

## Retorno de Testes de Integração da API Rest

- Permission -> New Endpoint

The screenshot shows the API client interface with the 'Permission' folder selected. The 'POST new Endpoint' item is highlighted. The JSON tab displays the request body: 

```
1 {  
2   "url": "/professor/"  
3 }
```

. The Preview tab displays the response body: 

```
1 {  
2   "message": "New endpoint successfully registered"  
3 }
```

.

- Permission -> New UserType

The screenshot shows the API client interface with the 'Permission' folder selected. The 'POST new UserType' item is highlighted. The JSON tab displays the request body: 

```
1 {  
2   "description": "Professor"  
3 }
```

. The Preview tab displays the response body: 

```
1 {  
2   "message": "Type of user successfully registered"  
3 }
```

.

- Permission -> New Permission

The screenshot shows the API client interface with the 'Permission' folder selected. The 'POST New Permission' item is highlighted. The JSON tab displays the request body: 

```
1 {  
2   "idUser": 1,  
3   "url": "/coordenador/"  
4 }
```

. The Preview tab displays the response body: 

```
1 {  
2   "command": "INSERT",  
3   "rowCount": 1,  
4   "oid": 0,  
5   "rows": [],  
6   "fields": [],  
7   "_types": {  
8     "_types": {  
9       "arrayParser": {},  
10      "builtins": {  
11        "BOOL": 16,  
12        "BYTEA": 17,  
13        "CHAR": 18,
```

- Permission -> List Permission

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

- Left Panel:** A sidebar with a search filter and a list of endpoints under the 'Permission' folder. The endpoints are: `POST new Endpoint`, `POST new Usertype`, `POST New Permission`, and `GET list permission` (which is highlighted).
- Body Panel:** Currently empty.
- Preview Panel:** Displays a JSON array of three permission objects:

```
1 [
2   {
3     "id": 3,
4     "endpoint": "/professor/",
5     "name": "Vitória Roza"
6   },
7   {
8     "id": 2,
9     "endpoint": "/gestorespaco/",
10    "name": "Vitória Roza"
11  },
12  {
13    "id": 1,
14    "endpoint": "/coordenador/",
```

- Users -> New User

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

- Left Panel:** A sidebar with a search filter and a list of endpoints under the 'Users' folder. The endpoints are: `POST New User` (highlighted) and `PUT Update User`.
- Body Panel:** Displays a JSON object representing a new user:

```
1 {
2   "name": "Lucas Furtado",
3   "username": "Furtado",
4   "email": "furtadolucas@gmail.com",
5   "password": "teste12345"
6 }
```
- Preview Panel:** Displays a JSON response indicating success:

```
1 {
2   "message": "User successfully registered"
3 }
```

- Users -> Update User

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

- Left Panel:** A sidebar with a search filter and a list of endpoints under the 'Users' folder. The endpoints are: `POST New User` and `PUT Update User` (highlighted).
- Body Panel:** Displays a JSON object representing an updated user:

```
1 {
2   "name": "Lucas Antunes",
3   "username": "Furtado",
4   "email": "lucasfurtado@gmail.com",
5   "password": "Teste12"
6 }
```
- Preview Panel:** Displays a JSON response indicating success:

```
1 {
2   "message": "Data updated successfully"
3 }
```

- Type\_Location -> Create

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

- Left Panel:** A sidebar with a search filter and a list of endpoints under the 'Type\_Location' folder. The endpoints are: `POST create` (highlighted), `GET list`, and `DEL delete`.
- Body Panel:** Displays a JSON object representing a new location type:

```
1 {
2   "description": "Auditório"
3 }
```
- Preview Panel:** Displays a JSON response indicating success:

```
1 {
2   "message": "Location type has been created"
3 }
```

