

## **CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS**

**P.W. – INNER JOIN**

## SISTEMA DE CADASTRO DE LIVRO

- Sistema de cadastro utilizando três tabelas;
  - Editora
  - Autor
  - Livro

“Livro” recebe duas chaves estrangeiras, sendo elas referentes a “Editora” e a “Autor”. O método *onDelete* está em CASCADE, ou seja, quando se deleta uma editora ou um ator, todas as ocorrências que estão relacionadas serão deletadas.

As três tabelas possuem CRUD.

Há uma tabela que recebe o nome “usuário”, a qual está relacionada a função de login.

## SQL

```
create database livraria;
```

```
use livraria;
```

```
create table usuario(
```

```
id_usuario INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
```

```
username VARCHAR (30),
```

```
email VARCHAR (255),
```

```
senha VARCHAR (40),
```

```
createdAt DATETIME, updatedAt DATETIME);
```

```
create table editora(
```

```
id_editora int not null auto_increment primary key,
```

```
nome varchar(50),
```

```
estado CHAR (2),  
  
cidade VARCHAR (40), logradouro VARCHAR (40), createdAt datetime,  
updatedAt datetime);  
  
create table autor(  
  
id_autor int not null auto_increment primary key,  
  
nomeA varchar (100), createdAt datetime,  
updatedAt datetime);  
  
create table livro (  
  
id_livro INT NOT NULL AUTO_INCREMENT PRIMARY KEY,  
  
titulo VARCHAR (300), createdAt datetime,  
updatedAt datetime,  
  
fk_editora int,  
  
fk_autor int,  
  
INDEX livro_FKIndex1(fk_editora),  
  
FOREIGN KEY(fk_editora)  
  
REFERENCES editora(id_editora)  
  
ON DELETE CASCADE  
  
ON UPDATE CASCADE, INDEX livro_FKIndex2(fk_autor), foreign key  
(fk_autor) references autor(id_autor) ON DELETE CASCADE ON UPDATE  
CASCADE);  
  
  
CREATE VIEW selecLivro AS  
  
SELECT livro.id_livro, livro.titulo, editora.nome AS editora, autor.nomeA as  
autor
```

FROM livro

INNER JOIN editora ON livro.fk\_editora = editora.id\_editora INNER JOIN autor  
ON livro.fk\_autor = autor.id\_autor;

select \* from selecLivro;

## Db.js

```
const Sequelize = require ('sequelize');

const sequelize = new Sequelize ('livraria', 'root', 'root', {
  host: "localhost",
  port: "3306",
  dialect: "mysql"
});

module.exports = {
  Sequelize: Sequelize,
  sequelize: sequelize
}
```

## MODELS

### Autor.js

```
const db = require('./db');
const Autor = db.sequelize.define('autor', {
  id_autor: {
    type: db.Sequelize.INTEGER,
    autoIncrement: true,
    allowNull: false,
    primaryKey: true
  },
  nomeA: {
    type: db.Sequelize.TEXT
  }
  //freezeTableName: true define
  //o nome da tabela sem o S
}, { freezeTableName: true });

module.exports = Autor;
```

### Editora.js

```

const db = require('./db');
const Editora = db.sequelize.define('editora', {
  id_editora: {
    type: db.Sequelize.INTEGER,
    autoIncrement: true,
    allowNull: false,
    primaryKey: true
  },
  nome: {
    type: db.Sequelize.TEXT
  },
  estado: {
    type: db.Sequelize.TEXT
  },
  cidade: {
    type: db.Sequelize.TEXT
  },
  logradouro: {
    type: db.Sequelize.TEXT
  }
}, { freezeTableName: true });

module.exports = Editora;

```

## Livro.js

```

const db = require('./db');
const Editora = require('../models/editora');
const Autor = require('../models/autor');
const Livro = db.sequelize.define('livro', {
  id_livro: {
    type: db.Sequelize.INTEGER,
    autoIncrement: true,
    allowNull: false,
    primaryKey: true
  },
  titulo: {
    type: db.Sequelize.STRING
  },
  fk_editora: {
    type: db.Sequelize.INTEGER,

