



# IPG

Politécnico  
da Guarda

Escola Superior  
de Tecnologia e Gestão

## APIRESTFUL

---

**Curso(s):** TESP Desenvolvimento de Aplicações  
Informáticas.

**Unidade(s) Curricular(es):** Sistemas Distribuídos

**Ano Letivo:** 2º

**Docente:** Pedro Pinto

**Estudantes:** João Paiva Nº 1700477  
Wesley Santos Nº 1700480

**Data:** 20/01/2019

---

## INTRODUÇÃO

---

O documento tem como finalidade apresentar um relatório geral do trabalho apresentando como foi elaborada a API Restful.

REST é um acrônimo para Representational State Transfer. É a arquitetura de padrões da Web e o protocolo HTTP. O estilo arquitetural REST descreve seis restrições que foram originalmente comunicadas por Roy Fielding em sua tese de doutorado e define a base do estilo RESTful como:

1. Interface Uniforme
2. Sem estado
3. Em cache
4. Servidor cliente
5. Sistema em camadas
6. Código sob Demanda (opcional)

Os aplicativos RESTful usam solicitações HTTP para executar quatro operações denominadas CRUD (C: create, R: read, U: update e D: delete). Criar e / ou atualizar é usado para postar dados, obter dados de leitura / listagem e excluir para remover dados.

RESTful é composto de métodos como; URL base, URL, tipos de mídia, etc.

---

## API RESTFUL

---

### 1. FERRAMENTAS UTILIZADAS

#### a. Node.js

Link download node.js:

<https://nodejs.org/en/download/>

#### b. MongoDB

Link de instalação MongoDB:

[https://translate.googleusercontent.com/translate\\_c?depth=1&hl=pt-BR&rurl=translate.google.com&sl=auto&sp=nmt4&tl=pt-BR&u=https://nodejs.org/en/download/package-manager/&xid=17259,15700022,15700124,15700186,15700190,15700201,15700248&usg=ALkJrhg0UCjNpEHhdHeMXfkBOOHdZ70r7w](https://translate.googleusercontent.com/translate_c?depth=1&hl=pt-BR&rurl=translate.google.com&sl=auto&sp=nmt4&tl=pt-BR&u=https://nodejs.org/en/download/package-manager/&xid=17259,15700022,15700124,15700186,15700190,15700201,15700248&usg=ALkJrhg0UCjNpEHhdHeMXfkBOOHdZ70r7w)

#### c. Visual Code

Link download visual code:

<https://code.visualstudio.com/download>

#### d. Postman

Link download Postman:

<https://www.getpostman.com/>

### 2. INICIAMOS O PROJETO

Para iniciarmos o projeto teremos de criar uma pasta conforme os comandos abaixo:

**Mkdir** listacontactos

**Cd** listacontactos

**Npm** init

### 3. Instalação das dependências

Para instalar as dependências tivemos de instalar os seguintes comandos:

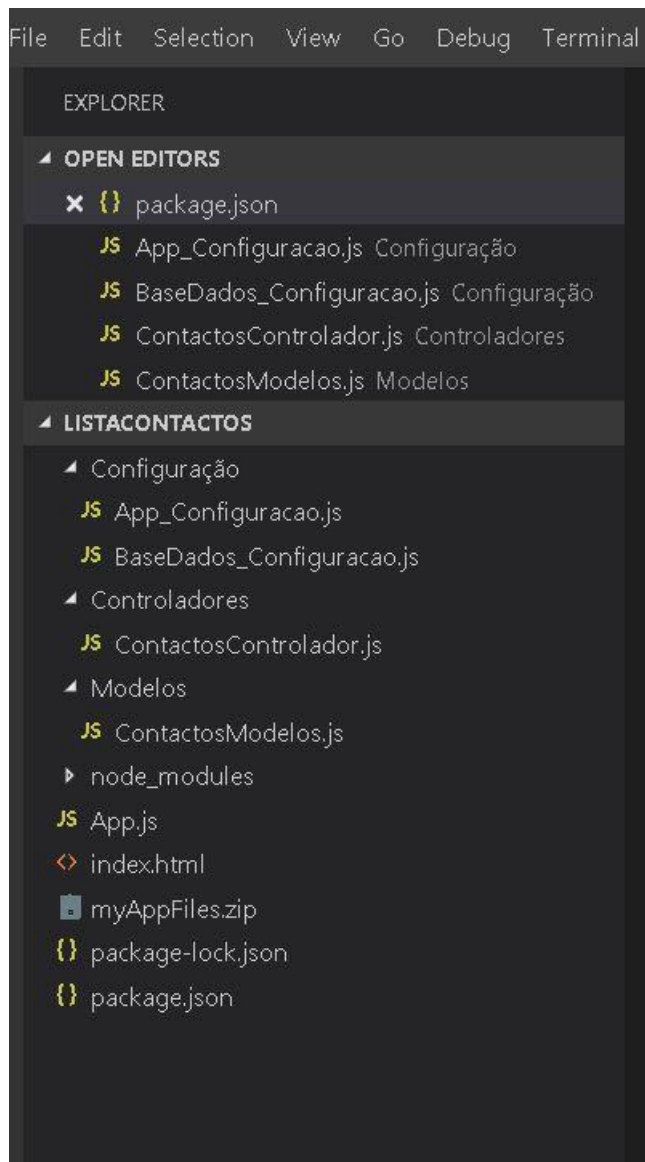
```
npm install nodemon --save
```

```
npm install express --save
```

```
npm install body-parser --save
```

```
npm install mongoose --save
```

De seguida criamos as seguintes pastas e ficheiros e o resultado foi o seguinte:



#### 4. Configuração da Aplicação

```
1  const express = require('express');
2  var bodyParser = require('body-parser');
3  var port = '1997';
4  const mongoose = require('mongoose');
5  var app = module.exports = express();
6
7  mongoose.connect('mongodb://JoaoP:qwerty123@ds155714.mlab.com:55714/listacontactos');
8  mongoose.connection.once('open', () => {
9    |   console.log("A conexão à Base de Dados foi bem sucedida")
10  });
11
12  app.listen(port);
13
14  app.use(bodyParser.urlencoded({extended:true}));
15  app.use(bodyParser.json());
16
17  app.use(function(req, res, next){
18    |   res.setHeader('Access-Control-Allow-Origin', '*');
19    |   res.setHeader('Access-Control-Allow-Methods', 'GET,POST,PUT,DELETE');
20    |   res.setHeader('Access-Control-Allow-Headers', 'X-Requested-With,content-type,Authorization');
21    |   next();
22  })
```