- Type\_Location -> List

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

- Left Panel:** A sidebar with a search filter, a folder icon, and the name 'Type\_Location'. Below it, three methods are listed: 'POST create', 'GET list' (highlighted in purple), and 'DEL delete'.
- Body Panel:** Currently empty.
- Auth Panel:** Currently empty.
- Query Panel:** Currently empty.
- Preview Panel:** Displays a JSON array of three objects:

```
1 [
2   {
3     "id": 1,
4     "description": "Laboratório"
5   },
6   {
7     "id": 2,
8     "description": "Sala de Reunião"
9   },
10  {
11    "id": 3,
```

- Type Location -> Delete

The screenshot shows the REST client interface with the 'delete' endpoint selected. The components are:

- Left Panel:** Similar to the previous screenshot, but 'DEL delete' is highlighted in red.
- Body Panel:** Empty.
- Auth Panel:** Empty.
- Query Panel:** Empty.
- Preview Panel:** Displays a JSON object:

```
1 {
2   "message": "Location type has been deleted"
3 }
```

- Location -> New Location

The screenshot shows the REST client interface for the 'Location' resource. The components are:

- Left Panel:** A sidebar with a search filter, a folder icon, and the name 'Location'. Below it, six methods are listed: 'POST new Location' (highlighted in green), 'PUT update Location', 'GET search location', 'GET detail location', 'GET list location', and 'DEL delete location'.
- Body Panel:** Displays a JSON object:

```
1 {
2   "tp_location": "3",
3   "comments": "Auditório II",
4   "capacity": "550"
5 }
```
- Auth Panel:** Empty.
- Query Panel:** Empty.
- Header Panel:** Shows 1 header.
- Preview Panel:** Displays a JSON object:

```
1 {
2   "message": "Place successfully registered"
3 }
```

- Location -> Update Location

The screenshot shows the REST client interface with the 'update Location' endpoint selected. The components are:

- Left Panel:** Similar to the previous screenshot, but 'PUT update Location' is highlighted in orange.
- Body Panel:** Displays a JSON object:

```
1 {
2   "tp_location": "2",
3   "comments": "Laboratório de Teste",
4   "capacity": "100"
5 }
```
- Auth Panel:** Empty.
- Query Panel:** Empty.
- Header Panel:** Shows 1 header.
- Preview Panel:** Displays a JSON object:

```
1 {
2   "message": "Location changed successfully"
3 }
```

- Location -> Search Location

The screenshot shows the Postman interface with a GET request selected. The left sidebar lists the following actions: POST new Location, PUT update Location, GET search location (highlighted), GET detail location, GET list location, and DEL delete location. The main panel displays the URL `http://localhost:3333/location/?term=Lab&type` and a query parameter table with 'term' set to 'Lab' and 'type' set to 'value'. The right sidebar shows the JSON response in the Preview tab:

```
1 [
2   {
3     "id": 2,
4     "comments": "Laboratório de Física",
5     "capacity": 90,
6     "type": 1
7   }
8 ]
```

- Location -> Detail Location

The screenshot shows the Postman interface with a GET request selected. The left sidebar lists the following actions: POST new Location, PUT update Location, GET search location, GET detail location (highlighted), GET list location, and DEL delete location. The main panel is empty, and the right sidebar shows the JSON response in the Preview tab:

```
1 [
2   {
3     "id": 2,
4     "comments": "Laboratório de Física",
5     "capacity": 90,
6     "type": 1
7   }
8 ]
```

- Location -> List Location

The screenshot shows the Postman interface with a GET request selected. The left sidebar lists the following actions: POST new Location, PUT update Location, GET search location, GET detail location, GET list location (highlighted), and DEL delete location. The main panel is empty, and the right sidebar shows the JSON response in the Preview tab:

```
1 [
2   {
3     "id": 2,
4     "comments": "Laboratório de Física",
5     "capacity": 90,
6     "type": 1
7   },
8   {
9     "id": 3,
10    "comments": "Sala de Reunião I",
11    "capacity": 10,
12    "type": 2
13  },
14  {
15    "id": 5,
16    "comments": "Auditório I",
17    "capacity": 300,
18    "type": 3
19  }
20 ]
```