```

```

        references: { model: 'editora', key: 'id_editora' },
        onDelete: 'CASCADE',
        allowNull: false,
      },
      fk_autor: {
        type: db.Sequelize.INTEGER,
        references: { model: 'autor', key: 'id_autor' },
        onDelete: 'CASCADE',
        allowNull: false,
      }
    }, {freezeTableName: true});

module.exports = Livro;

```

## Usuario.js

```

const db = require('./db');

const Usuario = db.sequelize.define('usuario', {
  id_usuario: {
    type: db.Sequelize.INTEGER,
    autoIncrement: true,
    allowNull: false,
    primaryKey: true
  },
  username: {
    type: db.Sequelize.TEXT
  },
  email: {
    type: db.Sequelize.TEXT
  },
  senha: {
    type: db.Sequelize.TEXT
  }
}, {freezeTableName: true define
  //o nome da tabela sem o S
}, { freezeTableName: true });

module.exports = Usuario;

```

## MIDDLEWARE

Quando se acessa uma página pela URL, é preciso fazer um login para verificação, e quem cria esta obrigatoriedade é o middleware.

### ***VerificarAutenticacao.js***

```
function verificarAutenticacao(req, res, next) {  
  if (req.session.usuario) {  
    return next();  
  }  
  req.session.returnTo = req.originalUrl;  
  
  res.redirect('/auth/login');  
}  
module.exports = verificarAutenticacao;
```

## ROUTES

### *Rota\_auth.js*

```
// routes/auth.js
const express = require('express');
const router = express.Router();
const Usuario = require('../models/usuario');

router.get('/login', (req, res) => {
  res.render('./admin/login');
});
router.get('/register', (req, res) => {
  res.render('./admin/singup');
})
router.post('/register', (req, res) => {
  Usuario.create({
    username: req.body.username,
    email: req.body.email,
    senha: req.body.senha
  }).then(() => {
    res.redirect("/auth/login");
  }).catch((erro) => {
    res.send('Houve um erro: ' + erro);
  });
});

router.post('/login', async (req, res) => {
  const { username, senha } = req.body;

  try {
    const usuario = await Usuario.findOne({ where: { username } });

    if (usuario && usuario.senha === senha) {
      req.session.usuario = usuario;

      const returnTo = req.session.returnTo || '/';
      delete req.session.returnTo;

      res.redirect(returnTo);
    } else {
      res.render('login', { error: 'Credenciais inválidas' });
    }
  } catch (error) {
    console.error(error);
    res.status(500).json({ error: 'Erro ao autenticar usuário' });
  }
});
module.exports = router;
```



## rota\_autor.js

```
const express = require('express');
const router = express.Router();

const Autor = require('../models/autor')

const verificarAutenticacao =
require('../middleware/verificarAutenticacao');

router.get('/autor', verificarAutenticacao, (req, res) => {
  Autor.findAll().then((autores) => {
    autores = autores.map((autor) => {
      return autor.toJSON();
    });
    res.render("../admin/autor/autor", { autores: autores });
  });
});

router.get('/autor/add', (req, res) => {
  res.render('../admin/autor/addautor');
})

router.get('/editar_autor/:id', (req, res) => {
  Autor.findAll({ where: { 'id_autor':
req.params.id } }).then((autores) => {
    autores = autores.map((autor) => { return autor.toJSON() });
    res.render("../admin/autor/editautor", { autores: autores });
  });
});

router.post('/autor/novo', (req, res) => {
  Autor.create({
    nomeA: req.body.nomeA
  }).then(() => {
    res.redirect("/rota_autor/autor");
  }).catch((erro) => {
    res.send('Houve um erro: ' + erro);
  });
});

router.post('/autor/editar_autor', (req, res) => {
  Autor.update({
    nomeA: req.body.nomeA
  },
  {
    where: { id_autor: req.body.id_autor }
  })
});
```

```

    }).then(() => {
      res.redirect("/rota_autor/autor");
    }).catch((erro) => {
      res.send("Este(a) Autor(a) não existe " + erro);
    });
  });

router.get('/deletar_autor/:id', (req, res) => {
  Autor.destroy({ where: { 'id_autor': req.params.id } }).then(() => {
    res.redirect("/rota_autor/autor");
  }).catch((err) => {
    res.render("Esse(a) autor(a) não existe");
  });
});

module.exports = router;