#### 5. Configuração da Base de Dados

```
1  var mongoose = require('mongoose');
2
3  var urlString = 'mongodb://JoaoP:qwerty123@ds155714.mlab.com:55714/listacontactos';
4
5  //var urlString = 'mongodb://localhost/API';
6
7  mongoose.connect(urlString, function(err, res){
8    |   if(err){
9    |     |   console.log('Não foi possível a conexão a:' + urlString);
10    |   }
11    |   else{
12    |     |   console.log('Conexão bem sucedida a:' + urlString);
13    |   }
14  });
```

## 6. Controlador dos Contactos

```
1 var contacto = require('../Modelos/ContactosModelos');
2 {
3   exports.save = function(nome, telefone, tipotelefone, email, tipoemail, endereco, tipoendereco, grupos, callback){
4     new contacto({
5       'nome': nome,
6       'telefone': telefone,
7       'tipotelefone': tipotelefone,
8       'email': email,
9       'tipoemail': tipoemail,
10      'endereco': endereco,
11      'tipoendereco': tipoendereco,
12      'grupos': grupos,
13    }).save(function(error, contacto){
14      if(error){
15        callback({error: '0 Contacto não foi inserido com sucesso'}});
16      }else{
17        callback(contacto);
18      }
19    });
20 }
21
22 exports.list = function(callback){
23   contacto.find({}, function(error, contacto){
24     if(error){
25       callback({error: '0 Contacto não se encontra na lista'}});
26     }else{
27       callback(contacto);
28     }
29   });
30 }
31 }
```

```
32
33 exports.delete = function(id, callback){
34   contacto.findById(id, function(error, contacto){
35     if (error){
36       callback({error: '0 Contacto não foi apagado com sucesso'}});
37     }else{
38       contacto.remove(function(error){
39         if(!error){
40           callback({resposta:"Contacto apagado com sucesso"})
41         }
42       });
43     }
44   })
45 }
46 }
47 }
```

## 7. Modelo Contactos

```
1 var mongoose = require('mongoose');
2 var Schema = mongoose.Schema;
3
4 var ContactoSchema = new Schema({
5   nome: String,
6   telefone: String,
7   tipotelefone: String,
8   email: String,
9   tipoemail: String,
10  endereco: String,
11  tipoendereco: String,
12  grupos: String,
13 });
14
15 module.exports = mongoose.model('contacto', ContactoSchema);
16
17
```

## 8. Configuração da Aplicação

```
1 var app = require('./Configuração/App_Configuracao');
2 var Base = require('./Configuração/BaseDados_Configuracao');
3 var contacto = require('./Modelos/ContactosModelos');
4 var contactocontrolador = require('./Controladores/ContactosControlador');
5
6 app.get('/', function(req, res){
7   res.end('Contactos')
8 });
9
10
11 //route
12 app.get('/contacto', function(req, res){
13   contactocontrolador.list(function(resp){
14     res.json(resp);
15   })
16 });
17
18 app.post('/registo', function(req, res){
19   var nome = req.body.nome;
20   var telefone = req.body.telefone;
21   var tipotelefone = req.body.tipotelefone;
22   var email = req.body.email;
23   var tipoemail = req.body.tipoemail;
24   var endereco = req.body.endereco;
25   var tipoendereco = req.body.tipoendereco;
26   var grupos = req.body.grupos;
27
28   contactocontrolador.save(nome, telefone, tipotelefone, email, tipoemail, endereco, tipoendereco, grupos, function(resp){
29     res.json(resp);
30   });
31
32 });
33
```

```

33
34 app.delete('/eliminar/:id', function(req, res){
35     var id = req.params.id;
36
37     contactocontrolador.delete(id, function(resp){
38         res.json(resp);
39     });
40 });
41

```

## 9. Configuração da Interface

```

<!DOCTYPE html>

<html lang="en">
  <head>
  </head>
  <body>
    <!--<link rel="stylesheet" href="styles.css">-->
    <style>

        h1{
font-weight: 200;
letter-spacing: 0.4px;
font-family: "Raleway", Sans-serif;
color: #232323;
}
        h2{
font-weight: 150;
letter-spacing: 0.4px;
font-family: "Raleway", Sans-serif;
color: #232323;
}

        p{
font-weight: 50;
letter-spacing: 0.4px;
font-family: "Raleway", Sans-serif;
color: #232323;
}
    </style>
  </body>
</html>

```



```

    .button {
background-color: white;
border: 2px solid #232323;
color: black;
padding: 16px 32px;
text-align: center;
text-decoration: none;
display: inline-block;
font-size: 16px;
margin: 4px 2px;
-webkit-transition-duration: 0.4s;
transition-duration: 0.4s;
cursor: pointer;
}

.alertsucess {
padding: 20px;
background-color: #4CAF50;
color: white;
}

.alerterror {
padding: 20px;
background-color: red;
color: white;
}

.closebtn {
margin-left: 15px;
color: white;
font-weight: bold;
float: right;
font-size: 22px;
line-height: 20px;
cursor: pointer;
transition: 0.3s;
}

.closebtn:hover {
color: black;
}

table, th, td {
border: 1px solid black;
}

</style>

<table style="width:100%">
<th style="width:30%">

```

```

<h1> Contactos </h1>

<h1>Lista</h1>

<h2> Introduzir Contactos </h2>

<section id=section1>

<form id= "um">

    <p><label for="nome">Nome:</label><br>
        <input type= "texto" id="nomeid" name="nome">
    </p>

    <p><label for="telefone">Telefone:</label><br>
        <input type="texto" id="telefoneid"
name="telefone">
    </p>

    <p><label for="tipotelefone">Tipo Telefone:</label><br>
        <input type="texto" id="tipotelefoneid"
name="tipotelefone">
    </p>

    <p><label for="email">Email:</label><br>
        <input type="texto" id="emailid" name="email">
    </p>

    <p><label for="tipoemail">Tipo Email:</label><br>
        <input type="texto" id="tipoemailid"
name="tipoemail">
    </p>

    <p><label for="enderenco">Ederenço:</label><br>
        <input type="texto" id="enderecoid"
name="endereco">
    </p>

    <p><label for="tipoenderenco">Tipo Endereço:</label><br>
        <input type="texto" id="tipoenderecoid"
name="tipoendereco">
    </p>

    <p><label for="grupos">Grupos:</label><br>
        <input type="texto" id="gruposid" name="grupos">
    </p>

