



## **GloRepay Portal**

Spring HandsOn Assessment

**Assessment Flavour - Java Developer**

**Complexity** - Medium

**Target Band** - Band x / Band 1 / Band 2

**Title of the Project** -

**Time:** 3 hours

**Topics Covered:**

Technology	Topics
Spring Web	REST API
Hibernate	Spring Data JPA
Core Java	Exception Handling

## SOFTWARE REQUIREMENTS

- [Java Latest Version](#)
- Any IDE that supports Java Application Development
  - [Intelij](#) or [STS](#)

## TABLE OF CONTENT

<b>SOFTWARE REQUIREMENTS</b>	<b>2</b>
DOWNLOADABLE STARTER CODE LINK (GITLAB)	2
GITLAB GENERAL INSTRUCTIONS	4
PROBLEM STATEMENT	6
PROJECT IMPLEMENTATION	7
Entity Relationship	8
Note:	8
<b>PROJECT BACKEND IMPLEMENTATION DETAILS</b>	<b>9</b>
Postman Testing	10
Note	16
<b>RUNNING THE BACKEND APPLICATION</b>	<b>17</b>

## **PURPOSE**

The GloRepay is an Employee Reimbursement System and is a REST API project to be developed using Java Spring Boot. The system allows employees to manage their expense reimbursements by providing various reimbursement modes and tracking their expenditures.

## **PROBLEM STATEMENT**

You are a backend developer working on the Employee Reimbursement System project. The project aims to develop a RESTful API using the Spring Boot framework. The Employee Reimbursement System is a web-based application that allows employees to manage their reimbursement claims for various expenses.

## **PROJECT IMPLEMENTATION**

Database Design: Application schema and tables

Note: You can ignore the create table command if your application is configured to auto-create tables using Hibernate/JPA properties.

Table Structure with data types

Employee		Description
id	Long	It Must be of Integer data type only. It is the primary ID that is unique for each user. Example: 1

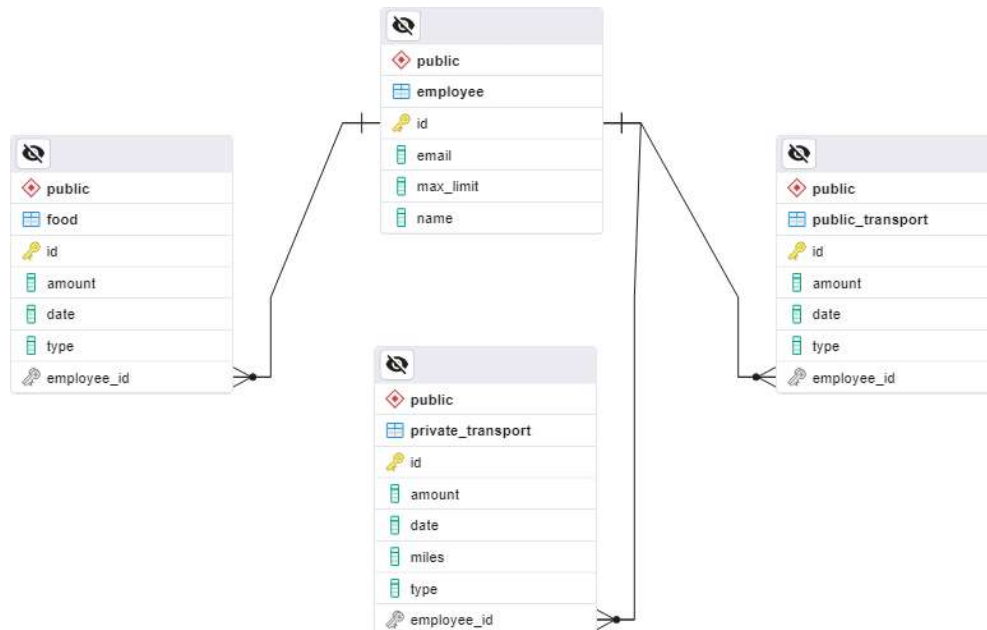
name	String	It must be of String data type. It is the name chosen by the user for the profile. Example: "JohnDoe"
email	String	It must be of String data type. It is the user's email address used for communication and login. Example: "johndoe@example.com"
maxLimit	Integer	It must be of Integer data type. It is the maximum amount allowed for reimbursement. Example: "password123"

PublicTransport		Description
id	Long	It must be of Integer data type only. It is the primary ID that is unique for each product. Example: 1
type	String	It must be of String data type. It is the name of the product. Example: "BUS"
date	Date	It must be of Date data type. It is the date on which the expense is made for public transport. Example: "yyyy-MM-dd"
amount	Integer	It must be of Integer data type. It is the expense that employees bore for the public transport. Example: 600

