

mongoDB®

API PHP COM MONGODB

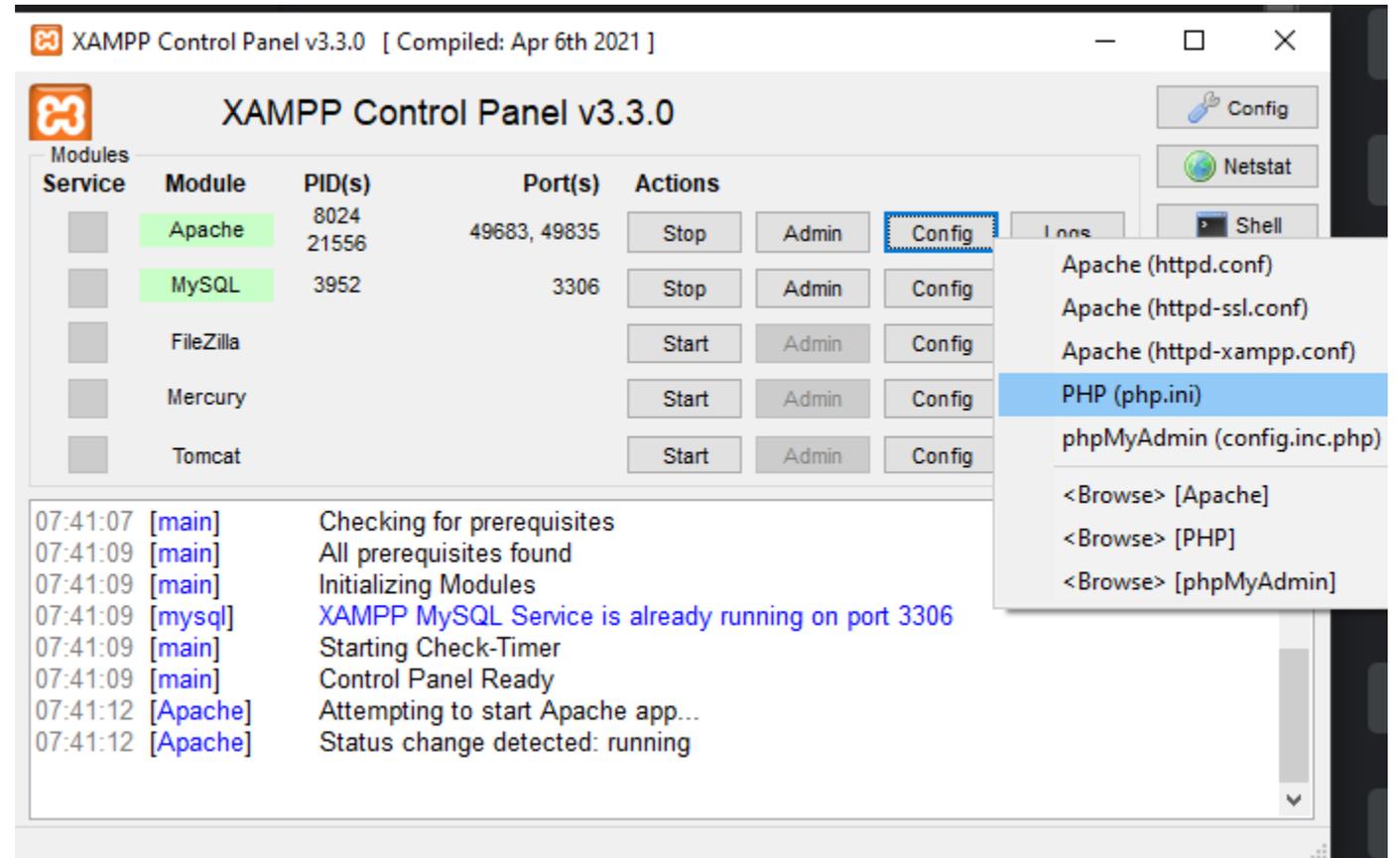
Prof. Me. Hélio Esperidião

# Instalar driver

- Download:
  - [https://windows.php.net/downloads/pecl/releases/mongodb/1.13.0/php\\_mongodb-1.13.0-7.4-ts-vc15-x64.zip](https://windows.php.net/downloads/pecl/releases/mongodb/1.13.0/php_mongodb-1.13.0-7.4-ts-vc15-x64.zip)
- Posicionar o arquivo : **php\_mongo.dll** em **C:\xampp\php\ext**