```

```

        <p>
            <button Class="button" type="button" id="botao"
style="vertical-align:middle" onclick="insertcontacto() "
><span>Guardar</span></button>
            <!--<input type="button" id="botao" value = Guardar
onclick="insertcontacto()"> >!-->
        </p>

    </form>

</section>

    <div id="mensagensucesso" style="visibility: hidden;"
class="alertsucces">
        <span class="closebtn"
onclick="this.parentElement.style.display='none';">&times;</span>
        <strong>Sucesso</strong> Dados inseridos com sucesso!
    </div>

    <div id="mensagemerro" style="visibility: hidden;"
class="alerterror">
        <span class="closebtn">&times;</span>

        <strong>ERRO</strong> Um ou mais campos estão
vazios!
    </div>

    <section id=section2>
        <form id= "m">

            <p><label for="contacto"></label><br>
                <output type= "texto" id="apresentarid"
name="outapresentar">
            </p>

            <p><input class="button" id="botao1" value ="Ver
Lista de Contactos" onclick="vercontacto()"></p>

            <script>

```

```

</script>

</form>
</th>
<th>
    <table cellpadding="1" cellspacing="1" border="0"
bgcolor="white" width="60%" align="center" style="visibility: hidden;">
        <thead>
        </thead>
        <tbody id="tabela"></tbody>
    </table>

</section>
</th>

</table>

<script>

    function insertcontato(){

        var xhttp = new XMLHttpRequest();
        xhttp.open("POST",
"http://localhost:1997/registo", true);
        xhttp.setRequestHeader('Content-
type','application/json; charset=utf-8');
        xhttp.onreadystatechange =
function(){
            if (this.readyState == 4 &&
this.status == 200) {

document.getElementById("mensagensucesso").style.visibility='visible';
            }
        };

        var dados = {}
        dados.nome = um.nomeid.value;
        dados.telefone =
um.telefoneid.value;

        dados.tipotelefone =
um.tipotelefoneid.value;

        dados.email= um.emailid.value;
        dados.tipoemail =
um.tipoemailid.value;
    }
}

```

```

                                dados.endereco =
um.enderecoid.value;

                                dados.tipoendereco =
um.tipoenderecoid.value;

                                dados.grupos =
um.gruposid.value;

                                if(dados.nome == null
|| dados.telefone == 0 || dados.tipotelefone == 0 || dados.email == 0 ||
dados.tipoemail == 0 || dados.endereco == 0 || dados.tipoendereco == 0
|| dados.grupos == 0){

document.getElementById("mensagemerro").style.visibility='visible';

                                var close =
document.getElementsByClassName("closebtn");
                                var i;
                                for (i = 0; i < close.length; i++) {
                                    close[i].onclick = function(){
                                        var div = this.parentElement;
                                        div.style.opacity = "0";
                                        setTimeout(function(){ div.style.display
= "none"; }, 600);
                                    }
                                }

                                }else{
                                    var dadosemJSON =
JSON.stringify(dados)

                                    xhttp.send(dadosemJSON);
                                    limparcampos();

                                }

                                }

function vercontacto(){
    console.log("so far so good");
    var xhttp = new XMLHttpRequest();
    xhttp.onreadystatechange = function(){
        if (this.readyState == 4 && this.status == 200) {
            var arrayJSON = JSON.parse(this.responseText);
            lines = "<table> <tr><th>Nome</th><th>Telefone</th><th>Tipo
Telefone</th><th>Email</th><th>Tipo Email</th><th>Endereço</th><th>Tipo
Endereço</th><th>Grupos</th></tr>";
            for (i = 0; i < arrayJSON.length; i++) {

                lines = lines + "<tr>";

