COM222 DESENVOLVIMENTO DE SISTEMAS WEB

Aula 12: Node + Mongo

Node + Mongo

- Conteúdo
  - Criação de um projeto Express
  - Instalação e configuração de um banco MongoDB
  - Criação de uma API para acessar o MongoDB

MongoDB

- Introdução ao MongoDB
  - https://cs.lmu.edu/~ray/notes/mongodb/

Instalar/criar projeto

- Baixar arquivo
  - https://baldochi.unifei.edu.br/COM222/nodetest1.zip
- Criar pasta “node”
  - Desempacotar nodetest1.zip nesta pasta
- cd nodetest1
- npm install
  - Verificar se o servidor está rodando
    - npm start
  - No browser
    - localhost:3000

Instalar/criar projeto

- Instalar as dependências do mongo e do mongoose
  - npm install –S mongodb
  - npm install –S mongoose
- Mongoose serve para fazer ORM
  - Object-Relational Mapping
- Note que o projeto nodetest1 tem uma pasta chamada “data”
  - Essa pasta será usada para armazenar os dados do Mongo

Instalar/criar projeto

- Importante notar que o projeto nodetest1 foi gerado pelo Express (não vamos rodar agora, pois já foi feito)
  - npm install –g express-generator
  - express –e nodetest1

- Arquivo app.js é o arquivo mais importante gerado pelo Express
  - Cria variáveis e as associa a
    - Pacotes
    - Dependências
    - Rotas
    - ...

Criar o banco

- Dentro da pasta do projeto (nodetest1) deve haver uma pasta “data”
- Entrar na basta bin dentro da pasta de instalação do MongoDB e digitar
  - mongod --dbpath c:\Node\nodetest1\data\
  - Este comando iniciará o servidor do Mongo
- Abra outro prompt de comando para iniciar o cliente do MongoDB, novamente na pasta bin, e digite
  - mongo

Criar o banco

- MongoDB oferece suporte a JSON, assim vamos inserir dados no banco utilizando esse formato
- Utilizando o cliente do MongoDB, digite:

```
use nodetest1
info= [{ "username" : "Joao", "email" : "joao@gmail.com" },
{ "username" : "Pedro", "email" : "pedro@gmail.com" },
{ "username" : "Maria", "email" : "maria@gmail.com" } ]
db.usercollection.insert(info);
```

- Para consultar os dados inseridos:

```
db.usercollection.find().pretty()
```

Conectar o Node com MongoDB

- Arquivo db.js é responsável por cuidar da conexão
  - Utiliza ORM Mongoose

```
var mongoose = require('mongoose');
mongoose.connect('mongodb://localhost:27017/nodetest1');
```

- Em seguida, define a estrutura da coleção de usuários e exporta um objeto contendo informações de conexão e da estrutura do banco

```
var userSchema = new mongoose.Schema({
    username: String,
    email: String
}, { collection: 'usercollection' }
);

module.exports = { Mongoose: mongoose, UserSchema: userSchema }
```

Exibir dados do MongoDB

- Para definir rotas temos que editar o arquivo \nodetest1\routes\index.js
  - Vamos definir a rota /userlist para permitir a visualização da lista de usuários

```
/* GET Userlist page. */
router.get('/userlist', function(req, res) {
   var db = require("../db");
   var Users = db.Mongoose.model('usercollection', db.UserSchema, 'usercollection');
   Users.find({}).lean().exec(
      function (e, docs) {
         res.render('userlist', { "userlist": docs });
   });
});
```

Exibir dados do MongoDB


```
router.get('/userlist', function(req, res) {
    var db = require("../db");
    var Users = db.Mongoose.model('usercollection', db.UserSchema, 'usercollection');
    Users.find({}).lean().exec(
        function (e, docs) {
            res.render('userlist', { "userlist": docs });
    });
});
```

- O comando require está voltando uma pasta e carregando o conteúdo do arquivo db.js que, por sua vez, está carregando a conexão com o banco de dados (via Mongoose) e o esquema da coleção de usuários
- db.Mongoose.model carrega a coleção de usuários, a qual é usada para dar um find por todos usuários (filtro vazio = {}). O lean() é opcional, mas uma boa prática de performance, para retornar um JSON text-plain ao invés de objetos Mongoose complexos

Exibir dados do MongoDB


```
router.get('/userlist', function(req, res) {
    var db = require("../db");
    var Users = db.Mongoose.model('usercollection', db.UserSchema, 'usercollection');
    Users.find({}).lean().exec(
        function (e, docs) {
            res.render('userlist', { "userlist": docs });
    });
});
```

- O comando exec executa a consulta em si, passando para o callback um objeto e (erro) e um objeto docs, que contém os resultados da pesquisa. No final, apenas esses o conteúdo de docs é renderizado na tela com res.render

Exibir dados do MongoDB

- O resultado é exibido na view
  - \nodetest\views\userlist.ejs

```
<!DOCTYPE html>
<html lang="en">
<head></head>
<body>
 <h1 class="text-center title-1"> Lista de Usu&aacute;rios </h1>
 <ul>
 <% userlist.forEach(function(user){ %>
 <li><a href="mailto:<%= user.email %>"><%= user.username %></a></li>
 <%})%>
 </ul>
</body>
</html>
```

- Para testar: localhost:3000/userlist

Escrever dados no MongoDB

- Vamos agora criar uma rota para receber um post de forma a permitir salvar novos registros no banco
- Inicialmente vamos criar um formulário na pasta views para coletar os dados
  - Arquivo newuser.ejs

```
<!DOCTYPE html>
<html lang="en">
<head></head>
<body>
   <h1 class="text-center title-1"> Cadastro de Usu&aacute;rio </h1>
   <form action="/adduser" method="post">
      <p>Username:<input type="text" name="username" /></p>
      <p>Email:<input type="text" name="useremail" /></p>
      <p><input type="submit" value="Salvar" /></p>
   </form>
</body>
</html>
```

Escrever dados no MongoDB

- Vamos editar o arquivo \nodetest1\routes\index.js e incluir duas novas rotas
  - Rota GET para exibir o formulário
  - Rota POST para coletar os dados do formulário

- Rota GET

```
/* GET New User page. */
router.get('/newuser', function(req, res) {
res.render('newuser', { title: 'Add New User' });
});
```

Escrever dados no MongoDB

- Rota POST

```
/* POST to Add User Service */
router.post('/adduser', function (req, res) {
    var db = require("../db");
    var userName = req.body.username;
    var userEmail = req.body.useremail;
    var Users = db.Mongoose.model('usercollection', db.UserSchema, 'usercollection');
    var user = new Users({ username: userName, email: userEmail });
    user.save(function (err) {
        if (err) {
            console.log("Error! " + err.message);
            return err;
        }
        else {
            console.log("Post saved");
            res.redirect("userlist");
        }
    });
});
```

- Para testar: localhost:3000/newuser

Créditos

- O código utilizado nesta aula foi desenvolvido por Luiz Duarte
  - https://www.luiztools.com.br/post/tutorial-nodejs-com-mongodb/