PrivateTransport		Description
id	Long	It must be of Integer data type only. It is the primary ID that is unique for each product. Example: 1
type	String	It must be of String data type. It is the name of the product. Example: "BUS"
date	Date	It must be of Date data type. It is the date on which the expense is made for public transport. Example: "yyyy-MM-dd"
miles	Integer	It must be of Integer data type. It is the distance traveled by the employee. Example: 10
amount	Integer	It must be of Integer data type. It is the expense that employees bore for the public transport. Example: 600

Food		Description
id	Long	It must be of Integer data type only. It is the primary ID that is unique for each product. Example: 1
type	String	It must be of String data type. It is the name of the product. Example: "LUNCH"
date	Date	It must be of Date data type. It is the date on which the expense is made for public transport. Example: "yyyy-MM-dd"
amount	Integer	It must be of Integer data type. It is the expense that employees bore for the public transport. Example: 600

## Entity Relationship



### Note:

There is bidirectional mapping which is one-to-many and many-to-one for the Entities as shown in the diagram.

## PROJECT BACKEND IMPLEMENTATION DETAILS

Application flow with functionality:

Note: Define appropriate custom exceptions for invalid values and handle the exceptions with appropriate error messages.

### API Details:

Employee Entity			
API Endpoints	HttpMethod	Description	Status Code
/api/employee	POST	Create a new employee.	201 Created
		Throws an exception with an informative message if the data is invalid.	400 Bad Request
/api/employee/{id}	GET	Retrieve details of a specific employee.	200 OK
		Throws an exception with an informative message if the employee ID is invalid.	404 Not Found

PublicTransport Enttiy			
API Endpoints	HttpMethod	Description	Status Code
/api/employee/{id}/publicTransport	POST	Create a new public transport entry	201 Created
		Throws an exception with an informative message if the data is invalid.	400 Bad Request
/api/employee/{id}/publicTransport/{publicTransport}	GET	Retrieve details of a specific employee public transport	200 OK
		Throws an exception with an informative message if the employee ID or	404 Not Found



		public transport id is invalid.	
--	--	---------------------------------	--

PrivateTransport Enttiy			
API Endpoints	HttpMethod	Description	Status Code
/api/employee/{id}/privateTransport	POST	Create a new private transport entry	201 Created
		Throws an exception with an informative message if the data is invalid.	400 Bad Request
/api/employee/{id}/publicTransport/{privateTransport}	GET	Retrieve details of a specific employee private transport	200 OK
		Throws an exception with an informative message if the employee ID or private transport id is invalid.	404 Not Found

Food Enttiy			
API Endpoints	HttpMethod	Description	Status Code
/api/employee/{id}/food	POST	Create a new food entry	201 Created
		Throws an exception with an informative message if the data is invalid.	400 Bad Request
/api/employee/{id}/food/{foodId}	GET	Retrieve details of a specific employee food expense	200 OK
		Throws an exception with an informative message if the employee ID or food id is invalid.	404 Not Found

## Postman Testing

### POST operation for Employee Entity

#### Case 1 - The user provided valid data

The screenshot shows the Postman interface for a POST request to `localhost:8080/api/employee`. The request body is a JSON object with the following data:

```
1 {
2   "name": "Vipin N",
3   "email": "vipin.nitoor@globallogic.com"
4 }
```

The response is displayed in the bottom pane, showing a status of 201 Created, a time of 299 ms, and a size of 250 B. The response body is a JSON object with the following data:

```
1 {
2   "id": 52,
3   "name": "Vipin N",
4   "email": "vipin.nitoor@globallogic.com",
5   "maxlimit": 5000
6 }
```

#### Case 2 - User provided the invalid data

The screenshot shows the Postman interface for a POST request to `localhost:8080/api/employee`. The request body is a JSON object with the following data:

```
1 {
2   "name": "",
3   "email": "vipin.nitoor@globallogic.com"
4 }
```

The response is displayed in the bottom pane, showing a status of 400 Bad Request, a time of 21 ms, and a size of 233 B. The response body is a JSON object with the following data:

```
1 {
2   "name": "Employee name should have 3 characters minimum",
3   "email": "Email should be valid"
4 }
```

### GET Operation for Employee Entity

## Case 1 - The user provided a valid Id

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/api/employee/2
- Status:** 200 OK
- Time:** 17 ms
- Size:** 255 B
- Body (JSON):**

```
{
  "id": 2,
  "name": "Shubham Gondane",
  "email": "shubham.gondane@globallogic.com",
  "maxLimit": 5000
}
```

## Case 2 - User-Provided the invalid Id

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/api/employee/20
- Status:** 400 Bad Request
- Time:** 7 ms
- Size:** 271 B
- Body (JSON):**

```
{
  "timestamp": "2024-03-20T09:48:06.738+00:00",
  "message": "Employee not found with id: '20'",
  "description": "uri=/api/employee/20"
}
```