```

```

        lines = lines + "<td>" + arrayJSON[i].nome + "</td>";
        lines = lines + "<td>" + arrayJSON[i].telefone + "</td>";
        lines = lines + "<td>" + arrayJSON[i].tipotelefone +
"</td>";

        lines = lines + "<td>" + arrayJSON[i].email + "</td>";
        lines = lines + "<td>" + arrayJSON[i].tipoemail + "</td>";
        lines = lines + "<td>" + arrayJSON[i].endereco + "</td>";
        lines = lines + "<td>" + arrayJSON[i].tipoendereco +
"</td>";

        lines = lines + "<td>" + arrayJSON[i].grupos + "</td>";

        lines = lines + "</tr>";
    }
    lines = lines + "</table>";
    document.getElementById("tabela").innerHTML = lines;

document.getElementById("tabela").style.visibility='visible';

    }
    };

    xhttp.open("GET",
"http://localhost:1997/contacto", true);

    u = m.apresentarid.value

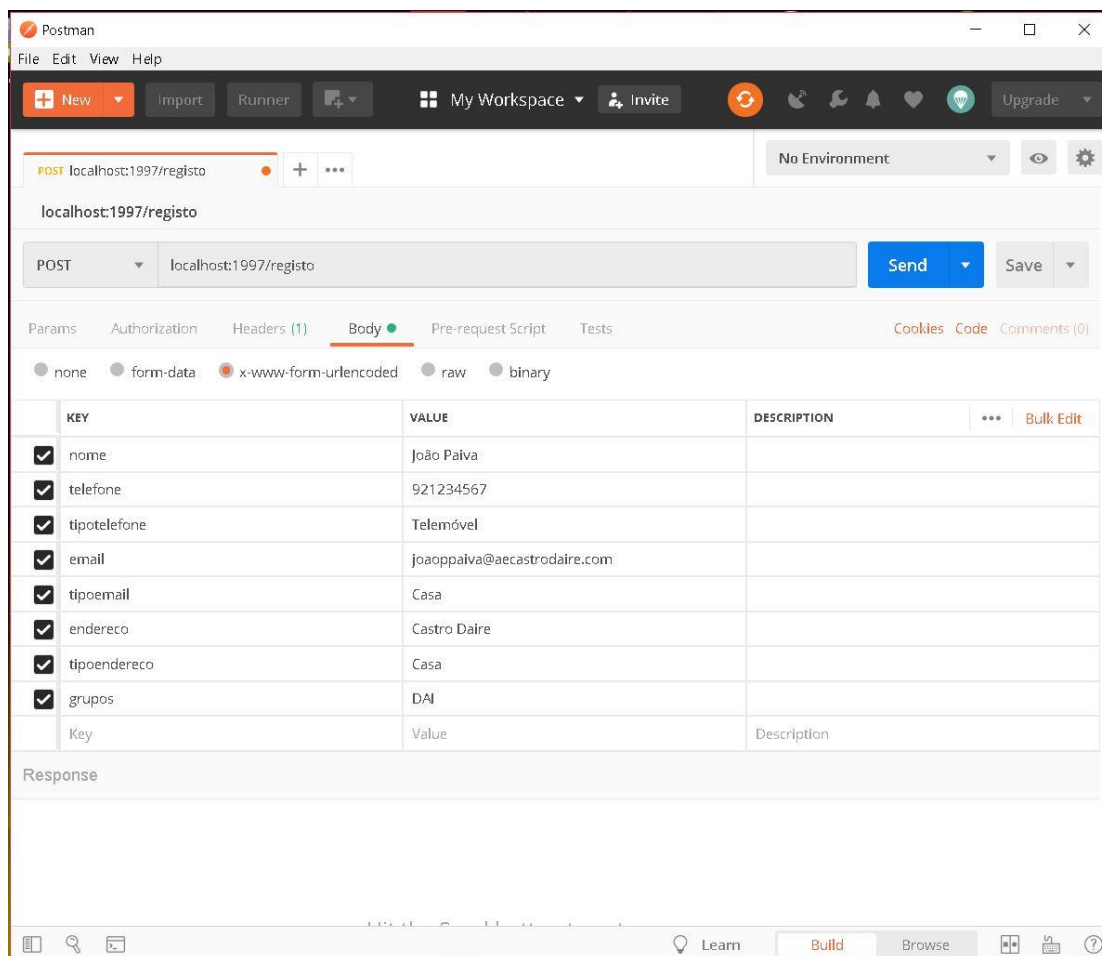
    xhttp.send(u)
}

function limparcampos(){
    um.nomeid.value = '';
    um.telefoneid.value = '';
    um.tipotelefoneid.value = '';
    um.emailid.value = '';
    um.tipoemailid.value = '';
    um.enderecoid.value = '';
    um.tipoenderecoid.value = '';
    um.gruposid.value = '';
}

</script>
</body>
</html>

```

## 10. Utilização do Postman



## 11. Interface

Contactos

Lista

Introduzir Contactos

Nome:

Telefone:

Tipo Telefone:

Email:

Tipo Email:

Endere o:

Tipo Endere o:

Grupos:

Guardar

Name	Telefone	Tipo Telefone	Email	Tipo Email	Endere�o	Tipo Endere�o	Grupos
Jo�o Paiva	925995255	Telem�vel	joaoppaiva@aecastrodai.re.com	Casa	Castro Daire	Casa	DAI
Wesley Santos	921234567	Telem�vel	wesley@hotmail.com	Casa	S�o Paulo	Casa	DAI