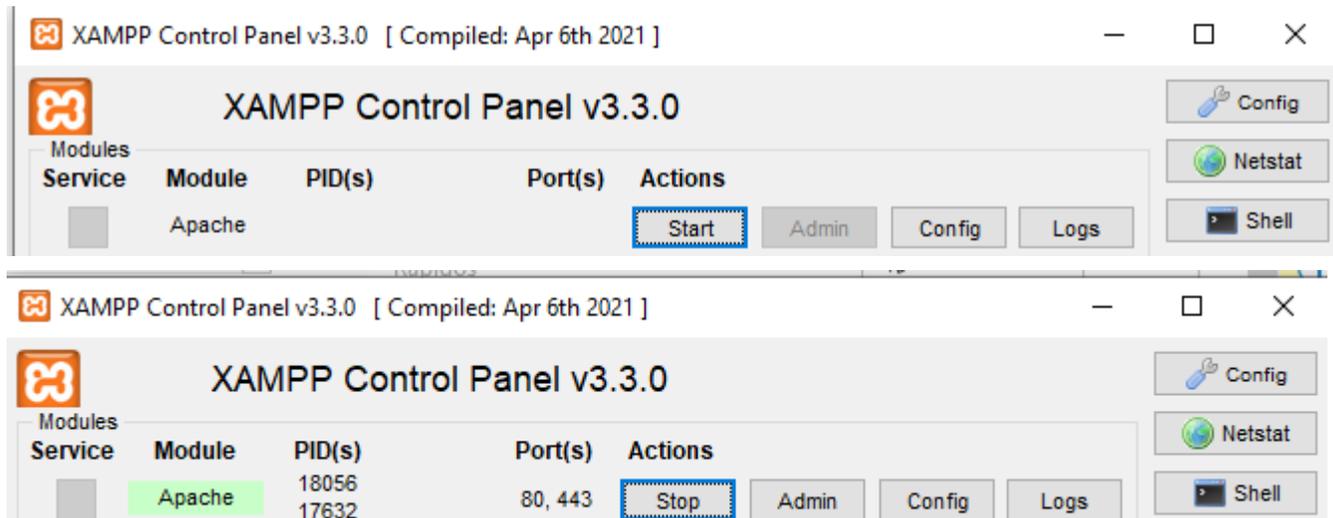
# Configuração: php.ini

- Abra o xampp vá em config do apache e escolha o arquivo **php.ini**



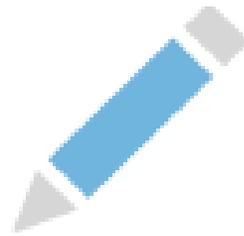
# Configure a extensão

- Acrescente a linha:
  - `extension=php_mongodb.dll`
- Salve o arquivo
- Pare e inicie novamente o servidor



```
php.ini - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
; directory, You may specify an absolute path to the library file:
;
; extension=/path/to/extension/mysqli.so
;
; Note : The syntax used in previous PHP versions ('extension=<ext>.
; 'extension='php_<ext>.dll') is supported for legacy reasons and ma
; deprecated in a future PHP major version. So, when it is possible,
; move to the new ('extension=<ext>') syntax.
;
; Notes for Windows environments :
;
; - Many DLL files are located in the extensions/ (PHP 4) or ext/ (P
; extension folders as well as the separate PECL DLL download (PHF
; Be sure to appropriately set the extension_dir directive.
;
extension=php_mongodb.dll
extension=bz2
extension=curl
;extension=ffi
;extension=ftp
extension=fileinfo
extension=gd2
extension=gettext
;extension=gmp
;extension=intl
;extension=imap
;extension=ldap
extension=mbstring
extension=exif ; Must be after mbstring as it depends on it
extension=mysqli
;extension=oci8_12c ; Use with Oracle Database 12c Instant Client
;extension=odbc
```