## POST Operation for PublicTransport Entity

## Case 1 - The user provided a valid employee Id and valid data

POST localhost:8080/api/employee/2/publicTransport

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "type": "FLIGHT",
3   "date": "2023-01-04",
4   "amount": 700
5 }
```

Body Cookies Headers (5) Test Results Status: 201 Created Time: 25 ms Size: 246 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 102,
3   "type": "FLIGHT",
4   "date": "Wed Jan 04 00:00:00 IST 2023",
5   "amount": 700
6 }
```

## Case 2 - The user provided the invalid employee Id with valid data

POST localhost:8080/api/employee/20/publicTransport

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "type": "FLIGHT",
3   "date": "2023-01-04",
4   "amount": 700
5 }
```

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 7 ms Size: 287 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-03-20T09:49:24.609+00:00",
3   "message": "Employee not found with id: '20'",
4   "description": "uri=/api/employee/20/publicTransport"
5 }
```

## Case 3 - User provided the invalid data in the request body

POST localhost:8080/api/employee/20/publicTransport Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

```

1 {
2   "type": "",
3   "date": null,
4   "amount": -1
5 }

```

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 12 ms Size: 256 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "date": "Date cannot be null",
3   "amount": "Amount must be a positive number or zero",
4   "type": "Type cannot be blank"
5 }

```

## GET Operation for PublicTransport Entity

Case 1 - The user provided a valid employee Id and public transport Id

GET localhost:8080/api/employee/2/publicTransport/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (5) Test Results Status: 200 OK Time: 248 ms Size: 232 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": 1,
3   "type": "FLIGHT",
4   "date": "2023-01-01 05:30:00.0",
5   "amount": 700
6 }

```

## Case 2 - The user provided an invalid employee Id

GET localhost:8080/api/employee/20/publicTransport/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 6 ms Size: 289 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-03-20T09:51:49.513+00:00",
3   "message": "Employee not found with id: '20'",
4   "description": "uri=/api/employee/20/publicTransport/1"
5 }
```

## Case 3 - The user provided an invalid publicTransportId

GET localhost:8080/api/employee/2/publicTransport/10 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 9 ms Size: 296 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-03-20T09:52:13.307+00:00",
3   "message": "PublicTransport not found with id: '10'",
4   "description": "uri=/api/employee/2/publicTransport/10"
5 }
```

## POST Operation for PrivateTransport Entity

Case 1 - The user provided a valid employee Id and valid data

POST localhost:8080/api/employee/1/privateTransport

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

```

1 {
2   "type": "CAR",
3   "date": "2023-09-12",
4   "miles": 100,
5   "amount": 50
6 }
7

```

Body Cookies Headers (5) Test Results Status: 201 Created Time: 20 ms Size: 253 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": 2,
3   "type": "CAR",
4   "date": "2023-09-12T00:00:00+00:00",
5   "miles": 100,
6   "amount": 50
7 }

```

Case 2 - The user provided the invalid employee Id with valid data

POST localhost:8080/api/employee/10/privateTransport

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded **raw** binary GraphQL JSON

```

1 {
2   "type": "CAR",
3   "date": "2023-09-12",
4   "miles": 100,
5   "amount": 50
6 }
7

```

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 7 ms Size: 288 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "timestamp": "2024-03-20T09:54:20.919+00:00",
3   "message": "Employee not found with id: '10'",
4   "description": "uri=/api/employee/10/privateTransport"
5 }

```

## Case 3 - User provided the invalid data in the request body

The screenshot shows a REST client interface with a POST request to `localhost:8080/api/employee/10/privateTransport`. The request body is a JSON object with several errors: `"type": ""`, `"date": null`, `"miles": 0`, and `"amount": -1`. The response status is `400 Bad Request` with a time of `9 ms` and size of `290 B`. The response body contains error messages for each field.

```
1 {
2   "type": "",
3   "date": null,
4   "miles": 0,
5   "amount": -1
6 }
```

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 9 ms Size: 290 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "date": "Date cannot be null",
3   "amount": "Amount must be a positive number",
4   "type": "Type cannot be blank",
5   "miles": "Miles must be a positive number"
6 }
```

## GET Operation for PrivateTransport Entity

### Case 1 - The user provided a valid employee Id and private transport Id

The screenshot shows a REST client interface with a GET request to `localhost:8080/api/employee/1/privateTransport/1`. The response status is `200 OK` with a time of `17 ms` and size of `248 B`. The response body is a JSON object representing a private transport entity.

GET localhost:8080/api/employee/1/privateTransport/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	Bulk Edit
Key	Value	Description	

Body Cookies Headers (5) Test Results Status: 200 OK Time: 17 ms Size: 248 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "type": "Car",
4   "date": "2023-09-12T00:00:00.000+00:00",
5   "miles": 100,
6   "amount": 50
7 }
```



