

## Listar Atividades

The image shows a development environment with VS Code on the left and Postman on the right. In VS Code, the `atividade_controller.py` file is open, showing the `listar_atividades` method in the `AtividadeController` class. The method calls `atividade_model.listar_atividades()` and returns the result as JSON. The terminal shows the application running on `http://localhost:5002`. In Postman, the `GET /atividades` endpoint is selected, and the response is displayed in JSON format. The response is a 200 OK status with a body containing a list of activities.

```
1 from flask import Blueprint, jsonify, request
2 from models import atividade_model
3 from clients.pessoa_service_client import PessoaServiceClient
4
5 atividade_bp = Blueprint('atividade_bp', __name__)
6
7 @atividade_bp.route('/', methods=['GET'])
8 def listar_atividades():
9     atividades = atividade_model.listar_atividades()
10    return jsonify(atividades)
11
12 @atividade_bp.route('/<int:id_atividade>', methods=['GET'])
13 def obter_atividade(id_atividade):
14     try:
15         atividade = atividade_model.obter_atividade(id_atividade)
16         return jsonify(atividade)
17     except atividade_model.AtividadeNotFound:
18         return jsonify({'erro': 'Atividade não encontrada'}), 404
19
20 @atividade_bp.route('/', methods=['POST'])
21 def criar_atividade():
22     data = request.get_json()
23     id_disciplina = data.get('id_disciplina')
24     enunciado = data.get('enunciado')
25     respostas = data.get('respostas', [])
26
27     nova_atividade = atividade_model.criar_atividade(id_disciplina, enunciado, respostas)
28     return jsonify(nova_atividade), 201
29
30 @atividade_bp.route('/<int:id_atividade>', methods=['PUT'])
31 def atualizar_atividade(id_atividade):
32     data = request.get_json()
33     id_disciplina = data.get('id_disciplina')
```

Postman Response (JSON):

```
{
  "enunciado": "Crie um app de todo em Flask",
  "id_atividade": 1,
  "id_disciplina": 1,
  "respostas": [
    {
      "id_aluno": 1,
      "nota": 9,
      "resposta": "todo.py"
    },
    {
      "id_aluno": 2,
      "resposta": "todo.zip.rar"
    },
    {
      "id_aluno": 4,
      "nota": 10,
      "resposta": "todo.zip"
    }
  ]
}
```

## Obter Atividade por ID

The image shows a development environment with VS Code on the left and Postman on the right. In VS Code, the `atividade_controller.py` file is open, showing the `obter_atividade` method in the `AtividadeController` class. The method calls `atividade_model.obter_atividade(id_atividade)` and returns the result as JSON. The terminal shows the application running on `http://localhost:5002`. In Postman, the `GET /atividades/1` endpoint is selected, and the response is displayed in JSON format. The response is a 200 OK status with a body containing a single activity object.

```
1 from flask import Blueprint, jsonify, request
2 from models import atividade_model
3 from clients.pessoa_service_client import PessoaServiceClient
4
5 atividade_bp = Blueprint('atividade_bp', __name__)
6
7 @atividade_bp.route('/', methods=['GET'])
8 def listar_atividades():
9     atividades = atividade_model.listar_atividades()
10    return jsonify(atividades)
11
12 @atividade_bp.route('/<int:id_atividade>', methods=['GET'])
13 def obter_atividade(id_atividade):
14     try:
15         atividade = atividade_model.obter_atividade(id_atividade)
16         return jsonify(atividade)
17     except atividade_model.AtividadeNotFound:
18         return jsonify({'erro': 'Atividade não encontrada'}), 404
19
20 @atividade_bp.route('/', methods=['POST'])
21 def criar_atividade():
22     data = request.get_json()
23     id_disciplina = data.get('id_disciplina')
24     enunciado = data.get('enunciado')
25     respostas = data.get('respostas', [])
26
27     nova_atividade = atividade_model.criar_atividade(id_disciplina, enunciado, respostas)
28     return jsonify(nova_atividade), 201
29
30 @atividade_bp.route('/<int:id_atividade>', methods=['PUT'])
31 def atualizar_atividade(id_atividade):
32     data = request.get_json()
33     id_disciplina = data.get('id_disciplina')
```

Postman Response (JSON):

```
{
  "enunciado": "Crie um app de todo em Flask",
  "id_atividade": 1,
  "id_disciplina": 1,
  "respostas": [
    {
      "id_aluno": 1,
      "nota": 9,
      "resposta": "todo.py"
    },
    {
      "id_aluno": 2,
      "resposta": "todo.zip.rar"
    },
    {
      "id_aluno": 4,
      "nota": 10,
      "resposta": "todo.zip"
    }
  ]
}
```

## Criar Atividade

The image displays two screenshots of a development environment, likely Visual Studio Code, showing the implementation of a REST API for creating and listing activities.

**Top Screenshot:**

- Left Panel (Code Editor):** Shows the `atividade_controller.py` file. The `criar_atividade` method is implemented, which takes a POST request, extracts data (disciplina, enunciado, respostas), and calls `atividade_model.criar_atividade` to create a new activity. The response is a 201 status code.
- Right Panel (API Network):** Shows the 'Criar Atividade' endpoint. The request is a POST to `localhost:5002/atividades` with a JSON body: 

```
{ "id_disciplina": 1, "enunciado": "Nova atividade de teste", "respostas": [] }
```

. The response is a 201 status code, indicating successful creation.

