5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

A – Criar uma aplicação com Node.js e ExpressJS, com a view implementando o AngularJS.

Neste projeto trabalharemos na camada de visualização. Definiremos um novo projeto baseado no Node.js com ExpressJS, acessando um webservice capaz de acessar dados do banco de dados MongoDB. A interação com o usuário será realizada através de uma única página, contemplando o recurso SPA (Single Page App) com AngularJS. Este projeto , por usar o Mongo, o Express, o Angular e o Node, será considerado uma aplicação **MEAN STACK**.

1. Na pasta **ProjetoEventos**, criar uma nova pasta chamada **Mean_AngularJS**.
2. Usando o express, criar um novo projeto chamado **appAngularJS**. Usar a sequencia de comandos:

```
express appAngularJS --ejs
cd appAngularJS
npm install
npm install body-parser --save
npm install express-load --save
```

3. Abrir o **VSCode** na pasta **appAngularJS**.
4. Criar as pasta **controllers**.
5. Apagar o conteúdo das pastas **routes** e **views**.
6. Na pasta **routes**, criar o arquivo **eventos.js**:

```
module.exports = function (app) {
    var evento = app.controllers.eventos;
    app.get('/', evento.index);
};
```

7. Na pasta **controllers** criar o arquivo **eventos.js**:

```
module.exports = function (app) {
    var EventosController = {
        index: function (req, res) {
```

```
                res.render('eventos/index');
            }
        };
        return EventosController;
    };
```

8. Atualizar o arquivo **app.js**:

```
var express = require('express');
var load = require('express-load');
app = express();

app.set('views', __dirname + '/views');
app.set('view engine', 'ejs');
app.use(express.static(__dirname + '/public'));

load('controllers')
  .then('routes')
  .into(app);

app.listen(3000, function () {
  console.log("Aplicação no ar.");
});
```

9. Criar a view: **views/eventos/index.ejs**:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel='stylesheet' href='/stylesheets/style.css' />
    <title>Aplicação AngularJS</title>
</head>
<body>
    <h1>Cadastro e Consulta de Eventos</h1>


</body>
</html>
```

10. Testar a aplicação até este ponto, para checar se está tudo bem. Executar no prompt de comandos: **node app.js**, e no browser, **localhost:3000**.

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

11. Vamos implementar as funcionalidades do AngularJS. Na pasta **public/javascripts**, incluir a referencia à biblioteca do angularJS (**arquivo angular.js**). Fazer o download desta biblioteca no link: https://angularjs.org/ . Incluir esta referencia no arquivo **index.ejs**:

```
<!DOCTYPE html>
<html ng-app>
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel='stylesheet' href='/stylesheets/style.css' />
    <title>Aplicação AngularJS</title>
</head>
<body>
    <h1>Cadastro e Consulta de Eventos</h1>


    <script src="/javascripts/angular.js"></script>
</body>
</html>
```

12. Esta aplicação utilizará o webservice desenvolvido no projeto anterior. Para que o webservice seja consumido por outra plataforma diferente daquela onde foi desenvolvido, é necessário habilitar o recurso “Cross-Origin”, conhecido como **CORS**. Antes de prosseguir com a aplicação, abrir o projeto **apiEventos**.
13. No prompt de comandos, executar o comando:

```
npm install cors --save
```

14. No arquivo **app.js** do webservice, realizar as seguintes alterações:

```
var express = require('express');
var load = require('express-load');
var cors = require('cors');

var app = express();
var bodyParser = require('body-parser');

app.use(cors());
```

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

```
app.use(function (req, res, next) {
  res.header("Access-Control-Allow-Origin", "*");
  res.header("Access-Control-Allow-Headers", "Origin, X-Requested-With,
Content-Type, Accept");
  next();
});
```

15. O webservice deve ser reiniciado.
16. Retornando ao nosso projeto AngularJS, vamos agora incluir uma tabela na nossa view. O objetivo é usar o webservice desenvolvido no projeto anterior, e já atualizado com o módulo **cors**, através do módulo **http** do **AngularJS**. Alterar o arquivo **index.ejs**, incluindo as implementações do módulo e do controller do Angular:

```
<!DOCTYPE html>
<html ng-app="appAngular">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel='stylesheet' href='/stylesheets/style.css' />
    <style>
        table,
        tr,
        td,
        th {
            border: 1px solid #ccc;
            padding: 10px;
            margin: 10px;
        }
    </style>
    <title>Aplicação AngularJS</title>
</head>

<body ng-controller="Principal as ctl">
    <h1>Cadastro e Consulta de Eventos</h1>
    <h2>Lista de eventos</h2>