Ver Lista de Contactos

## 12. Mensagem de Erro

**ERRO** Um ou mais campos est o vazios!



## 13. Mensagem de Sucesso

**Sucesso** Dados inseridos com sucesso!





---

## INTEGRAÇÃO DO SERVIDOR NODEJS E DA RESPECTIVA INTERFACE NA CLOUD

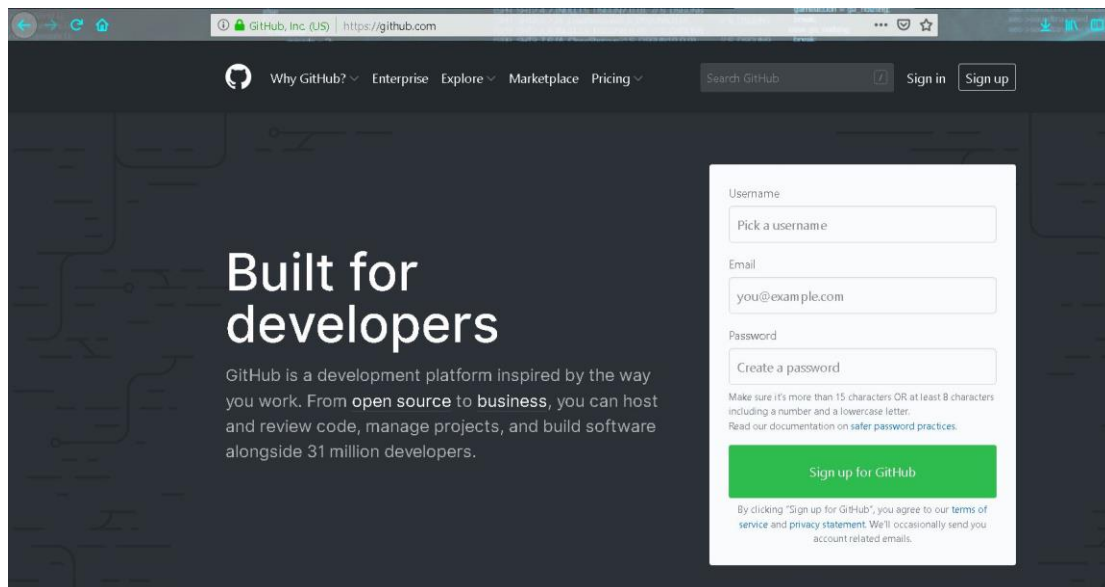
---

Serviço usado: Google Cloud

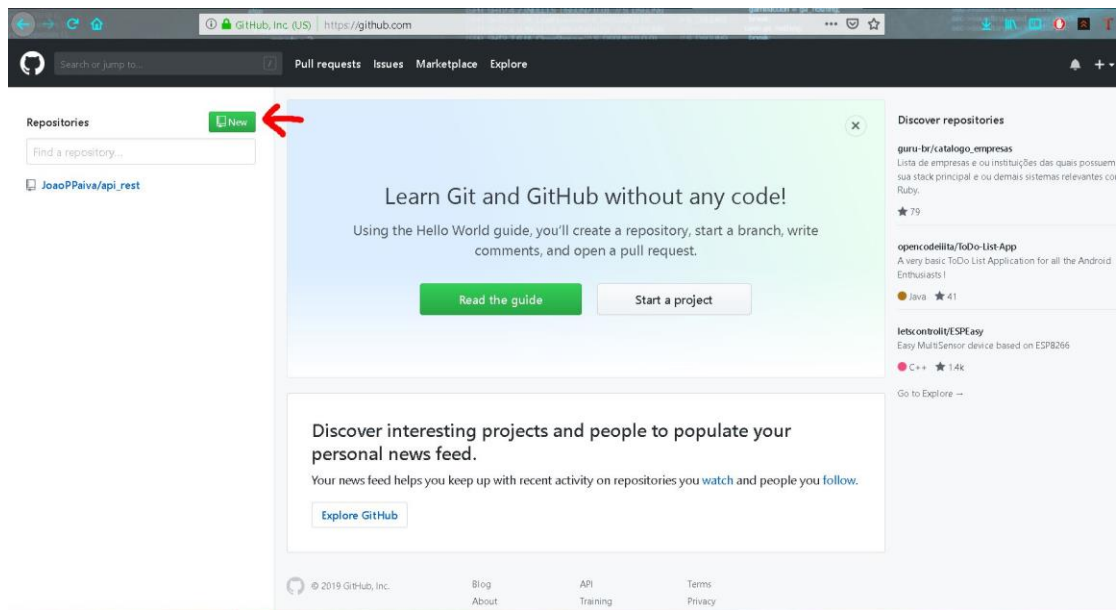
Requisitos necessários para esta implementação específica\*

- Conta no github
- Conta no Google Cloud platform(inclui especificar as informações de pagamento para ativar a avaliação gratuita

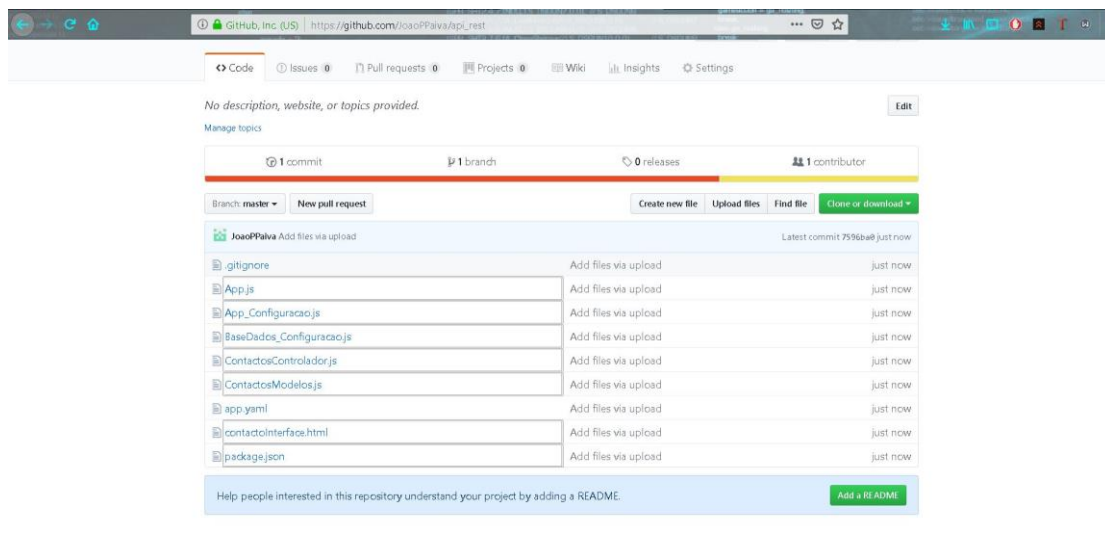
1º Passo: Github



Criamos um repositório aonde irão ser colocados os ficheiros necessários para a API