**Bottom Screenshot:**

- Left Panel (Code Editor):** Shows the `atividade_controller.py` file. The `listar_atividades` method is implemented, which calls `atividade_model.listar_atividades` to retrieve all activities. The response is a 200 status code.
- Right Panel (API Network):** Shows the 'Listar Atividades' endpoint. The request is a GET to `localhost:5002/atividades`. The response is a 200 status code, indicating successful retrieval of activities.

## Atualizar Atividade

The image displays a development environment with two main windows. The left window shows the source code for a Flask application, and the right window shows the API Network interface with test results.

**Source Code (Left Window):**

```
1 from flask import Blueprint, jsonify, request
2 from models import atividade_model
3 from clients.pessoa_service_client import PessoaServiceClient
4
5 atividade_bp = Blueprint('atividade_bp', __name__)
6
7 @atividade_bp.route('/', methods=['GET'])
8 def listar_atividades():
9     atividades = atividade_model.listar_atividades()
10    return jsonify(atividades)
11
12 @atividade_bp.route('/<int:id_atividade>', methods=['GET'])
13 def obter_atividade(id_atividade):
14     try:
15         atividade = atividade_model.obter_atividade(id_atividade)
16         return jsonify(atividade)
17     except atividade_model.AtividadeNotFound:
18         return jsonify({'erro': 'Atividade não encontrada'}), 404
19
20 @atividade_bp.route('/', methods=['POST'])
21 def criar_atividade():
22     data = request.get_json()
23     id_disciplina = data.get('id_disciplina')
24     enunciado = data.get('enunciado')
25     respostas = data.get('respostas', [])
26
27     nova_atividade = atividade_model.criar_atividade(id_disciplina, enunciado, respostas)
28     return jsonify(nova_atividade), 201
29
30 @atividade_bp.route('/<int:id_atividade>', methods=['PUT'])
31 def atualizar_atividade(id_atividade):
32     data = request.get_json()
33     id_disciplina = data.get('id_disciplina')
34     enunciado = data.get('enunciado')
```

**API Network (Right Window):**

The API Network interface shows a collection of tests for the 'Atividade Service Te... / Atualizar Atividade' endpoint. The selected test is a PUT request to 'localhost:5002/atividades/1'.

**Test Results:**

The test results show a successful PUT request with a 200 OK status. The response body is a JSON object:

```
{
  "id_disciplina": 1,
  "enunciado": "Atividade de teste atualizada",
  "id_atividade": 1,
  "id_disciplina": 1,
  "respostas": [
    {
      "id_aluno": 1,
      "nota": 8,
      "resposta": "resposta1"
    }
  ]
}
```

The bottom window shows the same source code and API Network interface, but with a different test selected: 'GET /atividades'. The test results show a successful GET request with a 200 OK status. The response body is a JSON object:

```
{
  "enunciado": "Atividade de teste atualizada",
  "id_atividade": 1,
  "id_disciplina": 1,
  "respostas": [
    {
      "id_aluno": 1,
      "nota": 8,
      "resposta": "resposta1"
    }
  ]
}
```

## Excluir Atividade

The image displays a development environment with two main windows: a code editor on the left and a Postman API client on the right.

**Code Editor (Left):** Shows the implementation of a REST API for managing activities. The code is written in Python using Flask and SQLAlchemy. It includes endpoints for listing, creating, updating, and deleting activities.

```
1 from flask import Blueprint, jsonify, request
2 from models import atividade_model
3 from clients.pessoa_service_client import PessoaServiceClient
4
5 atividade_bp = Blueprint('atividade_bp', __name__)
6
7 @atividade_bp.route('/', methods=['GET'])
8 def listar_atividades():
9     atividades = atividade_model.listar_atividades()
10    return jsonify(atividades)
11
12 @atividade_bp.route('/<int: id>/atividades', methods=['GET'])
13 def obter_atividade(id_atividade):
14     try:
15         atividade = atividade_model.obter_atividade(id_atividade)
16         return jsonify(atividade)
17     except atividade_model.AtividadeNotFound:
18         return jsonify({'erro': 'Atividade não encontrada'}), 404
19
20 @atividade_bp.route('/', methods=['POST'])
21 def criar_atividade():
22     data = request.get_json()
23     id_disciplina = data.get('id_disciplina')
24     enunciado = data.get('enunciado')
25     respostas = data.get('respostas', [])
26
27     nova_atividade = atividade_model.criar_atividade(id_disciplina, enunciado, respostas)
28     return jsonify(nova_atividade), 201
29
30 @atividade_bp.route('/<int: id>/atividades/', methods=['PUT'])
31 def atualizar_atividade(id_atividade):
32     data = request.get_json()
33     id_disciplina = data.get('id_disciplina')
34     enunciado = data.get('enunciado')
```

**Postman (Right):** Shows the API being tested. The first screenshot shows a DELETE request to `localhost:5002/atividades/1` with a status of 204 NO CONTENT. The second screenshot shows a GET request to `localhost:5002/atividades` with a status of 200 OK, returning a JSON array of activities.

```
1 [
2   {
3     "enunciado": "Crie um servidor que envia email em Flask",
4     "id_atividade": 2,
5     "id_disciplina": 1,
6     "respostas": [
7       {
8         "id_aluno": 4,
9         "nota": 10,
10        "resposta": "email.zip"
11      }
12    ]
13  },
14  {
15    "enunciado": "Nova atividade de teste",
16    "id_atividade": 3,
17    "id_disciplina": 1,
18    "respostas": []
19  }
20 ]
```