# Exemplo Completo CRUD Clientes - MongoDB



CREATE

**C**



READ

**R**



UPDATE

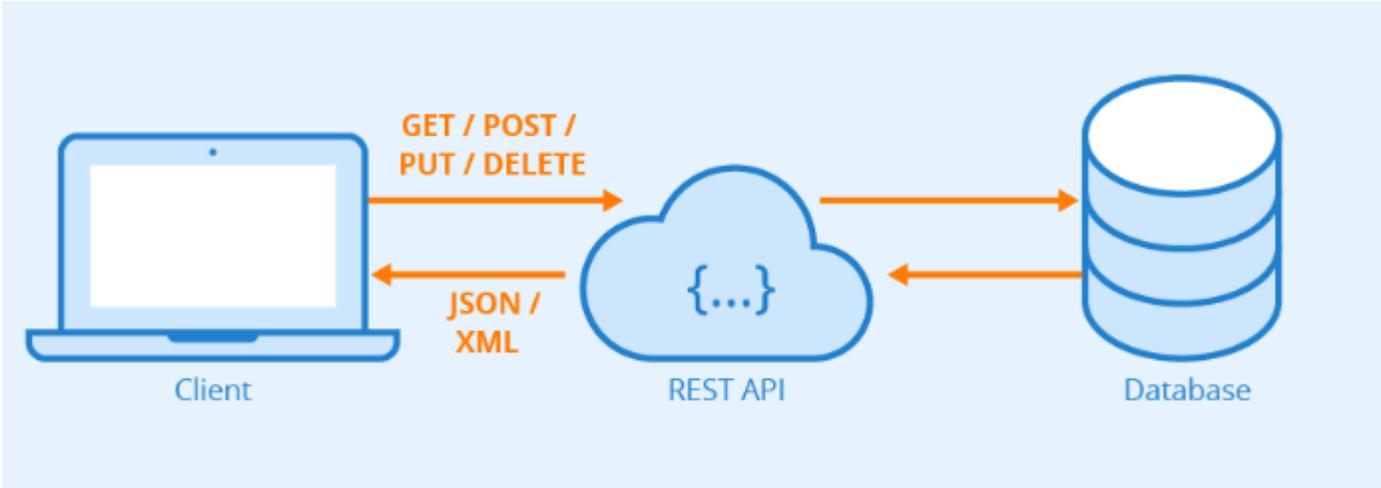
**U**



DELETE

**D**

# Arquitetura



# Arquitetura de pastas e arquivos.

- .htaccess
- Index.php
- Controle
  - Cliente\_cadastrar.php
  - Cliente\_excluir.php
  - Cliente\_buscar.php
  - Cliente\_listar.php
  - Cliente\_atualizar.php
- Modelo
  - BancoMongo.php
  - Cliente.php

# Rotas e recursos

VERBOS	URI(ROTA)	OPERAÇÃO	ARQUIVO RESPONSÁVEL
GET	<a href="http://localhost/clientes/">http://localhost/clientes/</a>	Retorna todos os clientes.	controle/Cliente_listar.php
GET	<a href="http://localhost/clientes/id">http://localhost/clientes/id</a>	Retorna um cliente pelo id	controle/Cliente_buscar.php
POST	<a href="http://localhost/clientes">http://localhost/clientes</a>	Cadastra um novo cliente	controle/Cliente_cadastrar.php
PUT	<a href="http://localhost/clientes/id">http://localhost/clientes/id</a>	Atualiza um cliente pelo seu id.	Controle/Cliente_atualizar.php
DELETE	<a href="http://localhost/clientes/id">http://localhost/clientes/id</a>	Exclui um cliente.	controle/Cliente_excluir.php

# .htaccess

RewriteEngine on

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule ^(.\*)\$ /index.php?path=\$1 [NC,L,QSA]

# Index.php (roteador)

VERBOS	URI(ROTA)	ARQUIVO RESPONSÁVEL
GET	/clientes/	controle/Cliente_listar.php
GET	/clientes/id	controle/Cliente_buscar.php
POST	/clientes	controle/Cliente_cadastrar.php
PUT	/clientes/id	Controle/Cliente_atualizar.php
DELETE	/clientes/id	controle/Cliente_excluir.php

# Index.php

```
<?php
require_once "modelo/Router.php";
$router = new Router();
$router->post('/clientes', function () {
    require_once "controle/Cliente_cadastrar.php";
});
$router->get('/clientes/', function () {
    require_once "controle/Cliente_listar.php";
});
$router->get('/clientes/(\w+)', function ($v1) {
    require_once "controle/Cliente_buscar.php";
});
$router->put('/clientes/(\w+)', function ($v1) {
    require_once "controle/Cliente_atualizar.php";
});
$router->delete('/clientes/(\w+)', function ($v1) {
    require_once "controle/Cliente_excluir.php";
});
$router->run();
```

# Estrutura dos Documentos

```
[{ "_id": { "$oid": "id" },  
  "nome": "",  
  "email": "",  
  "nascimento": "",  
  "salario": "",  
  "senha": ""}]
```

```
use MongoDB\Driver\Manager;
```

```
class BancoMongo{
```

```
    private $host = "localhost";
```

```
    private $usuario = "";
```

```
    private $senha = "";
```

```
    private $banco = "";
```

```
    private $porta = "27017";
```

```
    private $con = null;
```

```
    private function conectar() {
```

```
        $host= $this->host;
```

```
        $porta= $this->porta;
```

```
        $this->con = $mongo = new Manager("mongodb://$host:$porta");
```

```
    }
```

```
    public function getConexao(){
```

```
        if ($this->con == null) {
```

```
            $this->conectar();
```

```
        }
```

```
        return $this->con;
```

```
    }
```

```
}
```