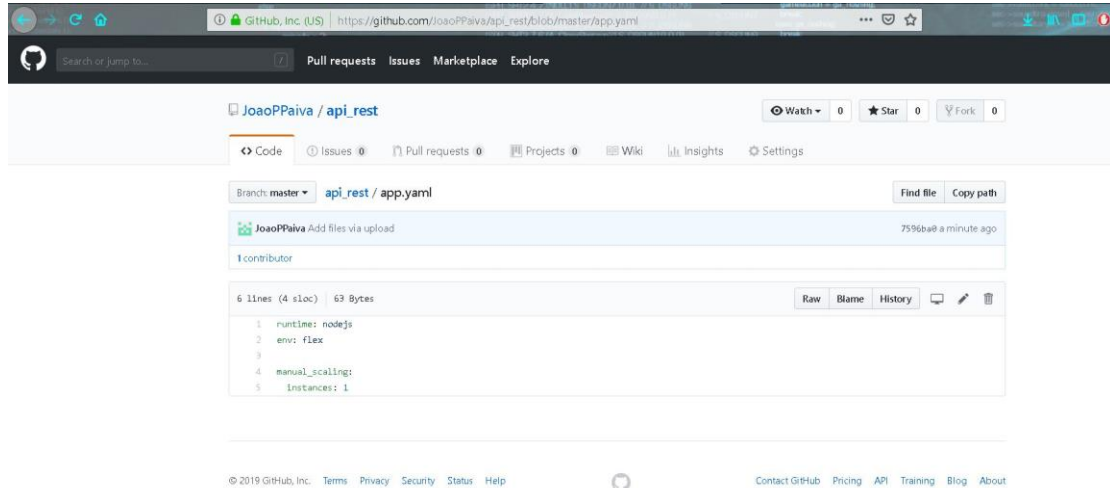


Os ficheiros que adicionámos no repositório foram os seguintes



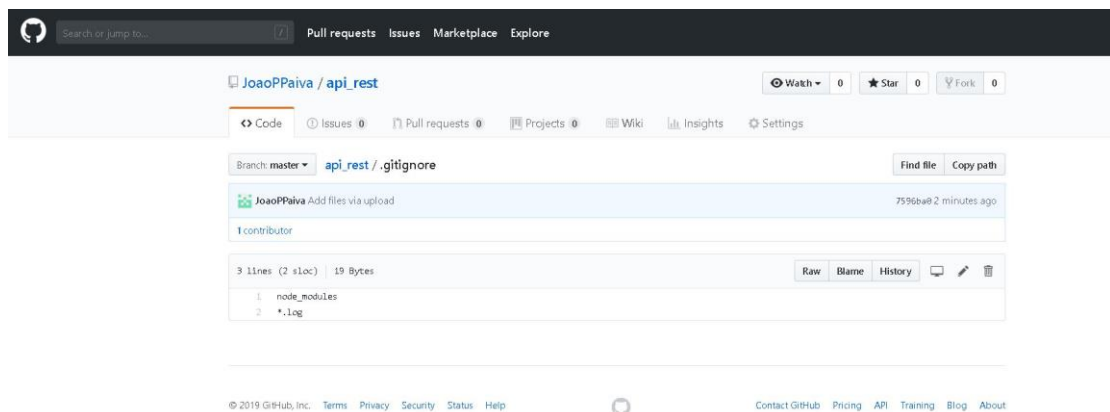
Dois ficheiros que não adicionámos, foram criados de raiz são o app.yaml e o .gitignore

- app.yaml



O ficheiro é necessário para que a ferramenta App engine do google cloud compreenda que estamos a usar nodejs

- .gitignore



O .gitignore é necessário neste caso para ignorar que os ficheiros que necessitem de módulos não procurem no repositório pois o GitHub já oferece os que necessitamos.

- package.json

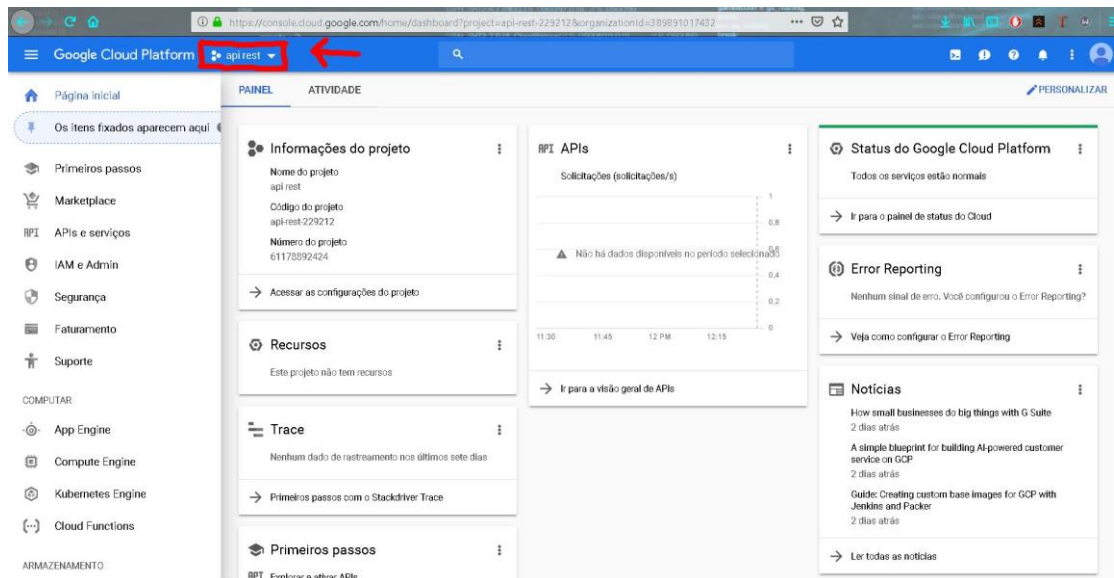
Efetuámos uma alteração no package.json para que o App Engine do Google Cloud platform executasse o ficheiro principal

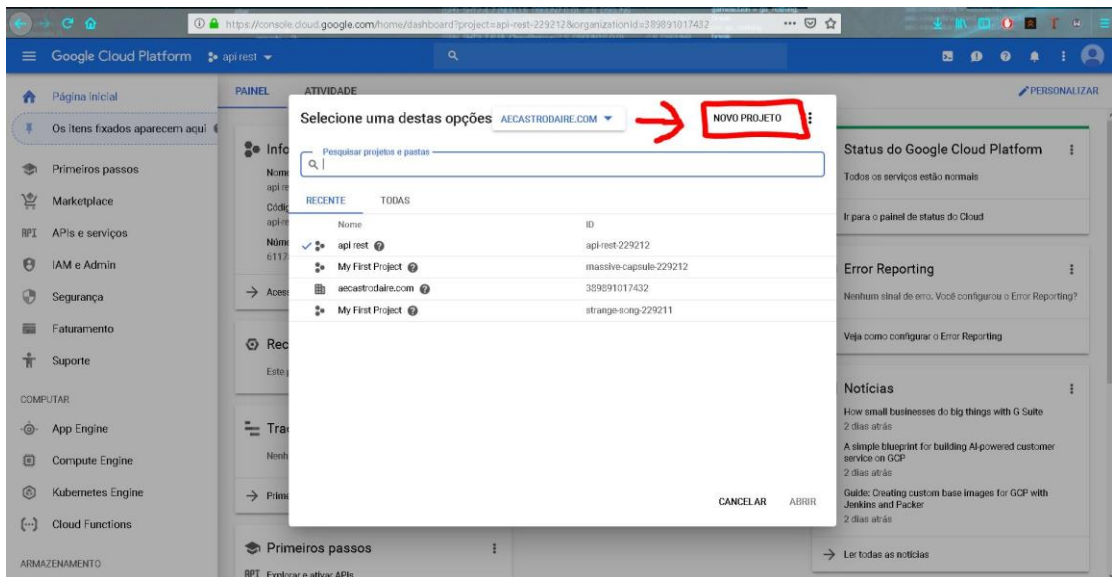
```

1 {
2   "name": "listcontactos",
3   "version": "1.0.0",
4   "description": "",
5   "main": "App.js",
6   "dependencies": {
7     "body-parser": "^1.18.3",
8     "express": "^4.16.4",
9     "mongoose": "^5.4.2",
10    "node": "^11.6.0",
11    "nodemon": "^1.18.9"
12  },
13  "devDependencies": {},
14  "scripts": {
15    "start": "echo 'Error: no test specified' && exit 1",
16    "test": "node App.js"
17  },
18  "author": "Joao Paiva",
19  "license": "ISC"
20 }
  
