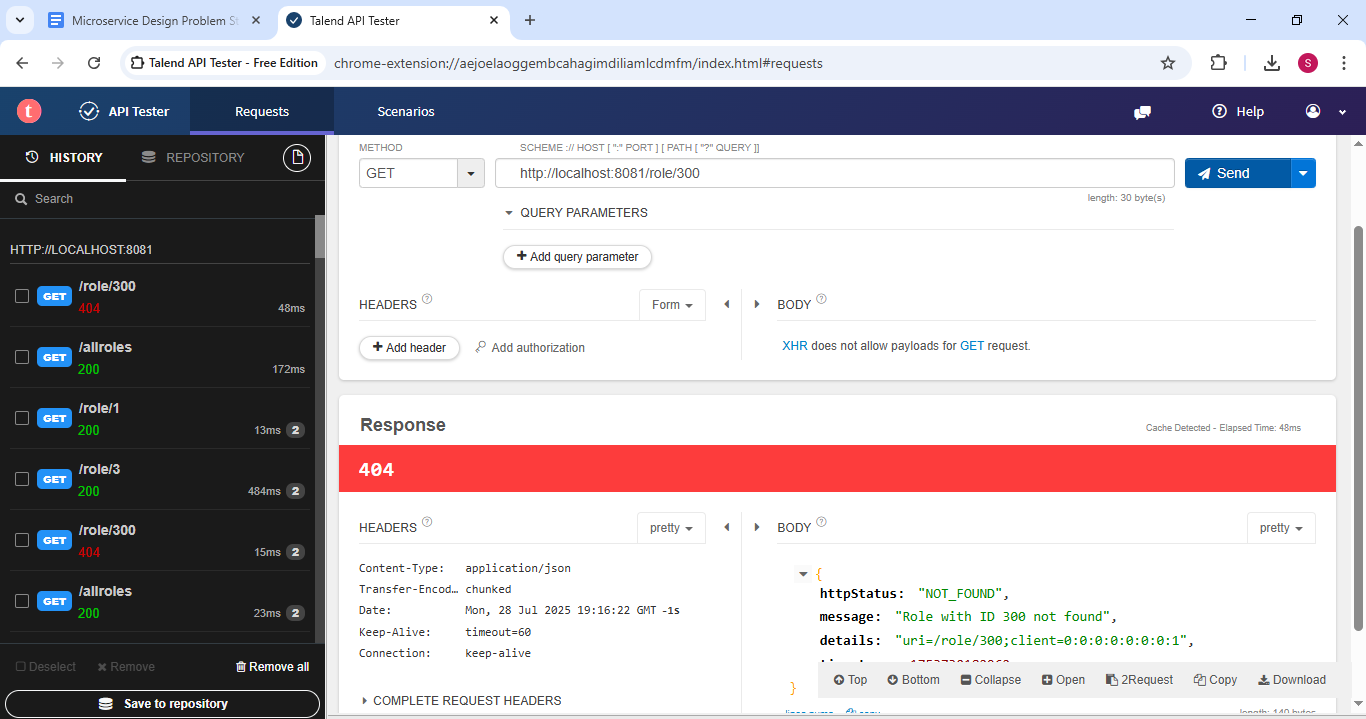


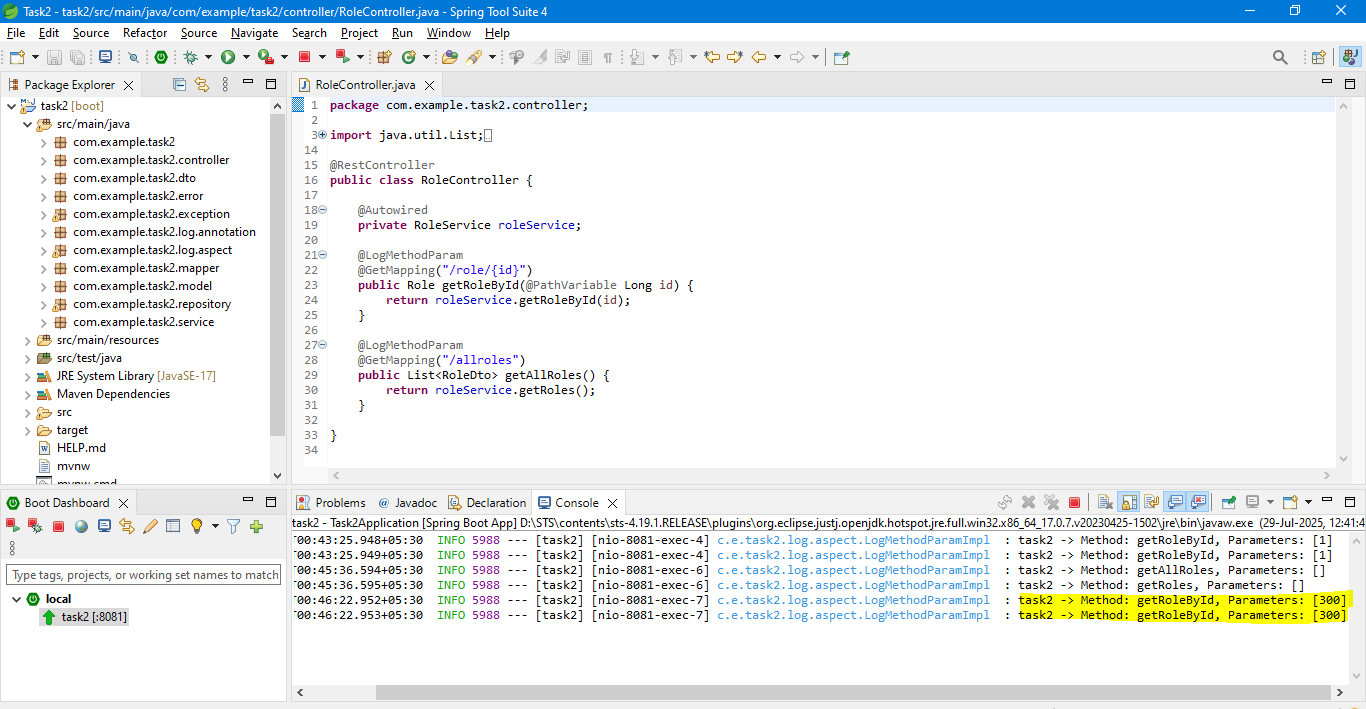


|  |
| --- |
| [  {  "name": "Warrior",  "color": "red",  "subClasses": [  {  "name": "Fighter",  "color": "blue",  "subClasses": [  ]  },  {  "name": "Paladin",  "color": "lightblue",  "subClasses": [  ]  },  {  "name": "Ranger",  "color": "lightgreen",  "subClasses": [  ]  }  ]  },  {  "name": "Wizard",  "color": "green",  "subClasses": [  {  "name": "Mage",  "color": "grey",  "subClasses": [  ]  },  {  "name": "Specialist wizard",  "color": "lightgrey",  "subClasses": [  ]  }  ]  },  {  "name": "Priest",  "color": "white",  "subClasses": [  {  "name": "Cleric",  "color": "red",  "subClasses": [  ]  },  {  "name": "Druid",  "color": "green",  "subClasses": [  ]  },  {  "name": "Priest of specific mythos",  "color": "white",  "subClasses": [  ]  }  ]  },  {  "name": "Rogue",  "color": "yellow",  "subClasses": [  {  "name": "Thief",  "color": "yellow",  "subClasses": [  {  "name": "Assassin",  "color": "lightblue",  "subClasses": [  ]  }  ]  },  {  "name": "Bard",  "color": "blue",  "subClasses": [  ]  }  ]  }  ] |

# Negative Testing

## Pass incorrect roleId: GET /role/{id} - get role by Id





|  |
| --- |
| {  "httpStatus": "NOT\_FOUND",  "message": "Role with ID 300 not found",  "details": "uri=/role/300;client=0:0:0:0:0:0:0:1",  "timestamp": 1753730182962  } |