# BancoMongo.php

# Cliente.php (1/8) - Atributos

```
include "BancoMongo.php";
```

Inclui o arquivo contendo a classe Banco

```
use MongoDB\Driver\BulkWrite;
```

```
use MongoDB\BSON\ObjectId;
```

```
use MongoDB\Driver\Query;
```

Importe as seguintes bibliotecas

```
class Cliente implements JsonSerializer {
```

```
    private $idCliente;
```

```
    private $nome;
```

```
    private $email;
```

```
    private $senha;
```

```
    private $nascimento;
```

```
    private $salario;
```

```
    private $banco;
```

```
    private $colecacao = "aulaPAW.Clientes";
```

Atributos da classe Cliente, observe que os atributos correspondem aos campos da estrutura da coleção

# Cliente.php (2/8) – jsonSerialize

```
public function jsonSerialize() {  
    $json= array();  
    $json['idCliente']=$this->getIdCliente();  
    $json['nome']=$this->getNome();  
    $json['email']=$this->getEmail();  
    $json['nascimento']=$this->getNascimento();  
    $json['salario']=$this->getSalario();  
    return $json;  
}
```

É chamado quando a instancia do objeto é convertida em json.

Observe que não é interessante enviar para o cliente a senha do usuário.

É chamado no echo `json_encode($vetor);`

# Cliente.php (3/8) - cadastrar()

```
public function cadastrar(){
```

```
    $this->banco = new BancoMongo();
```

```
    $documento = new BulkWrite();
```

```
    //cria um vetor associativo equivalente ao documento que será inserido
```

```
    $doc = ['nome' => $this->getNome(),
```

```
           'email' => $this->getEmail(),
```

```
           'nascimento' => $this->getNascimento(),
```

```
           'salario' => $this->getSalario(),
```

```
           'senha' => $this->getSenha(),
```

```
    ];
```

```
    $idCadastrado= (string) $documento->insert($doc); //recuperar o id do documento que será criado.
```

```
    $this->banco->getConexao()->executeBulkWrite($this->colecao, $documento); // escreve o documento
```

```
    $this->setIdCliente( $idCadastrado);
```

```
    return $this;
```

```
}
```

Por meio da instancia da classe banco recupera a conexão.  
Tento a conexão é possível executar comandos no sgbd

BulkWrite() -> permite inserir, atualizar e excluir documentos

# excluir()

```
public function excluir(){
    $documento = new BulkWrite();
    $this->banco = new BancoMongo();
    $filter = ['_id' => new ObjectID($this->idCliente)];
    $options = ['limit' => 1];

    $documento->delete($filter, $options);
    $this->banco->getConexao()->executeBulkWrite($this->colecao,
    $documento);
    return $this;
}
```

```
public function atualizar(){
    $this->banco = new BancoMongo();
    $documento = new BulkWrite();
    $filter = ['_id' => new ObjectID($this->idCliente)];
    $update = ['$set' => [ 'nome'=>$this->getNome(),
        'email'=>$this->getEmail(),
        'senha'=>$this->getSenha(),
        'salario'=> $this->getSalario(),
        'nascimento'=> $this->getNascimento()
    ]];
    $options = ['multi' => false, 'upsert' => false];
    $documento->update($filter, $update, $options);
    $cursor= $this->banco->getConexao()->executeBulkWrite($this->colecao,
    $documento);
    return true;
}
```

atualizar()

```
public function buscar($idCliente){
    $this->banco = new BancoMongo();
    $filter = ['_id' => new ObjectID($idCliente)];
    //echo $this->idCliente;
    $options = ['limit' => 1];
    $query = new Query($filter, $options);
    $cursor= $this->banco->getConexao()->executeQuery($this->colecao, $query);
    foreach ($cursor as $document) {
        $this->setIdCliente($idCliente);
        $this->setNome($document->nome);
        $this->setEmail($document->email);
        $this->setSalario($document->salario);
        $this->setNascimento($document->nascimento);
    }
    return $this;
}
```

# buscar(\$idCliente)

# listar

```
public function listar(){
    $this->banco = new BancoMongo();
    $vetorClientes = array();
    $i = 0;  $filter = array(); $options = [];
    $query = new Query($filter, $options);
    $cursor= $this->banco->getConexao()->executeQuery($this->colecao, $query);
    foreach ($cursor as $document) {
        $vetorClientes[$i]= new Cliente();
        $vetorClientes[$i]->setIdCliente((String) new ObjectID($document->_id) );
        $vetorClientes[$i]->setNome($document->nome);
        $vetorClientes[$i]->setEmail($document->email);
        $vetorClientes[$i]->setSalario($document->salario);
        $vetorClientes[$i]->setNascimento($document->nascimento);
        $i++;
    }
    return $vetorClientes;
}
```

