Projeto de Banco de Dados NoSQL

Sistema de Recomendação de Negócios

Autores:

Leonardo Matheus	RM: 99824
Cauã Couto	RM: 97755
Kaique Agostinho	RM: 550815
Thiago Gil	RM: 551211

Instituição: FIAP

Disciplina: Mastering Relational and Non-Relational Databases

Turma: 2TDSPN / 2TDSS / 2TDSPV

Data: October 8, 2024

Contents

1 Descrição do Projeto

O sistema visa fornecer recomendações personalizadas de negócios para clientes que desejam investir em diferentes setores, como tecnologia. O sistema coleta informações sobre os objetivos do cliente, capital disponível e outros fatores relevantes, utilizando essas informações para sugerir as melhores práticas e estratégias.

2 Justificativa para a Escolha do MongoDB

A escolha do MongoDB como banco de dados NoSQL se deve à sua flexibilidade em armazenar documentos não estruturados e à capacidade de escalar horizontalmente. A estrutura de dados em BSON permite que informações variadas sobre clientes e recomendações sejam armazenadas de maneira eficiente.

3 Modelo de Dados e Justificativas

3.1 Estruturas de Dados

O modelo de dados incluirá duas coleções principais:

- **Clientes**: Informações sobre os clientes que buscam recomendações.
- **Recomendações**: Sugestões personalizadas para cada cliente.

3.2 Modelo de Clientes

```
{
    "_id": "1",
    "nome": "João Silva",
    "email": "joao@email.com",
    "capital": 100000,
    "objetivos": [
        "crescimento",
        "inovação"
    ],
    "data_registro": "2024-01-15",
    "telefone": "123456789",
    "setor_preferido": "tecnologia",
    "localizacao": "São Paulo",
    "historico_investimentos": [
        "startups",
        "ações"
    ],
    "preferencias": {
        "risco": "moderado",
        "retorno": "alto"
    }
}
```

3.3 Modelo de Recomendações


```
{
    "_id": "1",
    "cliente_id": "1",
    "recomendacao": "Investir em startups de tecnologia focadas em inteligência
    ↪  artificial.",
    "data_recomendacao": "2024-01-16",
    "feedback": [],
    "status": "ativa",
    "prioridade": "alta",
    "custo_estimado": 50000,
    "retorno_estimado": 150000,
    "duracao": "12 meses",
    "risco": "moderado"
}
```

4 Construção de Dados e Operações

4.1 Criação de Documentos JSON/BSON

Aqui estão 10 documentos em formato JSON para as coleções `clientes` e `recomendacoes`.

4.1.1 Documentos da Coleção de Clientes

```
[
    {
        "_id": "1",
        "nome": "João Silva",
        "email": "joao@email.com",
        "capital": 100000,
        "objetivos": ["crescimento", "inovação"],
        "data_registro": "2024-01-15",
        "telefone": "11987654321",
        "setor_interesse": "Tecnologia",
        "localizacao": "São Paulo",
        "feedback": [],
        "status": "ativo"
    },
    {
        "_id": "2",
        "nome": "Maria Oliveira",
        "email": "maria@email.com",
        "capital": 75000,
        "objetivos": ["segurança", "crescimento"],
        "data_registro": "2024-02-10",
        "telefone": "21987654321",
        "setor_interesse": "Finanças",
        "localizacao": "Rio de Janeiro",
        "feedback": [],
        "status": "ativo"
    },
    {
        "_id": "3",
        "nome": "Pedro Santos",
        "email": "pedro@email.com",
        "capital": 120000,
        "objetivos": ["inovação"],
        "data_registro": "2024-03-05",
        "telefone": "31987654321",
        "setor_interesse": "Tecnologia",
        "localizacao": "Belo Horizonte",
        "feedback": [],
        "status": "ativo"
    },
    {
        "_id": "4",
        "nome": "Ana Costa",
        "email": "ana@email.com",
        "capital": 50000,
        "objetivos": ["crescimento"],
        "data_registro": "2024-04-15",
        "telefone": "41987654321",
        "setor_interesse": "Saúde",
        "localizacao": "Curitiba",
        "feedback": [],
        "status": "ativo"
    },
    {
        "_id": "5",
        "nome": "Ricardo Lima",
        "email": "ricardo@email.com",
        "capital": 95000,
        "objetivos": ["inovação", "segurança"],
        "data_registro": "2024-05-20",
        "telefone": "51987654321",
        "setor_interesse": "Educação",
        "localizacao": "Porto Alegre",
        "feedback": [],
        "status": "ativo"
    }
    // Adicione mais documentos conforme necessário
]
```