    <table>
```

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

```
        <thead>

          <tr>
              <th>Descrição</th>
              <th>Data</th>
              <th>Preço</th>
          </tr>
        </thead>

          <tbody>
              <tr ng-repeat="evento in ctl.items">
                  <td>{{evento.descricao}}</td>
                  <td>{{evento.data}}</td>
                  <td>{{evento.preco}}</td>
              </tr>
          </tbody>
    </table>
```

```
    <script src="/javascripts/angular.js"></script>
    <script type="text/javascript">

        angular.module("appAngular", [])
            .controller('Principal', ['$http', function ($http) {

                var self = this;

                self.items = [];

                var listarTodos = function () {
                    return $http.get('http://localhost:3200/eventos/')
                    .then(function (response) {
                        self.items = response.data;
                    }, function (error) {
                        alert('Erro reportado: ' + error);
                    });
                };

                listarTodos();
            }]);
    </script>
</body>

</html>
```

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

17. Ao executar a aplicação, podemos visualizar a lista de eventos na página. A chamada foi executada na carga da página, mas sua execução foi realizada de forma assíncrona.
18. O próximo passo é definir um formulário que permita a inclusão de um novo evento. O formulário será desenvolvido na mesma página que a listagem, caracterizando assim uma aplicação SPA. Adicionar o formulário e a função javascript, conforme modelo a seguir:

```
<!DOCTYPE html>
<html ng-app="appAngular">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel='stylesheet' href='/stylesheets/style.css' />
    <style>
        table,
        tr,
        td,
        th {
            border: 1px solid #ccc;
            padding: 10px;
            margin: 10px;
        }
    </style>
    <title>Aplicação AngularJS</title>
</head>

<body ng-controller="Principal as ctl">
    <h1>Cadastro e Consulta de Eventos</h1>
    <h2>Lista de eventos</h2>

    <table>
        <thead>
            <tr>
                <th>Descrição</th>
                <th>Data</th>
                <th>Preço</th>
            </tr>
        </thead>
        <tbody>
```

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

```
            <tr ng-repeat="evento in ctl.items">
                <td>{{evento.descricao}}</td>
                <td>{{evento.data | date:'dd/MM/yyyy'}}</td>
                <td>{{evento.preco | currency}}</td>
            </tr>
        </tbody>
    </table>
```

```
    <h2>Inclusão de um novo evento</h2>
    <div>
        <form name="incluirForm" ng-submit="ctl.adicionar()">
            Descriçãodo evento:<br/>
            <input type="text" ng-model="ctl.novoEvento.descricao" />
            <br/>Data (dd/mm/yyyy):<br />
            <input type="date" ng-model="ctl.novoEvento.data" />
            <br/>Preço:<br />
            <input type="text" ng-model="ctl.novoEvento.preco" />
            <br />
            <input type="submit" value="Adicionar" />
        </form>
    </div>
```

```
    <script src="/javascripts/angular.js"></script>
    <script type="text/javascript">

        angular.module("appAngular", [])
            .controller('Principal', ['$http', function ($http) {

                var self = this;

                self.items = [];
                self.novoEvento = {};

                var listarTodos = function () {
                    return $http.get('http://localhost:3200/eventos/')
                    .then(function (response) {
                        self.items = response.data;
                    }, function (error) {
                        alert('Erro reportado: ' + error);
                    });
                };

                listarTodos();
```

5. Definindo uma aplicação com AngularJS - MEAN (Parte 1)

```
                self.adicionar = function () {

                    $http({
                        url: 'http://localhost:3200/eventos/',
                        method: 'POST',
                        data: self.novoEvento,
                        headers: { 'Content-Type': 'application/json' }
                    }).then(function (response) {
                            self.novoEvento = {};
                        }, function (error) {
                            alert('Erro reportado: ' + error);
                        }).then(listarTodos);

                };
            }]);
    </script>
</body>

</html>
```

19. Executar a aplicação e incluir alguns eventos. Observe que a lista é atualizada tão logo o evento seja incluído.
20. Para completar, vamos apresentar uma formatação mais adequada para a data e para a moeda. Para tanto, aplicaremos o conceito de filtros. Vamos alterar a tabela de valores para contemplar os filtros:

```
    <table>
        <thead>
            <tr>
                <th>Descrição</th>
                <th>Data</th>
                <th>Preço</th>
            </tr>
        </thead>
        <tbody>
            <tr ng-repeat="evento in ctl.items">
                <td>{{evento.descricao}}</td>
                <td>{{evento.data | date:'dd/MM/yyyy'}}</td>
                <td>{{evento.preco | currency }}</td>
            </tr>
        </tbody>
```

```
</table>
```

21. Visualizar.