# Get/set

```
public function getIdCliente(){return $this->idCliente;}
    public function setIdCliente($v){$this->idCliente=$v;}
    public function getNome(){return $this->nome;}
    public function setNome($nome){$this->nome = $nome;}
    public function getEmail(){return $this->email;}
    public function setEmail($v){$this->email=$v;}
    public function getSenha(){return $this->senha;}
    public function setSenha($v){$this->senha = $v;}
    public function getNascimento(){return $this->nascimento;}
    public function setNascimento($v){$this->nascimento=$v;}
    public function getSalario(){return $this->salario;}
    public function setSalario($v){$this->salario=$v;}

}
```

# controle/Cliente\_cadastrar.php - 1

```
include "modelo/Cliente.php";  
// cria um array para utilizar na resposta.  
$resposta = array();  
//recupera os dados(texto json) que vieram no corpo da requisicao.  
$jsonRecebido = file_get_contents('php://input');  
//converte os dados em um objeto json  
$obj = json_decode($jsonRecebido);  
//strip_tags remove tags html  
//dos dados vindos dos clientes  
//impede exemplo <b>helio</b>  
$nome = strip_tags($obj->nome);  
$email = strip_tags($obj->email);  
$senha = strip_tags($obj->senha);  
$salario = strip_tags($obj->salario);  
$nascimento = strip_tags($obj->nascimento);
```

# controle/Cliente\_cadastrar.php - 2

```
//verifica se os dados foram preenchidos corretamente
```

```
//este exemplo é simples
```

```
//verifique conforme a necessidade da aplicação.
```

```
if($nome==""){
```

```
    $resposta['cod']="erro";
```

```
    $resposta['msg']="Nome não preenchido";
```

```
}else if($email==""){
```

```
    $resposta['cod']="erro";
```

```
    $resposta['msg']="E-mail não preenchido";
```

```
}else{
```

# controle/Cliente\_cadastrar.php - 3

```
//após passar por todas as verificações
$cliente = new Cliente();
//passa os dados para o objeto
$cliente->setNome($nome);
$cliente->setEmail($email);
$cliente->setSenha($senha);
$cliente->setSalario($salario);
$cliente->setNascimento($nascimento);
//chama o método cadastrar
$resultado = $cliente->cadastrar();
```

# controle/Cliente\_cadastrar.php - 4

```
if($resultado==true){
    header('HTTP/1.1 201 Created');
    $resposta['cod']="ok";
    $resposta['msg']="cadastrado com sucesso";
    $resposta['POST']=$cliente;
}else{
    header('HTTP/1.1 200 ok');
    $resposta['cod']="erro";
    $resposta['msg']="Não foi cadastrado";
}
}
//converte o array resposta em json para
//enviar de resposta ao cliente
echo json_encode($resposta);
?>
```

# Testar o controle

## POST: http://localhost/clientes/

- Resposta: **201 Created**
- Observe que um json é enviado para a rota.
- O recurso retorna uma mensagem do servidor.

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

- Request:** Method: POST, URL: http://localhost/clientes/. The JSON body is: 

```
{
  "nome": "Hélio Esperidião",
  "email": "helioesperidiao@gmail.com",
  "senha": "batataQueijo",
  "salario": 11359.00,
  "nascimento": "1985/10/03"
}
```
- Response:** Status: 201 Created, Time: 23.8 ms, Size: 209 B. The JSON body is: 

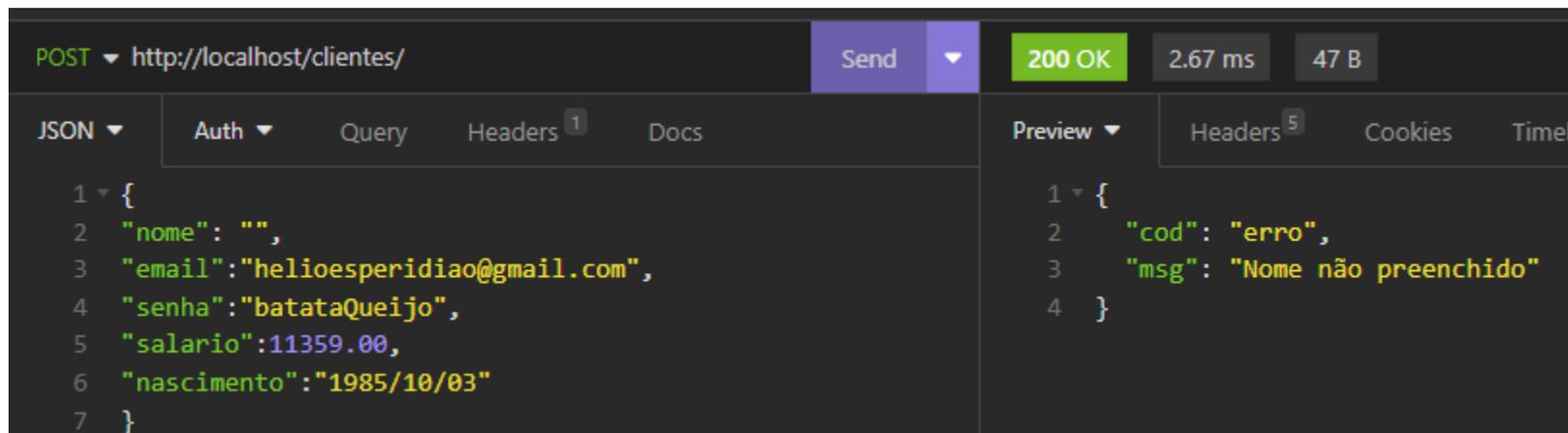
```
{
  "cod": "ok",
  "msg": "cadastrado com sucesso",
  "POST": {
    "idCliente": "6485bf60a056c09e6e0b83af",
    "nome": "Hélio Esperidião",
    "email": "helioesperidiao@gmail.com",
    "nascimento": "1985\10\03",
    "salario": "11359"
  }
}
```