```

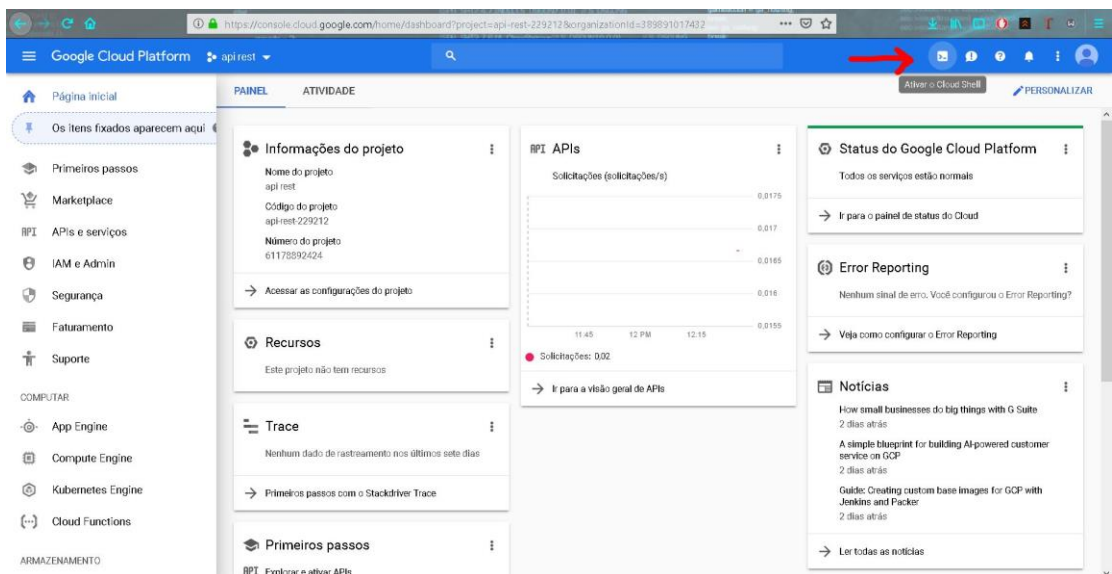
## 2º Passo: Google Cloud

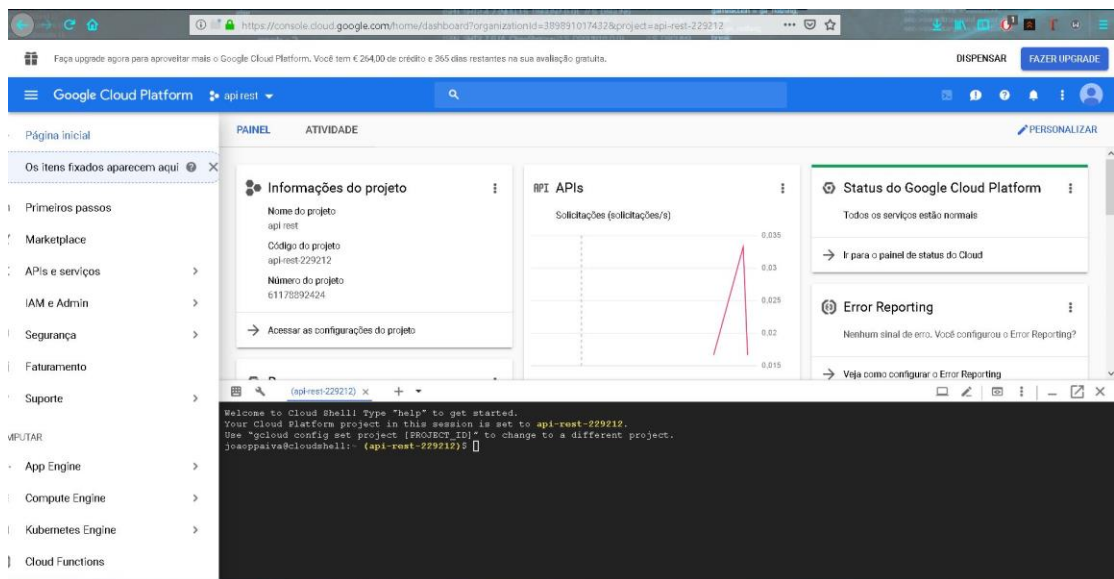
Criámos um projeto no google cloud, já possuíamos uma conta google





Depois do projeto criado e de um nome atribuído usamos o google cloud Shell para clonar o repositório do github e fazer o deploy da aplicação no App Engine





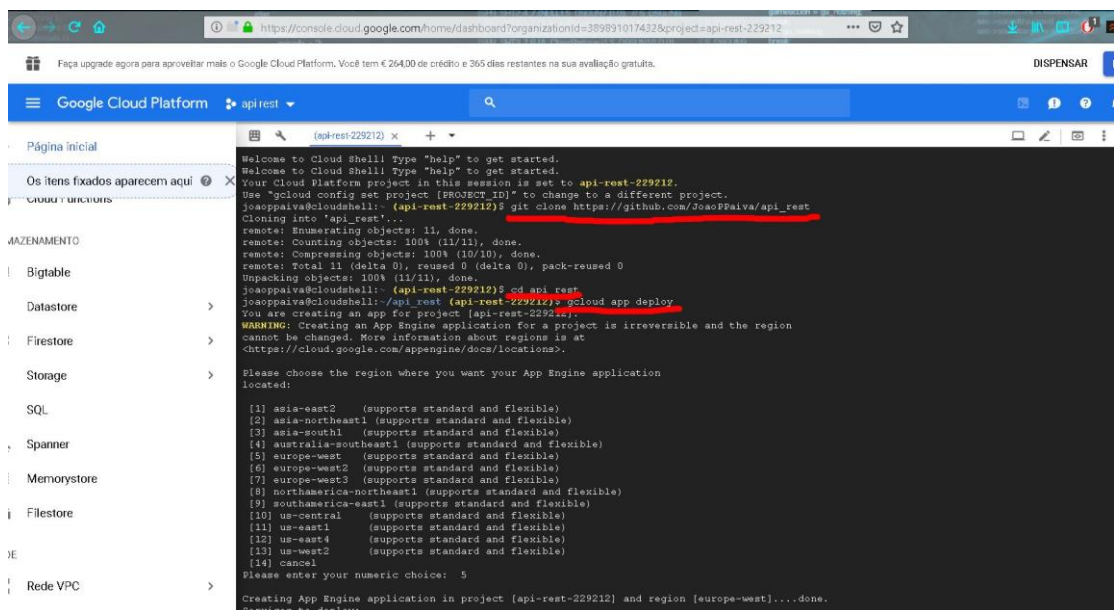
Fizemos um clone do repositório do github:

*git clone* [https://github.com/carv3l/api\\_rest](https://github.com/carv3l/api_rest)

Dentro do repositório fizemos o deploy para a app engine

*cd api\_rest*

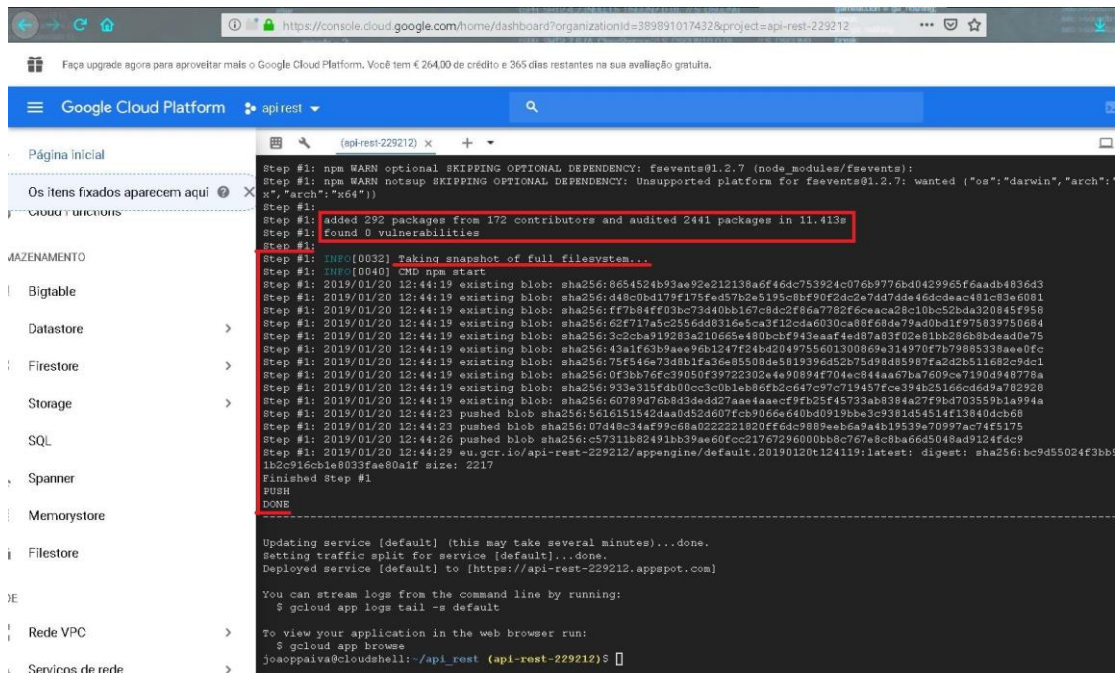
*gcloud app deploy*



Durante este processo deu um erro do qual verificámos que não tínhamos a parte do pagamento definida para poder ativar a avaliação grátis

Depois a App Engine instalou umas dependências necessárias mais concretamente o para poder publicar a API e fez um snapshot do sistema



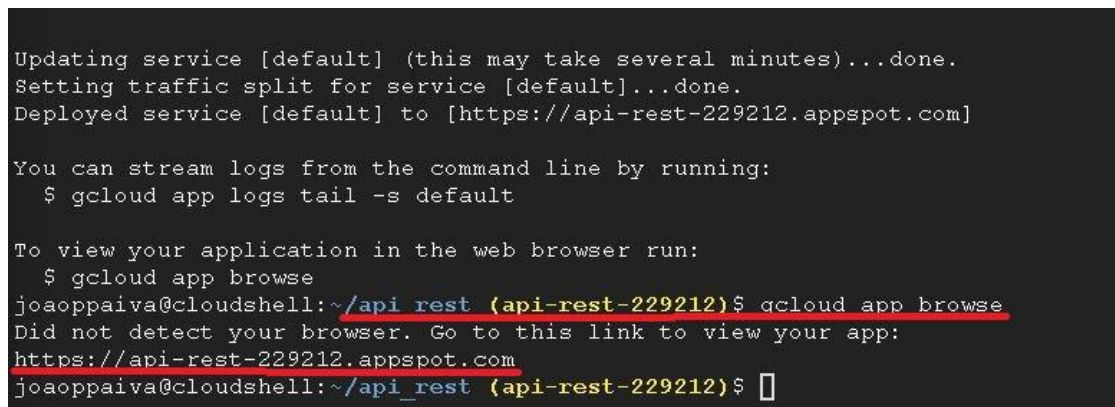


Depois de esperarmos uns bons minutos estamos prontos para publicar a aplicação.

Corremos o comando

***gcloud app browse***

Aacemos à API com o link que a App Engine providencia.



### 3º Passo: Pós modificações

Foi necessário modificar o url de GET e POST de localhost para o url providenciado pela Google Cloud

```
190         function insertcontacto(){
191
192             var xhttp = new XMLHttpRequest();
193             xhttp.open("POST", "https://api-rest-229212.appspot.com/contacto", true);
194             xhttp.setRequestHeader('Content-type', 'application/json; charset=utf-8');
195             xhttp.onreadystatechange = function(){
196                 if (this.readyState == 4 && this.status == 200) {
197
198                     document.getElementById("mensagensucesso").style.visibility='visible';
199                 }
200             };
201
202
203
204
205             xhttp.open("GET", "https://api-rest-229212.appspot.com/registo", true);
206
207             u = m.apresentarid.value
208
209             xhttp.send(u)
210         }
211
```

Substituímos a interface da API pelo URL da Interface no ficheiro App.js que é por onde a App Engine começa a executar.

```
41 lines (32 sloc) | 1.1 KB
1  var app = require('./App_Configuracao');
2  var Base = require('./BaseDados_Configuracao');
3  var contacto = require('./ContactosModelos')
4  var contactocontrolador = require('./ContactosControlador');
5
6  app.get('/', function(request, response){
7      response.sendFile('contactoInterface.html');
8  });
9
```