```

### **rota\_editora.js**

```

const express = require('express');
const router = express.Router();

const Editora = require('../models/editora')

const verificarAutenticacao =
require('../middleware/verificarAutenticacao');

router.get('/editora', verificarAutenticacao, (req, res) => {
  Editora.findAll().then((editoras) => {
    editoras = editoras.map((editora) => {
      return editora.toJSON();
    });
    res.render("./admin/editora/editora", { editoras: editoras });
  });
});

router.get('/editora/add', (req, res) => {
  res.render('./admin/editora/addeditora');
})

router.get('/editar_editora/:id', (req, res) => {
  Editora.findAll({ where: { 'id_editora':
req.params.id } }).then((editoras) => {
    editoras = editoras.map((editora) => { return
editora.toJSON() });
    res.render("./admin/editora/editeditora", {editoras: editoras });
  });
});

```

```

});

router.post('/editora/nova', (req, res) => {
  Editora.create({
    nome: req.body.nome,
    estado: req.body.estado,
    cidade: req.body.cidade,
    logradouro: req.body.logradouro
  }).then(() => {
    res.redirect("/rota_editora/editora");
  }).catch((erro) => {
    res.send('Houve um erro: ' + erro);
  });
});

router.post('/editora/editar_editora', (req, res) => {
  Editora.update({
    nome: req.body.nome,
    estado: req.body.estado,
    cidade: req.body.cidade,
    logradouro: req.body.logradouro
  },
  {
    where: { id_editora: req.body.id_editora }
  }).then(() => {
    res.redirect("/rota_editora/editora");
  }).catch((erro) => {
    res.send("Esta editora não existe " + erro);
  });
});

router.get('/deletar_editora/:id', (req, res) => {
  Editora.destroy({ where: { 'id_editora': req.params.id } }).then(()
=> {
    res.redirect("/rota_editora/editora");
  }).catch((err) => {
    res.send("Esse editora não existe" + err);
  });
});

module.exports = router;

```

### **rota\_livro.js**

```

const express = require('express');
const router = express.Router();

const Autor = require('../models/autor');

```

```

const Editora = require('../models/editora');
const Livro = require('../models/livro');

const verificarAutenticacao =
require('../middleware/verificarAutenticacao');

router.get('/livro', verificarAutenticacao, (req, res) => {
  Livro.sequelize.query("select * from selecLivro",
    { model: Livro }).then(function (livros) {
      var nlivros = JSON.parse(JSON.stringify(livros));
      res.render("admin/livro/livro",
        { livros: nlivros });
    });
});

router.get('/livro/add', async (req, res) => {
  try {
    const [editoras, autores] = await Promise.all([
      Editora.findAll(),
      Autor.findAll()
    ]);

    var neditoras = JSON.parse(JSON.stringify(editoras));
    var nautores = JSON.parse(JSON.stringify(autores));

    res.render("admin/livro/addlivro", { editoras: neditoras,
      autores: nautores });
  } catch (erro) {
    res.send("Ocorreu um erro ao buscar editoras e autores: " +
erro);
  }
});

router.get('/editar_livroEditora/:id', (req, res) => {
  Livro.findAll({ where: { 'id_livro':
req.params.id } }).then((livros)=> {
    //pega as turmas cadastradas para popular o select do html
    Editora.findAll().then((editoras) => {
      var neditora = JSON.parse(JSON.stringify(editoras));
      var nlivro = JSON.parse(JSON.stringify(livros));
      res.render("admin/livro/editlivro", {
        livros: nlivro,
        editoras: neditora
      });
    });
  });
});

router.get('/editar_livroAutor/:id', (req, res) => {

```

```

    Livro.findAll({ where: { 'id_livro':
req.params.id } }).then((livros)=> {
    //pega as turmas cadastradas para popular o select do html
    Autor.findAll().then((autores) => {
        var nautor = JSON.parse(JSON.stringify(autores));
        var nlivro = JSON.parse(JSON.stringify(livros));
        res.render("admin/livro/editlivroAutor", {
            autores: nautor,
            livros: nlivro
        });
    });
});
});
});

router.post('/livro/novo', (req, res) => {

    Livro.create({
        titulo: req.body.titulo,
        fk_editora: req.body.fk_editora,
        fk_autor: req.body.fk_autor
    }).then(() => {
        res.redirect("/rota_livro/livro");
    }).catch((erro) => {
        res.send('Houve um erro' + erro);
    });
});

router.post('/livro/editar_livroEditora', (req, res) => {
    Livro.update({
        titulo: req.body.titulo,
        fk_editora: req.body.fk_editora
    },
    {
        where: { id_livro: req.body.id_livro }
    }).then(() => {
        res.redirect("/rota_livro/livro");
    }).catch((erro) => {
        res.send("Este livro não existe " + erro);
    });
});

router.post('/livro/editar_livroAutor', (req, res) => {
    Livro.update({