## Case 2 - The user provided an invalid employee Id

GET localhost:8080/api/employee/10/privateTransport/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 6 ms Size: 290 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-03-20T09:57:33.669+00:00",
3   "message": "Employee not found with id: '10'",
4   "description": "uri=/api/employee/10/privateTransport/1"
5 }
```

## Case 3 - The user provided an invalid privateTransportId

GET localhost:8080/api/employee/1/privateTransport/10 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time: 9 ms Size: 298 B Save as example

Pretty Raw Preview Visualize JSON

```
1 {
2   "timestamp": "2024-03-20T09:58:14.360+00:00",
3   "message": "PrivateTransport not found with id: '10'",
4   "description": "uri=/api/employee/1/privateTransport/10"
5 }
```

## POST Operation for Food Entity

Case 1 - The user provided a valid employee Id and valid data

POST localhost:8080/api/employee/1/food

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "type": "LUNCH",
3   "date": "2023-12-24",
4   "amount": 50
5 }
6

```

Body Cookies Headers (5) Test Results

Status: 201 Created Time: 25 ms Size: 245 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "id": 152,
3   "type": "LUNCH",
4   "date": "2023-12-24T00:00:00.000+00:00",
5   "amount": 50
6 }

```

Case 2 - The user provided the invalid employee Id with valid data

POST localhost:8080/api/employee/10/food

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "type": "LUNCH",
3   "date": "2023-12-24",
4   "amount": 50
5 }
6

```

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time: 6 ms Size: 276 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "timestamp": "2024-03-20T10:02:26.188+00:00",
3   "message": "Employee not found with id: '10'",
4   "description": "uri=/api/employee/10/food"
5 }

```

Case 3 - User provided the invalid data in the request body

POST localhost:8080/api/employee/1/food

Params Authorization Headers (9) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1 {
2   "type": " ",
3   "date": null,
4   "amount": -1
5 }
6

```

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time: 6 ms Size: 256 B Save as example

Pretty Raw Preview Visualize JSON

```

1 {
2   "date": "Date cannot be null",
3   "amount": "Amount must be a positive number or zero",
4   "type": "Type cannot be blank"
5 }

```

## GET Operation for Food Entity

Case 1 - The user provided a valid employee Id and private transport Id

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/api/employee/1/food/1
- Status:** 200 OK
- Time:** 11 ms
- Size:** 238 B
- Body (JSON):**

```
{  "id": 1,  "type": "LUNCH",  "date": "2012-09-14T00:00:00.000+00:00",  "amount": 50}
```

Case 2 - The user provided an invalid employee Id

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/api/employee/10/food/1
- Status:** 400 Bad Request
- Time:** 5 ms
- Size:** 278 B
- Body (JSON):**

```
{  "timestamp": "2024-03-20T10:04:18.724+00:00",  "message": "Employee not found with id: '10'",  "description": "uri=/api/employee/10/food/1"}
```

## Case 3 - The user provided an invalid privateTransportId

The screenshot shows a REST client interface. At the top, a GET request is configured with the URL `localhost:8080/api/employee/1/food/10`. Below the URL bar, tabs for Params, Authorization, Headers (7), Body, Pre-request Script, Tests, and Settings are visible. The Params tab is active, showing a table for Query Params with columns Key, Value, and Description. A table with two rows is shown below the header, with the first row containing 'Key' and 'Value'.

Below the Params tab, the Body tab is active, showing the response in JSON format. The status is 400 Bad Request, Time: 9 ms, and Size: 274 B. The response body is:

```
1 {
2   "timestamp": "2024-03-20T10:04:41.404+00:00",
3   "message": "Food not found with id: '10'",
4   "description": "uri=/api/employee/1/food/10"
5 }
```

### Note

- Use the Utility class provided in the codebase for mapping DTO to Entity and vice versa
- Use the Exception handling utility class to handle the exception in the application as and when needed

## RUNNING THE BACKEND APPLICATION

Test the application by running the main file in the starter code.

**Note:** Main file code is commented. Uncomment the code, test the application, and comment it back before submitting the assessment.

### EVALUATION METRICS

Sl. No.	Assessing Parameter	Marks
1	Creating the well-defined Controller Layer	5
2	Creating the JPA Entity with appropriate Entity Mapping	5
3	Creating the Dto class for all the entities	5
4	Handling the invalid data in the Dto class using the Hibernate Validator	5
5	Creating the Repository layer	5
6	Creating the Business Logic for all the entities with proper exception handling.	20
7	Coding standards and best practices	5
Total Marks		50 Marks