# Testar o controle

## http://localhost/clientes/

Resposta: 200 ok

A requisição foi processada, mas **não** foi realizado o cadastro.



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

- Method: POST
- URL: http://localhost/clientes/
- Status: 200 OK
- Time: 2.67 ms
- Size: 47 B
- Request Body (JSON):

```
1 {
2   "nome": "",
3   "email": "helioesperidiao@gmail.com",
4   "senha": "batataQueijo",
5   "salario": 11359.00,
6   "nascimento": "1985/10/03"
7 }
```
- Response Body (JSON):

```
1 {
2   "cod": "erro",
3   "msg": "Nome não preenchido"
4 }
```

# controle/Cliente\_atualizar.php -1

PUT

/clientes/**id**

Controle/Cliente\_atualizar.php

```
<?php
include "modelo/Cliente.php";
$resposta = array();

$jsonRecebido = file_get_contents('php://input');
$obj = json_decode($jsonRecebido);

$nome = strip_tags($obj->nome);
$email = strip_tags($obj->email);
$senha = strip_tags($obj->senha);
$salario = strip_tags($obj->salario);
$nascimento = strip_tags($obj->nascimento);
```

Como os dados são enviados dos clientes no corpo da requisição os vetores `$_GET` e `$_POST` não podem ser utilizados.

Para recuperar os dados no corpo da requisição utilize:  
`file_get_contents('php://input');`

`$obj` é um objeto json criado a partir dos dados (texto json) que vieram no corpo da requisição.

## controle/Cliente\_atualizar.php -2

```
//verifica se os dados foram preenchidos corretamente
//este exemplo é simples
//verifique conforme a necessidade da aplicação.
if($nome==""){
    $resposta['cod']="erro";
    $resposta['msg']="Nome não preenchido";

}else if($email==""){
    $resposta['cod']="erro";
    $resposta['msg']="E-mail não preenchido";
}else{
```

# controle/Cliente\_atualizar.php -3

`$idCliente` veio do arquivo `index.php`

```
$cliente = new Cliente();

$cliente->setIdCliente($idCliente);
$cliente->setNome($nome);
$cliente->setEmail($email);
$cliente->setSenha($senha);
$cliente->setSalario($salario);
$cliente->setNascimento($nascimento);

$resultado = $cliente->atualizar();
```

# controle/Cliente\_atualizar.php -4

```
if($resultado==true){
    header("HTTP/1.1 200 OK");
    $resposta['cod']="ok";
    $resposta['msg']="Atualizado com sucesso";
    $resposta['PUT']=$cliente;
}else{
    header("HTTP/1.1 200 OK");
    $resposta['cod']="erro";
    $resposta['msg']="Não foi atualizado";
}

}

//converte o array resposta em json para
//enviar de resposta ao cliente
echo json_encode($resposta);
```

# Testar o controle

## PUT: http://localhost/clientes/id

- OBSERVE QUE FOI ENVIADO UM PUT.