```

```

        titulo: req.body.titulo,
        fk_autor: req.body.fk_autor
    },
    {
        where: { id_livro: req.body.id_livro }
    }).then(() => {
        res.redirect("/rota_livro/livro");
    }).catch((erro) => {
        res.send("Este livro não existe " + erro);
    });
});

router.get('/deletar_livro/:id', (req, res) => {
    Livro.destroy({ where: { 'id_livro': req.params.id } }).then(() => {
        res.redirect("/rota_livro/livro");
    }).catch((err) => {
        res.render("Esse livro não existe");
    });
});

module.exports = router;

```

## HANDLEBARS

**Autor:**

**addAutor**

```

<main>
  <div class="geral">
    <div class="container">
      <h3> Novo Autor: </h3>

      <div>
        <form action="/rota_autor/autor/novo" method="POST">
          <div class="inputs">
            <label for="nomeA">Nome: </label>
            <input type="text" id="nomeA" name="nomeA" >
          </div>
          <br>
          <button type="submit">Cadastrar Autor(a)</button>
        </form>
        <a href="/rota_autor/autor/">
          <button>
            Voltar
          </button>
        </a>
      </div>
    </div>
  </main>

```

```

        </button>
      </a>
    </div>
  </div>
</main>

```

## Editautor

```

<main>
  <div class="geral">
    <div class="container">
      <h3> Editar Autor: </h3>
      <div class="">
        <div class="">
          <form action="/rota_autor/autor/editar_autor"
method="POST">

            {{#each autores}}
            <div class="inputs">
              <label for="id_autor">ID: </label>
              <input type="text" name="id_autor"
readonly="true" class="form-control" value="{{id_autor}}">
              <br>

              <label for="nomeA">Nome: </label>
              <input type="text" id="nomeA" name="nomeA"
class="form-control" value="{{nomeA}}">
              <br>

              <button type="submit" class="btn
btnsuccess">Editar Autor(a)</button>
            </div>
          {{/each}}
        </form>
      </div>
    </div>
  </div>
</main>

```

## Autor

```

<main>
  <div class="geral">
    <div class="header">
      <h2>Lista Autores</h2>

```





```

        <a href="/rota_editora/editora">
            <button>
                Editoras
            </button>
        </a>
        <a href="/rota_livro/livro/">
            <button>
                Livros
            </button>
        </a>
    </div>
</div>
</main>

```

**Editora:**

**Addeditora**

```

<main>
    <div class="geral">
        <div class="container">
            <h3> Nova editora: </h3>

            <form action="/rota_editora/editora/nova"
method="POST">
                <div class="inputs">
                    <label for="nome">Nome: </label>
                    <input type="text" id="nome" name="nome" ><br>
                    <label for="estado">Estado: </label>
                    <input type="text" id="estado" name="estado" ><br>
                    <label for="cidade">Cidade: </label>
                    <input type="text" id="cidade" name="cidade" ><br>
                    <label for="nome">Logradouro: </label>
                    <input type="text" id="logradouro" name="logradouro"
                >
                    <br>
                    <button type="submit">Cadastrar editora</button>
                </div>

                </form>
                <a href="/rota_autor/autor/">
                    <a href="/rota_editora/editora">
                        <button>
                            Voltar
                        </button>
                    </a>
                </div>