4.1.2 Documentos da Coleção de Recomendações


```
[
[
    {
        "_id": "1",
        "cliente_id": "1",
        "recomendacao": "Investir em startups de tecnologia focadas em inteligência artificial.",
        "data_recomendacao": "2024-01-16",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "2",
        "cliente_id": "2",
        "recomendacao": "Diversificar investimentos em fundos imobiliários.",
        "data_recomendacao": "2024-02-20",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "3",
        "cliente_id": "3",
        "recomendacao": "Focar em ações de empresas sustentáveis.",
        "data_recomendacao": "2024-03-05",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "4",
        "cliente_id": "4",
        "recomendacao": "Considerar criptomoedas como parte do portfólio.",
        "data_recomendacao": "2024-04-10",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "5",
        "cliente_id": "5",
        "recomendacao": "Aumentar a reserva de emergência com investimentos de baixo risco.",
        "data_recomendacao": "2024-05-15",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "6",
        "cliente_id": "6",
        "recomendacao": "Investir em tecnologia limpa e energias renováveis.",
        "data_recomendacao": "2024-06-01",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "7",
        "cliente_id": "7",
        "recomendacao": "Explorar oportunidades em ações de empresas emergentes.",
        "data_recomendacao": "2024-07-12",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "8",
        "cliente_id": "8",
        "recomendacao": "Participar de crowdfunding de projetos inovadores.",
        "data_recomendacao": "2024-08-20",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "9",
        "cliente_id": "9",
        "recomendacao": "Investir em setores de saúde e biotecnologia.",
        "data_recomendacao": "2024-09-05",
        "feedback": [],
        "status": "ativa"
    },
    {
        "_id": "10",
        "cliente_id": "10",
        "recomendacao": "Considerar ações de empresas com dividendos altos.",
        "data_recomendacao": "2024-10-01",
        "feedback": [],
        "status": "ativa"
    }
]
```

5 Interface de Consulta de Dados

A interface será implementada utilizando Node.js e o framework Express. Ela permitirá a visualização dos clientes e suas recomendações, além de operações CRUD.

5.1 Configuração do Ambiente

Inicie o projeto e instale as dependências necessárias:

```
mkdir sistema-recomendacoes
cd sistema-recomendacoes
npm init -y
npm install express mongoose body-parser ejs
```

5.2 Estrutura do Projeto

A estrutura do projeto é organizada da seguinte forma:

```
sistema-recomendacoes/
|
+-- models/
|   +-- Cliente.js
|   +-- Recomendacao.js
|
+-- views/
|   +-- clientes.ejs
|   +-- recomendacoes.ejs
|
+-- public/
|   +-- styles.css
|
+-- app.js
+-- package.json
```

Descrição dos Componentes:

- **models/**: Contém os modelos de dados utilizados pela aplicação. Cada modelo representa uma coleção no banco de dados MongoDB.
- **views/**: Contém os arquivos EJS que são responsáveis pela renderização das páginas HTML. Esses arquivos utilizam dados dinâmicos enviados pelo servidor.
- **public/**: Contém arquivos estáticos como folhas de estilo CSS e imagens, que são servidos diretamente ao cliente.
- **app.js**: O ponto de entrada da aplicação, onde o servidor é configurado, as rotas são definidas e os middlewares são aplicados.
- **package.json**: Contém informações sobre o projeto, incluindo as dependências utilizadas e scripts de execução.

5.3 Modelos de Dados

Os modelos de dados estão definidos no diretório `models/`.

5.3.1 Cliente.js

Este modelo representa os clientes que receberão recomendações.

```
const mongoose = require('mongoose');

const clientSchema = new mongoose.Schema({
    name: { type: String, required: true },
    email: { type: String, required: true, unique: true, lowercase: true,
    ↪  trim: true },
    capital: { type: Number, required: true },
    goals: { type: [String], required: true },
    registrationDate: { type: Date, default: Date.now },
    phone: { type: String, required: true },
    sectorOfInterest: { type: String, required: true },
    location: { type: String, required: true },
    feedback: { type: [String], default: [] },
    status: { type: String, enum: ['active', 'inactive'], default:
    ↪  'active' }
});

module.exports = mongoose.model('Client', clientSchema);
```