- Location -> Delete Location

The screenshot shows the TCC API client interface. The top bar indicates a DELETE request to the endpoint `baseurl /location/ 5` with a status of **200 OK**, a response time of **301 ms**, and a body size of **43 B**. The left sidebar lists the 'Location' resource with methods: POST new Location, PUT update Location, GET search location, GET detail location, GET list location, and DEL delete location. The main panel shows the 'Body' tab, which is empty. The 'Preview' tab on the right displays the JSON response: 

```
1 {
2   "message": "Location successfully deleted"
3 }
```

- Reservation -> New Reserve

The screenshot shows the TCC API client interface for a POST request to create a new reservation. The left sidebar lists the 'Reservation' resource with methods: POST New Reserve, PUT Update Reserve, GET List Reserves, GET detail Reserve, and DEL Delete Reserve. The main panel shows the 'JSON' tab with the request body: 

```
1 {
2   "userId": "1",
3   "locationId": "3",
4   "date": "2020/10/01",
5   "time_start": "20:40",
6   "time_end": "22:00",
7   "classes": "Nenhuma",
8   "discipline": "Nenhuma",
9   "comments": "Reunião Diretoria"
10 }
```

 The 'Preview' tab on the right displays the JSON response: 

```
1 {
2   "message": "Successful booking"
3 }
```

- Reserve -> Update Reserve

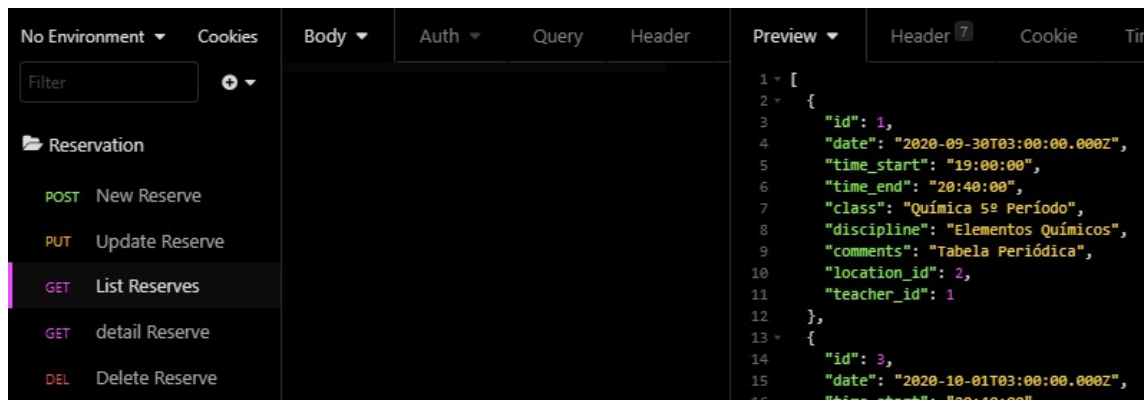
The screenshot shows the TCC API client interface for a PUT request to update a reservation. The left sidebar lists the 'Reservation' resource with methods: POST New Reserve, PUT Update Reserve, GET List Reserves, GET detail Reserve, and DEL Delete Reserve. The main panel shows the 'JSON' tab with the request body: 

```
1 {
2   "userId": "1",
3   "locationId": "5",
4   "date": "2020/10/10",
5   "time_start": "21:00",
6   "time_end": "22:00",
7   "classes": "Nenhuma",
8   "discipline": "Nenhuma",
9   "comments": "Palestra - Inteligência Artificial"
10 }
```