```

```

        </div>
    </div>
</main>

```

## Edit Editora

```

<main>
    <div class="geral">
        <div class="container">
            <h3> Editar Editora: </h3>

            <form
                action="/rota_editora/editora/editar_editora" method="POST">

                {{#each editoras}}
                    <div class="inputs">
                        <label for="id_editora">ID: </label>
                        <input type="text" name="id_editora"
                            readonly="true" class="form-control" value="{{id_editora}}">
                        <br>

                        <label for="nome">Nome: </label>
                        <input type="text" id="nome" name="nome"
                            class="form-control" value="{{nome}}">
                        <br>
                        <label for="estado">Estado: </label>
                        <input type="text" id="estado" name="estado"
                            class="form-control" value="{{estado}}">
                        <br>
                        <label for="cidade">Cidade: </label>
                        <input type="text" id="cidade" name="cidade"
                            class="form-control" value="{{cidade}}">
                        <br>
                        <label for="logradouro">Logradouro: </label>
                        <input type="text" id="logradouro"
                            name="logradouro" class="form-control" value="{{logradouro}}">
                        <br>

                        <button type="submit" class="btn
                            btnsuccess">Editar Editora</button>
                    </div>

                    {{/each}}

                </form>

            </div>
        </div>
    </main>

```

## Editora

```
<main>
  <div class="geral">
    <div class="header">
      <h2>Lista Editoras</h2>
      <hr>
      <a href="/rota_editora/editora/add">
        <button>Nova Editora</button>
      </a>
    </div>
    <div class="container_listar">
      {{#each editoras}}
        <div class="card">
          <div class="leftcard">

            <div class="id">
              <h3>ID:{{id_editora}}</h3>
            </div>
            <div class="buttons">
              <a
href="/rota_editora/editar_editora/{{id_editora}}">

                <button>
                  Editar Editora
                </button>
              </a>

              <a
href="/rota_editora/deletar_editora/{{id_editora}}">
                <button>
                  Deletar Editora
                </button>
              </a>
            </div>
          </div>
          <div class="rightcard">
            <div class="toprightcard">
              <small><h4>Nome:</h4><h5>{{nome}}</h5></small>
            </div>
            <div class="bottomrightcard">

              <small><b>Estado:</b>{{estado}}</small>

              <small><b>Cidade:</b>{{cidade}}</small>
              <small><b>Logradouro:</b>{{logradouro}}</small>
            </div>
          </div>
        </div>
      </each>
    </div>
  </div>
</main>
```

```

        </div>
    </div>
    {{/each}}

    </div>
    <br> <br>
    <div class="buttonsTo">
        <a href="/rota_autor/autor/">
            <button>
                Autores
            </button>
        </a>
        <a href="/rota_editora/editora">
            <button>
                Editoras
            </button>
        </a>
        <a href="/rota_livro/livro/">
            <button>
                Livros
            </button>
        </a>
    </div>
</div>
</main>

```

## Livro:

### editlivroEditora

```

<main>
    <div class="geral">
        <div class="container">
            <h3>Editar Livro:</h3>

            <form action="/rota_livro/livro/editar_livroEditora"
method="post">

                <div class="inputs">
                    {{#each livros}}

                        <label for="id_livro">ID:</label>
                        <input type="text" readonly="true"
name="id_livro" id="id_livro" value="{{id_livro}}">
                        <br>
                        <label for="Título">Título: </label>
                        <input type="text" id="titulo" name="titulo"
value="{{titulo}}">
                    
```

```

        {{/each}}

        <select name="fk_editora" class="custom-select">
            {{#each editoras}}
            <option
value="{{id_editora}}">{{nome}}</option>
            {{else}}
            <option value="0">Nenhuma Editora
                registrada</option>
            {{/each}}
        </select>

        <br>
        <br>
        <button type="submit" class="btn btn-
success">Editar

                Livro</button>
            </div>
        </form>

    </div>
</div>
</main>

```

### **editlivroAutor**

```

<main>
    <div class="geral">
        <div class="container">
            <h3>Editar Livro:</h3>

            <form action="/rota_livro/livro/editar_livroAutor"
method="post">

                <div class="inputs">
                    {{#each livros}}

                        <label for="id_livro">ID:</label>
                        <input type="text" readonly="true"
name="id_livro" id="id_livro" class="form-control" value="{{id_livro}}">
                        <br>
                        <label for="Título">Título: </label>
                        <input type="text" id="titulo" name="titulo"
value="{{titulo}}">

                    {{/each}}
                </div>
            </form>
        </div>
    </div>
</main>