5.3.2 Recomendacao.js

Este modelo armazena as recomendações feitas aos clientes.

```
const mongoose = require('mongoose');

const recomendacaoSchema = new mongoose.Schema({
    cliente_id: { type: mongoose.Schema.Types.ObjectId, ref: 'Cliente',
    ↪  required: true },
    recomendacao: { type: String, required: true },
    data_recomendacao: { type: Date, default: Date.now },
    feedback: { type: [String] },
    status: { type: String, enum: ['ativa', 'finalizada'], default:
    ↪  'ativa' }
});

module.exports = mongoose.model('Recomendacao', recomendacaoSchema);
```

5.4 Configuração do Servidor

A configuração do servidor Express é realizada no arquivo `app.js`.

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const Cliente = require('./models/Cliente');
const Recomendacao = require('./models/Recomendacao');

const app = express();
const PORT = process.env.PORT || 3000;

// Conexão com o MongoDB
mongoose.connect('mongodb://localhost:27017/recomendacoes', {
    useNewUrlParser: true,
    useUnifiedTopology: true
})
.then(() => console.log('Conectado ao MongoDB'))
.catch((error) => console.error('Erro ao conectar ao MongoDB:', error));

// Configuração do middleware
app.use(bodyParser.urlencoded({ extended: true }));
app.set('view engine', 'ejs');
app.use(express.static('public'));

// Rotas
app.get('/clientes', async (req, res) => {
    try {
        const clientes = await Cliente.find();
        res.render('clientes', { clientes });
    } catch (error) {
        console.error('Erro ao buscar clientes:', error);
        res.status(500).send('Erro ao buscar clientes');
    }
});

app.get('/recomendacoes', async (req, res) => {
    try {
        const recomendacoes = await Recomendacao.find().populate('cliente_id');
        res.render('recomendacoes', { recomendacoes });
    } catch (error) {
        console.error('Erro ao buscar recomendações:', error);
        res.status(500).send('Erro ao buscar recomendações');
    }
});

// Iniciar o servidor
app.listen(PORT, () => {
    console.log(`Servidor rodando na porta ${PORT}`);
});
```

6 Views

Crie os arquivos EJS no diretório `views/` para renderizar as páginas de clientes e recomendações. Os arquivos `clientes.ejs` e `recomendacoes.ejs` devem conter a estrutura HTML necessária para exibir os dados.

6.1 clientes.ejs

O arquivo `clientes.ejs` deve conter a seguinte estrutura:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale
        =1.0">
    <title>Clientes</title>
    <link rel="stylesheet" href="/styles.css">
</head>
<body>
    <header>
        <h1>Lista de Clientes</h1>
    </header>
    <main>
        <table>
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Nome</th>
                    <th>Email</th>
                    <th>Capital</th>
                    <th>Setor de Interesse</th>
                    <th>Status</th>
                </tr>
            </thead>
            <tbody>
                <% clientes.forEach(cliente => { %>
                    <tr>
                        <td><%= cliente._id %></td>
                        <td><%= cliente.nome %></td>
                        <td><%= cliente.email %></td>
                        <td><%= cliente.capital %></td>
                        <td><%= cliente.setor_interesse %></td>
                        <td><%= cliente.status %></td>
                    </tr>
                <% }) %>
            </tbody>
        </table>
    </main>
</body>
</html>
```

6.2 recomendacoes.ejs

O arquivo `recomendacoes.ejs` deve conter a seguinte estrutura:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale
        =1.0">
    <title>Recomendacoes</title>
    <link rel="stylesheet" href="/styles.css">
</head>
<body>
    <header>
        <h1>Recomendacoes</h1>
    </header>
    <main>
        <ul>
            <% recomendacoes.forEach(recomendacao => { %>
                <li>
                    <strong>Cliente:</strong> <%= recomendacao.
                        cliente_nome %><br>
                    <strong>Recomendacao:</strong> <%= recomendacao.
                        mensagem %>
                </li>
            <% }) %>
        </ul>
    </main>
</body>
</html>
```

7 Conclusão

O sistema de recomendação de negócios proposto utilizando MongoDB permite a personalização das sugestões de investimento para os clientes. A flexibilidade do MongoDB possibilita uma rápida adaptação às necessidades do mercado, garantindo que os clientes recebam recomendações relevantes.