Reimbursement Management System API

In this hands-on, you need to create a REST Api in Spring boot, which is used to manage and provide reimbursement for the employees.

Models:

Role Model:

Field Name	Datatype	Primary Key	Foreign Key	Comments
id	Integer	Yes		autoincrement
rolemodel	String			Unique

User Model:

Filed Name	Datatype	Primary Key	Foreign Key	Comments
id	Integer	Yes		autoincrement
username	String			
email	String			Unique
password	String			
role	Integer		Yes	

Reimbursement Model:

Field Name	Datatype	Primary Key	Foreign Key	Comments
id	**Integer**	**Yes**		**autoincrement**
employeeName	**String**			
employeeNumber	**Integer**			
productType	**String**			
productName	**String**			
productProvider	**String**			
amount	**Double**			
submissionDate	**String**			**Current Date**
comments	**String**			
isApproved	**String**			**Default value = pending**
approvalDate	**String**			**Current Date**
approveRemarks	**String**			
employeeId	**Integer**		**Yes**	

If any of the above validations are failing, return with a response code of 400 - Bad Request

Initialize the database with the following data.

Role:

id	**rolename**
1	EMPLOYEE
2	SUPERVISOR

User:

username	**password**	**email**	**role**
employee1	emp123	employee1@dummy.com	1
employee2	emp234	employee2@dummy.com	1
supervisor1	supp234	Supervisor1@dummy.com	2

Implement JWT based authorization and authentication with the two above mentioned roles.

EMPLOYEE and SUPERVISOR should be identified from the JWT token

JWT token should be sent as a Bearer token in Authorization request header. For example:

Authorization value would be ***Bearer<SPACE><JWT TOKEN>***

End-Points Marked in

- Red is accessible only by employees.
- Blue is accessible only by supervisor.
- Green is accessible by both employees and supervisor.

All other endpoints except login should be authenticated and authorized with the above.

Endpoints:

1. POST METHOD - **/login**

Authenticates and creates JWT token with respective authorization.

Request Parameters	Success Response	Error Response

Json Body:
{
 	
"email":"employee2@dummy.com",
"password":"emp234"
}
	200 OK
	
	{
	"jwt":"your_jwt_token",
	"status":200
	}
		400 Bad Request on individual credentials

2. POST METHOD - **/reimbursement/add**

Adds a new reimbursement. Note: employeeId should point to the owner who created the reimbursement.

Request Parameters	Success Response

Json Body
{
"empName": "emp1Name",
"empNumber": 25255,
"productType": "myProdType",
"productName": "MyProdName",
"productProvider": "ProdProvider",
"amount": 454.0,
"comments":"my comment on product"
}
	201 CREATED
	{
	"id": 1,
	"empName": "emp1Name",
	"empNumber": 25255,
	"productType": "myProdType",
	"productName": "MyProdName",
	"productProvider": "ProdProvider",
	"amount": 454.0,
	"submissionDate": "2024-12-03",
	"isApproved": "pending",
	"approvalDate": null,
	"approvedRemarks": null,
	"comments": "my comment on product",
	"employeeId": 3
	}

3. GET METHOD - **/reimbursement/ {employeeNumber}**

To get all the details of the reimbursement data with the path variable as employeeNumber with

status code 200.

If no data available for given value, then return “no data available” as a response with |

status code 400. E.g., /reimbursement/ value

4. PATCH METHOD - **/reimbursement/update/ {id}**

Updates the reimbursements approved status, approval Date and approverRemarks .

Request Parameters	Success Response

JSON Body -
{
"isApproved":"approved",
"approvedRemars":"Approved from our end"
}
	200 OK
	{
	"id": 1,
	"empName": "emp1Name",
	"empNumber": 25255,
	"productType": "myProdType",
	"productName": "MyProdName",
	"productProvider": "ProdProvider",
	"amount": 454.0,
	"submissionDate": "2024-12-04",
	"isApproved": "approved",
	"approvalDate": "2024-12-04",
	"approvedRemarks": "Approved from our end",
	"comments": "my comment on product",
	"employeeId": 1

	}

Error Response: 400 Bad Request on invalid credentials.

5. DELETE METHOD - **/reimbursement/delete {id}**

Note: This Method requires authentication. User who was authenticated and have role **“Employee”** and creator of the given id object, they can be able to access this endpoint.

Get the reimbursement data object with given id from reimbursement detail model, delete it and return “deleted successfully” with status code 204.

If the given id is not found, then return “not found” with status code 400.

If the authenticated user is not a creator of the given id object, then return “you don't have permission” with status code 403.