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

- Request:** Method: PUT, URL: http://localhost/clientes/6485bf60a056c09e6e0b83af. The request body is a JSON object: 

```
{  "nome": "Hélio Lourenço Esperidião Ferreira",  "email": "helioesperidiao@gmail.com",  "senha": "batataQueijo",  "salario": 11359.00,  "nascimento": "1985/10/03"}
```
- Response:** Status: 200 OK, Time: 3.87 ms, Size: 231 B. The response body is a JSON object: 

```
{  "cod": "ok",  "msg": "Atualizado com sucesso",  "PUT": {    "idCliente": "6485bf60a056c09e6e0b83af",    "nome": "Hélio Lourenço Esperidião Ferreira",    "email": "helioesperidiao@gmail.com",    "nascimento": "1985\10\03",    "salario": "11359"  }}
```

# controle/Cliente\_buscar.php -1

`$idCliente` veio do arquivo `index.php`

```
<?php
```

```
include "modelo/Cliente.php";
```

```
$resposta = array();
```

```
$cliente = new Cliente();
```

```
$clientes = $cliente->buscar($idCliente);
```

```
header("HTTP/1.1 200 OK");
```

```
echo json_encode($clientes);
```

# Testar o controle

## GET: http://localhost/clientes/id

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

- Method: GET
- URL: http://localhost/clientes/6485bf60a056c09e6e0b83af
- Status: 200 OK
- Response Time: 2.97 ms
- Response Size: 181 B
- Response Type: JSON
- Response Content: 

```
1 {
2   "idCliente": "6485bf60a056c09e6e0b83af",
3   "nome": "Hélio Lourenço Esperidião Ferreira",
4   "email": "helioesperidiao@gmail.com",
5   "nascimento": "1985\10\03",
6   "salario": "11359"
7 }
```

# controle/Cliente\_excluir.php -1

```
include "modelo/Cliente.php";  
$resposta = array();  
$cliente = new Cliente();  
$cliente->setIdCliente($idCliente);  
$resultado = $cliente->excluir();
```

\$idCliente veio do arquivo index.php

## controle/Cliente\_excluir.php -2

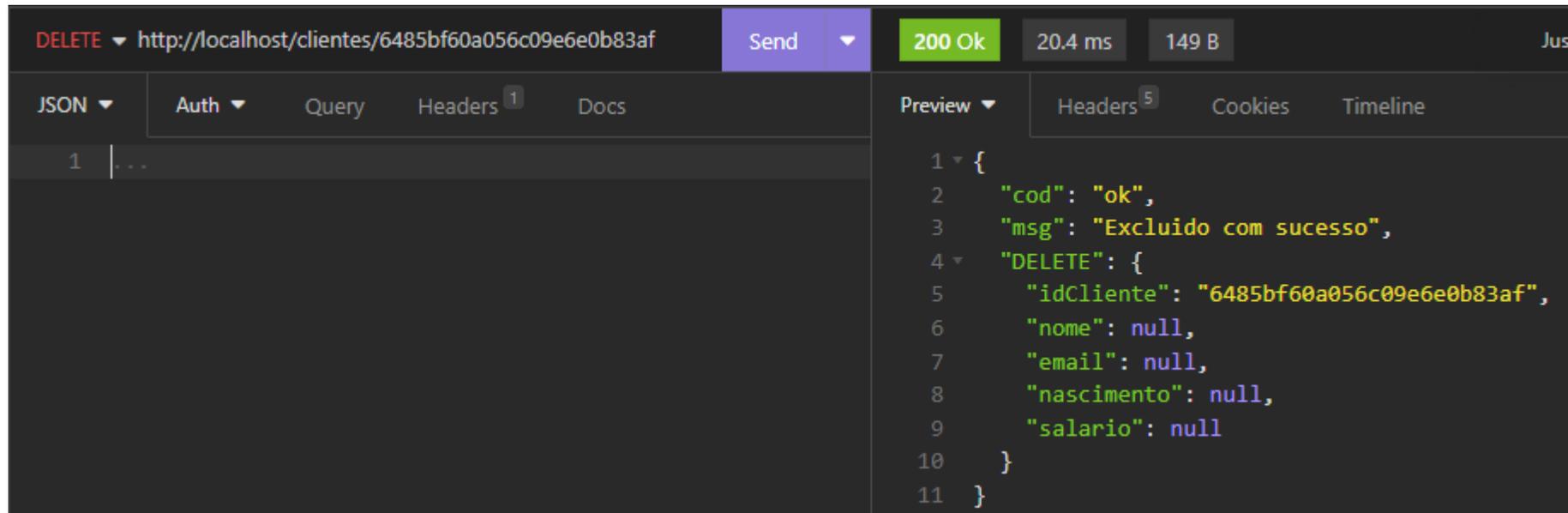
```
if ($resultado == true) {
    $resposta['cod'] = "ok";
    $resposta['msg'] = "Excluido com sucesso";
    $resposta['DELETE'] = $cliente;
} else {
    $resposta['cod'] = "erro";
    $resposta['msg'] = "Não foi Excluido";
}

header("HTTP/1.1 200 Ok");
echo json_encode($resposta);
```