```

```

                <select name="fk_autor" class="custom-select">
                    {{#each autores}}
                    <option
value="{{id_autor}}">{{nomeA}}</option>
                    {{else}}
                    <option value="0">Nenhuma turma
                        registrada</option>
                    {{/each}}
                </select>
                <br>
                <br>
                <button type="submit" class="btn btn-
success">Editar
                    Livros</button>
                </div>
            </form>

        </div>
    </div>
</main>

```

## Addlivro

```

<main>
    <div class="geral">
        <div class="container">
            <h3>Novo Livro:</h3>

            <form action="/rota_livro/livro/novo" method="post">
<div class="inputs">
                <label for="titulo">Título</label>
                <input type="text" id="titulo" name="titulo"
placeholder="Título do livro">
                <br>
                <label for="editora">Editora:</label>
                <select name="fk_editora" class="custom-select">
                    {{#each editoras}}
                    <option
value="{{id_editora}}">{{nome}}</option>
                    {{else}}
                    <option value="0">Nenhuma Editora
registrada</option>
                    {{/each}}
                </select>
                <br>
                <label for="autor">Autor(a):</label>

```

```

                <select name="fk_autor" class="custom-select">
                    {{#each autores}}
                    <option
value="{{id_autor}}">{{nomeA}}</option>
                    {{else}}
                    <option value="0">Nenhum(a) Autor(a)
registrado(a)</option>
                    {{/each}}
                </select>

                <br>
                <br>
                <button type="submit" class="btn btn-
success">Criar

                    Livro</button>
                </div>
            </form>
            <a href="/rota_livro/livro/">
                <button>
                    Voltar
                </button>
            </a>

        </div>
    </div>
</main>

```

## Livro

```

<main>
    <div class="geral">
        <div class="header">
            <h2>Lista de Livros</h2>
            <a href="/rota_livro/livro/add">
                <button>Novo Livro</button>
            </a>
        </div>
        <div class="container_listar">
            {{#each livros}}
                <div class="card">
                    <div class="leftcard">
                        <div class="id">
                            <h4>ID: {{id_livro}}</h4>
                        </div>
                        <div class="buttons">
                            <a
href="/rota_livro/editar_livroEditora/{{id_livro}}">
                                <button>Editar Livro (Editora)</button>

```

```

        </a>
        <a
href="/rota_livro/editar_livroAutor/{{id_livro}}">
        <button>Editar Livro (Autor)</button>
        </a>
        <a
href="/rota_livro/deletar_livro/{{id_livro}}">
        <button>Deletar Livro</button>
        </a> <br>
    </div>
</div>
<div class="rightcard">
    <div class="toprightcard">
        <small><h4>Titulo:</h4>
<h5>{{titulo}}</h5></small>
    </div>
    <div class="bottomrightcard">
        <small><b>Editora:</b>
{{editora}}</small><br>
        <small><b>Autor:</b> {{autor}}</small><br>
    </div>
</div>
    </div>
    {{/each}}
</div>
<br> <br>
<div class="buttonsTo">
    <a href="/rota_autor/autor/">
        <button>
            Autores
        </button>
    </a>
    <a href="/rota_editora/editora">
        <button>
            Editoras
        </button>
    </a>
    <a href="/rota_livro/livro/">
        <button>
            Livros
        </button>
    </a>
</div>
</div>
</main>

```



## Usuario:

### Login

```
<!-- views/login.handlebars -->
<div class="geral">
  <div class="container">
<div class="inputs">
<form action="/auth/login" method="POST">
  <input type="text" name="username" placeholder="Nome de Usuário"
required> <br>
  <input type="password" name="senha" placeholder="Senha" required>
<br>
  <button type="submit">Entrar</button>

</form>
</div>

  <p style="color: white;">Não possui uma conta? Cadastre-se aqui</p>
  <a href="/auth/register"><button type="submit">Cadastrar</button></a>
</div>
</div>
<p>{{error}}</p>
```

### Singup

```
<div class="geral">
  <div class="container">
<div class="inputs">
<h1>Registro</h1>

  <form action="/auth/register" method="POST">
    <input type="text" name="username" placeholder="Nome de Usuário"
required> <BR>
    <input type="email" name="email" placeholder="E-mail" required>
<BR>
    <input type="password" name="senha" placeholder="Senha" required>
<BR>
    <button type="submit">Registrar</button>
  </form>
</div>
  <p>{{error}}</p>

  <a href="/auth/login"><p style="color: white;">Já tem uma conta? Faça
login aqui</p></a>
</div>
</div>
```

```
</body>  
</html>
```

