Aplicações para a Internet II

Node.js + express + React.js + PostGres

2021/2022

<Criação do Projeto/>

Criar a pasta do nosso projeto:

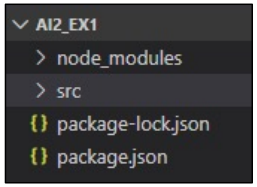
mkdir ai2_ex1

Para criar o arquivo package.json usar o comando seguinte:

npm init --yes

```
{
  "name": "ai2_ex1",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Após da instalação do primeiro pacote (por exemplo, o pacote express, ver a seguinte seção Express) será adicionado o ficheiro **package-lock.json e a pasta node_modules**



<Express/>

- O Express e o Node.js deram ao JavaScript uma nova funcionalidade de criar software no lado do servidor. A combinação dos dois permite a construção de uma aplicação completamente baseada em JavaScript.
- As aplicações são desenvolvidas do lado do servidor com Node.js e, em seguida, publicadas com o Express, capaz de desenvolver uma camada na estrutura interna e publicar funções necessárias para construir um site ou uma aplicação.

- Instalação da *framework* do node.js **Express**, sendo um dos mais populares:

  sudo npm install express

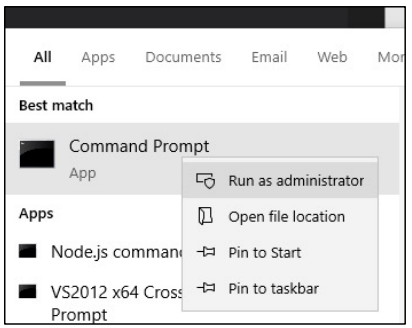
- O comando sudo é usado em sistemas Linux. O resultado será idêntico ao seguinte:

```
+ express@4.17.1
added 50 packages from 37 contributors and audited 126 packages in 2.077s
found 0 vulnerabilities
```

- Em Windows é necessário executar o mesmo comando sem o sudo:

  npm install express

  Atenção que deve ser executado em Linha de Comandos como Administrador.

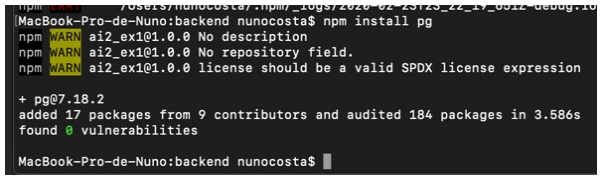


<PostGres/>

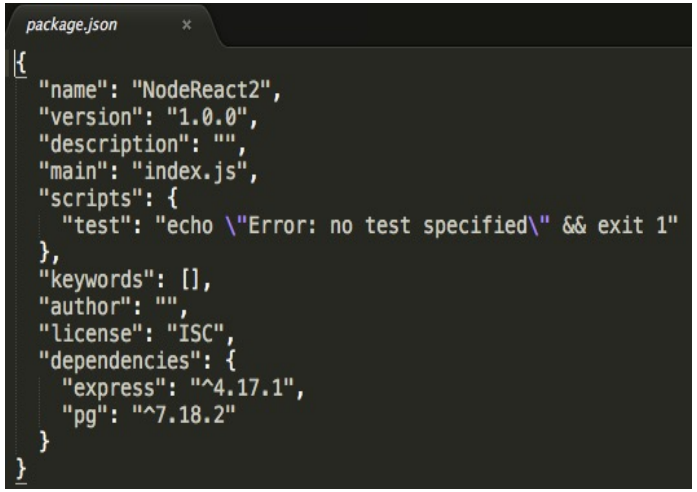
- O sistema de gestão de base de dados que vamos usar é o **Postgres**. Para a sua instalação devemos usar o seguinte comando:

  sudo npm install --save pg

- A execução do comando vai gerar um resultado idêntico ao seguinte:



- Após as instalações anteriores, é de notar que o ficheiro package.json está diferente (ver a Figura 6 - Package.json após instalações):



<Raiz do projeto/>

- Qualquer aplicação web tem um local de início. Para isso, e tendo em conta as boas práticas da programação, vamos criar uma pasta ***src*** e criar o ficheiro **App.js**:

```
const express = require('express');
const app = express();
//Configurações
app.set('port', process.env.PORT || 3000);
//Middlewares
app.use(express.json());
//Rotas
app.use('/teste',(req,res)=>{
    res.send("Rota TESTE.");
  });
app.use('/',(req,res)=>{
  res.send("Hello World");
});

app.listen(app.get('port'),()=>{
    console.log("Start server on port "+app.get('port'))
})
```

Neste ficheiro, podemos verificar que temos:
Importação da framework Express;
Configuração da Porta, pela qual a aplicação vai ser acedida;
Definidas duas rotas:
- Rota Raiz (‘/’);
- Rota Teste (‘/teste’).

< Nodemon/ >

- Executar:
- **sudo npm install --g nodemon**

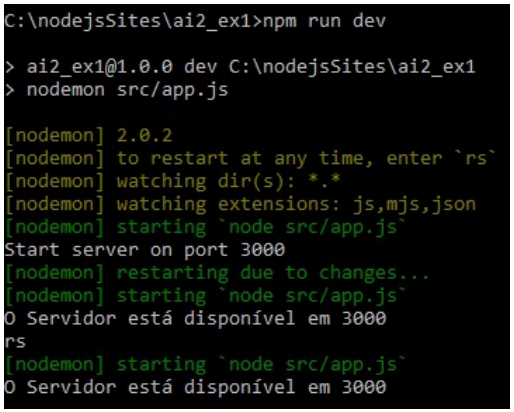
Alterar o ficheiro **package.json** de forma a incluir em script a seguinte linha de código:

```
{
  "name": "ai2_ex1",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "dev":"nodemon src/app.js"
  },
  "keywords": [],
  "author": "Nuno Miguel Martins da Costa",
  "license": "DI ESTGV IPV",
  "dependencies": {
    "express": "^4.17.1"
  }
}
```

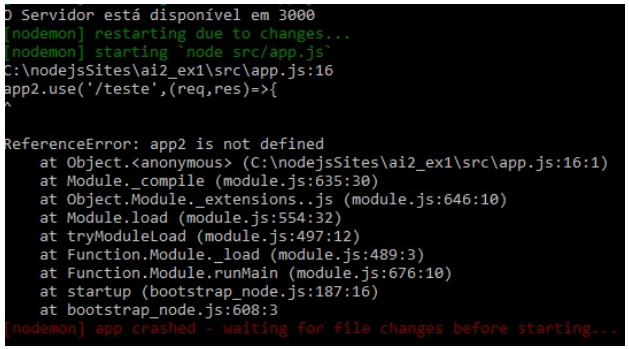
< Nodemon/ >

- E por fim, executar:
  - **npm run dev**

- Após a execução deste comando, o servidor vai estar disponível no porto 3000 de um browser. Sempre que haja alterações nos ficheiros (ou usando o comando rs), o servidor faz automaticamente um reinicio do serviço



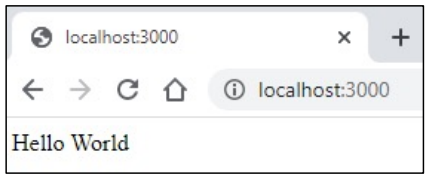
- É nesta janela que serão visualizados, em tempo real, os erros que possam existir na aplicação.



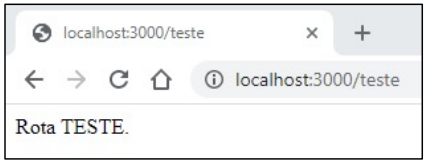
< Controladores e Rotas/ >

- O papel de uma rota no Express é o de decidir o que fazer quando é solicitado algum pedido.

http://localhost:3000/

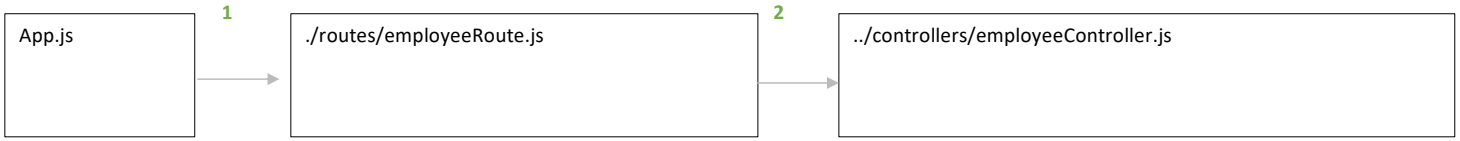


http://localhost:3000/teste



< Controladores e Rotas/ >

- Seguindo o padrão MVC vamos criar duas pastas em ***src*** com os nomes ***routes*** e ***controllers***.



- Na pasta ***controllers*** - criar o ficheiro **employeeController.js** com o seguinte código:

```
const controller = {}
controller.test = (req,res) => {
  const data = {
    name: "Nuno Costa",
    age: 42,
    city: 'Viseu'
  }
  console.log("Envio de dados do Controlador EMPLOYEE.");
  res.json(data);
};
module.exports = controller;
```

< Controladores e Rotas/ >

- Ficheiro **employeeRoute.js** na pasta **routes** com o seguinte código:

```
const express = require('express');
const router = express.Router();
//importer os controladores [2]
const employeeController = require('../controllers/employeeController')
router.get('/test',employeeController.test);

router.get('/save', (req, res) => {
  res.json({status: 'Employeed Saved'});
});
module.exports = router;
```

- Alterar o ficheiro **App.js** de forma que sejam importadas as rotas do ***employee***:

```
// importação de rotas [1]
const employeeRouters = require('./routes/employeeRoute.js')
//Rota
app.use('/employee',employeeRouters)
```

< Sequelize/ >

- O ***Sequelize*** é uma *framework* de ORM (*Object-Relational Mapping*) preparada para o Node.js. Suporta os Sistemas de Gestão de Base de Dados (SGBD) Postgres, MySQL, MariaDB, SQLite e Microsoft SQL Server.
- Instalação através do seguinte comando:

  npm install --save sequelize

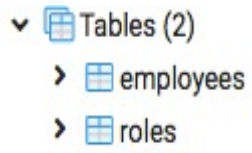
- No fim da sua execução são mostradas informações de versão e do número de pacotes adicionados ao projeto.

```
+ sequelize@5.21.3
added 21 packages from 91 contributors and audited 148 packages in 2.34s
found 0 vulnerabilities
```

< Models (Modelos)/ >

- Segundo o padrão MVC o ***Model*** (modelo) corresponde aos dados da aplicação, às regras de negócios, à lógica e às funções. É esta parte da estrutura MVC que gere o processo de negócio, respondendo a pedidos do controlador e apresenta os resultados numa ***View*** (vista).
- Criar dentro da pasta **src** uma pasta chamada **model**. Será dentro desta pasta que vamos criar a ligação à base de dados e os vários modelos da nossa aplicação.
- **Atenção:** A partir deste ponto é necessária uma ligação válida a uma das bases de dados que o ***Sequelize*** suporta. Neste exemplo, foi necessário instalar o pacote node para o postgres:

  npm install postgres



< Models (Modelos)/ >

- Criar o ficheiro **database.js** em **src/model/** com o código seguinte:

```
var Sequelize = require('sequelize');
const sequelize = new Sequelize(
  'teste',
  'postgres',
  'postgres',
  {
    host: 'localhost',
    port: '5432',
    dialect: 'postgres'
  }
);
module.exports = sequelize;
```

< Models (Modelos)/ >

- Criar outro ficheiro **Role.js** na mesma pasta com o código seguinte:

```
var Sequelize = require('sequelize');
var sequelize = require('./database');
var Role = sequelize.define('role', {
  role: Sequelize.STRING
},
{
      timestamps: false,
});
module.exports = Role
```

< Models (Modelos) />

- Criar outro ficheiro **Employee.js** na mesma pasta com o código seguinte:

```
var Sequelize = require('sequelize');
var sequelize = require('./database');
// importa o modelo - chave forasteira roleID
var Role = require('./Role');

var Employee = sequelize.define('employee', {
  id: {
    type: Sequelize.INTEGER,
    primaryKey: true,
    autoIncrement: true,
  },
  name: Sequelize.STRING,
  email: Sequelize.STRING,
  address: Sequelize.STRING,
  phone: Sequelize.BIGINT,
  roleId: {
    type: Sequelize.INTEGER,
    // referência a outro modelo
    references: {
      model: Role,
      key: 'id'
          }
        }
                                    },
{
        timestamps: false,
});
Employee.belongsTo(Role)
module.exports = Employee
```

< Models (Modelos)/ >

- Adicionar o código seguinte no ficheiro **src/controllers/employeeController.js**:

```
var Employee = require('../model/Employee');

var Role = require('../model/Role');

var sequelize = require('../model/database');

 const controllers = {}

 sequelize.sync()

. . .

controllers.testdata = async ( req, res) => {

  const response = await sequelize.sync().then(function() {

    /** APAGAR após a primeira EXECUÇÃO

    //Cria Role

    Role.create({

      role:  'Admin'

    });

    // Cria employee

    Employee.create({

      name: 'Nuno Costa',

      email:  'ncosta@estgv.ipv.pt',

      address: 'Campus Politécnico, Viseu, Portugal',

      phone: '232480533',

      roleId:1

    });

Employee.create({
      name: 'Sousa Marques',
      email:  'marquesousa@nop.pt',
      address: 'Rua da Missa, Lisboa, Portugal',
      phone: '221485543',
      roleId:1
    });
    */
     const data =  Employee.findAll()
     return data;
  })
  .catch(err => {
    return err;
  });
  res.json(response)
}

controllers.list = async ( req, res) => {

    const data = await Employee.findAll();
    res.json(data)

}
                                    . . .
```

< Models (Modelos)/ >

- Adicionar o código seguinte no ficheiro **src/routes/employeeRoute.js**

```
...

router.get('/testdata',employeeController.testdata );

router.get('/list',employeeController.list );

...
```