# Testar o controle

## DELETE: <http://localhost/clientes/id>

- OBSERVE QUE FOI ENVIADO UM DELETE



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

- Method:** DELETE
- URL:** <http://localhost/clientes/6485bf60a056c09e6e0b83af>
- Status:** 200 Ok
- Time:** 20.4 ms
- Size:** 149 B
- Response Body (JSON):**

```
1 ...
2 {
3   "cod": "ok",
4   "msg": "Excluido com sucesso",
5   "DELETE": {
6     "idCliente": "6485bf60a056c09e6e0b83af",
7     "nome": null,
8     "email": null,
9     "nascimento": null,
10    "salario": null
11  }
```

# controle/Cliente\_listar.php - 1

```
<?php
include "modelo/Cliente.php";
$resposta = array();

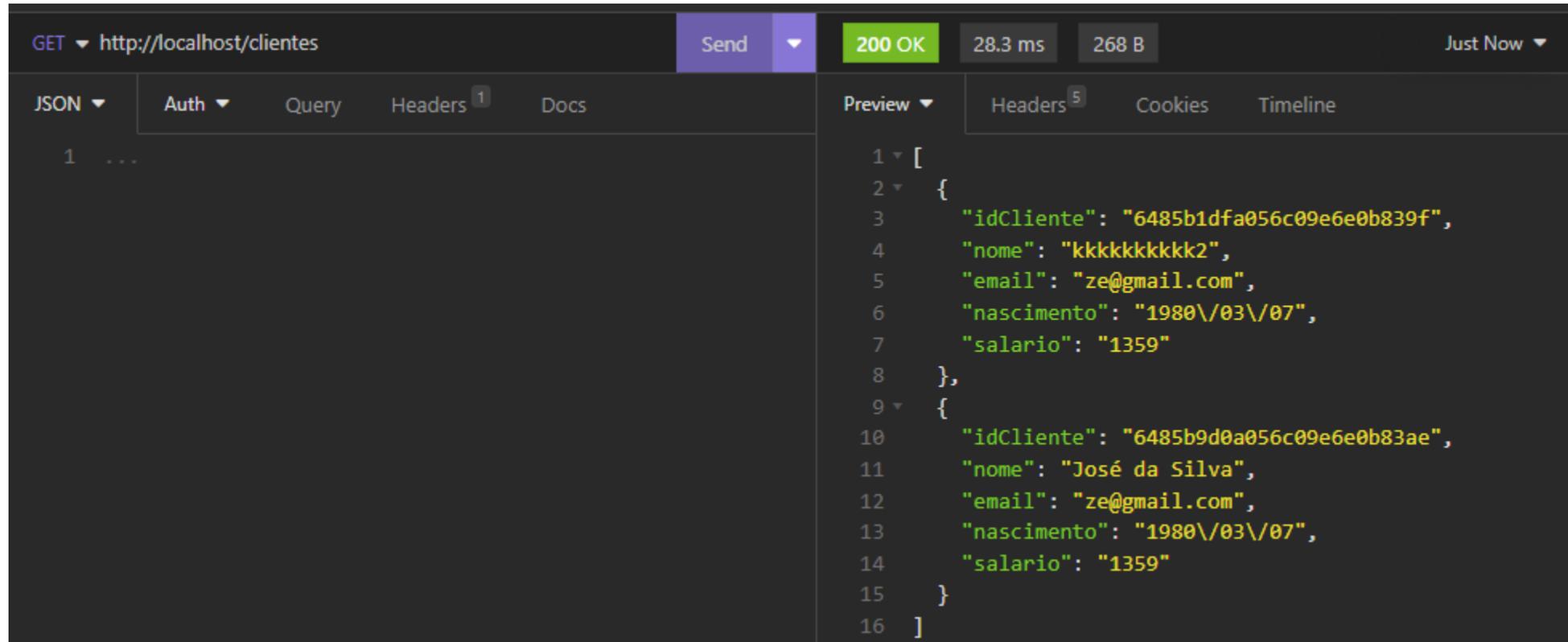
$cliente = new Cliente();

$clientes = $cliente->listar();

header("HTTP/1.1 200 OK");
echo json_encode($clientes);
```

# Testar o controle

## GET: http://localhost/clientes



GET http://localhost/clientes

Send 200 OK 28.3 ms 268 B Just Now

JSON Auth Query Headers 1 Docs

1 ...

Preview Headers 5 Cookies Timeline

```
1 [
2   {
3     "idCliente": "6485b1dfa056c09e6e0b839f",
4     "nome": "kkkkkkkkkk2",
5     "email": "ze@gmail.com",
6     "nascimento": "1980\03\07",
7     "salario": "1359"
8   },
9   {
10    "idCliente": "6485b9d0a056c09e6e0b83ae",
11    "nome": "José da Silva",
12    "email": "ze@gmail.com",
13    "nascimento": "1980\03\07",
14    "salario": "1359"
15  }
16 ]
```