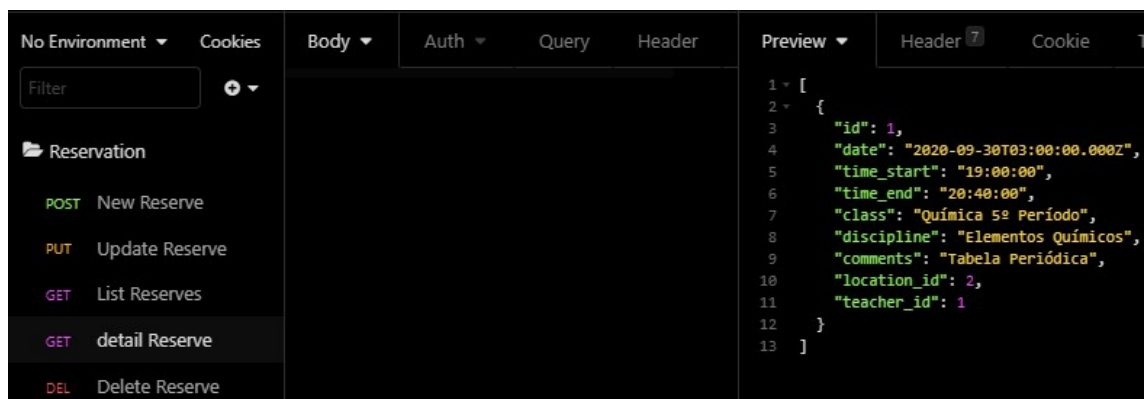
 The 'Preview' tab on the right displays the JSON response: 

```
1 {
2   "message": "Reservation updated successfully"
3 }
```

- Reserve -> List Reserves



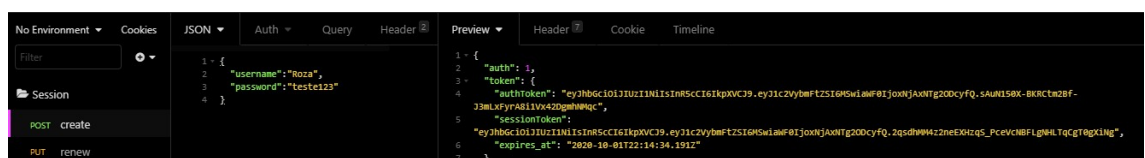
- Reserve -> Detail Reserves



- Reserve -> Delete Reserve



- Session -> Create



- Session -> Renew

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

- Left Panel:** A sidebar with a "Filter" input, a "Session" folder icon, and two actions: "POST create" and "PUT renew". The "PUT renew" action is selected.
- JSON Tab:** The main editor shows a JSON body for a PUT request:

```
1 {  
2   "authToken": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Im5wIiwiaWF0IjoxNjA4MDEyOTYyLm09cLyP8FwSseAMgQ18XX1A4jmeYiBNoY",  
3   "userId": "1"  
4 }.
```
- Preview Tab:** The right panel shows the response body:

```
1 {  
2   "status": "updated"  
3 }
```

- User Location -> Assign

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

- Left Panel:** A sidebar with a "Filter" input, a "User Location" folder icon, and three actions: "POST assign", "DEL unassign", and "GET list location per user". The "POST assign" action is selected.
- JSON Tab:** The main editor shows a JSON body for a POST request:

```
1 {  
2   "locationId": "5",  
3   "userId": "2"  
4 }
```
- Preview Tab:** The right panel shows the response body:

```
1 {  
2   "error": "Error when assigning responsibility"  
3 }
```

- User Location -> Unassign

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

- Left Panel:** A sidebar with a "Filter" input, a "User Location" folder icon, and three actions: "POST assign", "DEL unassign", and "GET list location per user". The "DEL unassign" action is selected.
- JSON Tab:** The main editor shows a single line of text: "1 ...".
- Preview Tab:** The right panel shows the response body:

```
1 {  
2   "message": "Excluded liability association"  
3 }
```

- User Location -> List Location per User

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

- Left Panel:** A sidebar with a "Filter" input, a "User Location" folder icon, and three actions: "POST assign", "DEL unassign", and "GET list location per user". The "GET list location per user" action is selected.
- JSON Tab:** The main editor shows a single line of text: "1 ...".
- Preview Tab:** The right panel shows the response body:

```
1 [  
2   {  
3     "id": 3,  
4     "location_id": 5,  
5     "user_id": 2  
6   }  
7 ]
```