## **Main**

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
  <meta charset="UTF-8">  
  <meta http-equiv="X-UA-Compatible" content="IE=edge">  
  <meta name="viewport" content="width=device-width, initial-  
scale=1.0">  
  <link rel="stylesheet" href="/public/home.css">  
  
  <title>livraria</title>  
</head>  
<body>  
  <div>  
    {{{body}}}  
  </div>  
</body>  
  
</html>
```

## App.js

```
const express = require("express");
const handlebars = require('express-handlebars');
const bodyParser = require('body-parser');
const app = express();
const rota_editora = require('./routes/rota_editora');
const rota_autor = require('./routes/rota_autor');
const rota_livro = require('./routes/rota_livro');
const rota_auth = require('./routes/rota_auth');
const session = require('express-session');

app.use(session({
  secret: 'seuSegredo',
  resave: false,
  saveUninitialized: true
}));

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

app.engine('handlebars', handlebars.engine({ defaultLayout: 'main' }));

app.use('/public', express.static('public/css'));

app.set('view engine', 'handlebars');

app.use('/rota_editora', rota_editora);
app.use('/rota_autor', rota_autor);
app.use('/rota_livro', rota_livro);
app.use('/auth', rota_auth);

const PORT = 8000;
app.listen(PORT, () => {
  console.log("Servidor Rodando");
});
```

## CSS

### Home.css

```
@import
url('https://fonts.googleapis.com/css2?family=Rubik:wght@400;700&display=
swap');

*{
  margin: 0;
  padding: 0;
  border: 0;
  font-family: 'Rubik', sans-serif;
}
body {background-color: rgb(27, 27, 56);}

main {
  width: 100vw;
  height: 100%;
  background-color: rgb(27, 27, 56);
}

.geral {
  width: 100%;
  height: 100%;
  justify-content: center;
  display: flex;
  align-items: center;
  flex-direction: column;
}

.header {
  width :100%;

  display:flex;
  padding: 30px 0;
  gap:10px;
  align-items: center;
  justify-content: center;
  flex-direction: column;
}

.container_listar {
  width: 100%;
  display: flex;
  flex-wrap: wrap;
  gap:20px;
  align-items: center;
```

```
        justify-content: center;
    }
    .card {
        width: 40%;
        height: 200px;
        display: flex;
        border-radius: 15px;
    }

    .leftcard {
        height: 100%;
        width: 30%;
        background-color: blueviolet;
        display: flex;
        flex-direction: column;
        align-items: center;
        justify-content: center;
    }

    .rightcard {
        height: 100%;
        width: 70%;
        display: flex;
        background-color: rgb(248, 245, 245);
        flex-direction: column;
    }
    .toprightcard {
        height: 70%;

        padding: 20px;
        justify-content: start;
    }

    .bottomrightcard {
        height: 30%;
        display: flex;
        justify-content: space-around;
        align-items: center;
    }

    small {
        display: flex;
        flex-direction: column;
        justify-content: center;
    }
    .id{
```

```
    height:45%;
    color:white;
    margin-top: 10px;
    font-weight: 700;
}
.buttons {
    display: flex;
    justify-content: center;
    align-items: center;
    flex-direction: column;
    gap:10px;
    height:55%;
}

b {
    font-weight: 700;
    color: #948d8d;
}

button {
    padding: 8px 12px;
    border: none ;
    border-radius: 8px;
    background: blue;
    cursor: pointer;
    color:white;
    font-weight: 400;
}
button:hover {
    transform: scale(1.05);
    transition: all 0.8s;
}

h2, h3{
    color:white;
    font-weight: 700;
}

.container {
    display: flex;
    flex-direction: column;
    align-items: center;
    justify-content: center;
    width: 50%;
    margin-top: 60px;
    gap:20px;
    height: 100%;
}
```

```
.inputs {
  border: 1px solid black;
  display: flex;
  background-color: aliceblue;
  padding: 20px 40px;
  border-radius: 15px;
  align-items: center;
  justify-content: center;
  flex-direction: column ;
}

label{
  font-weight: 500;
  color: black;
}

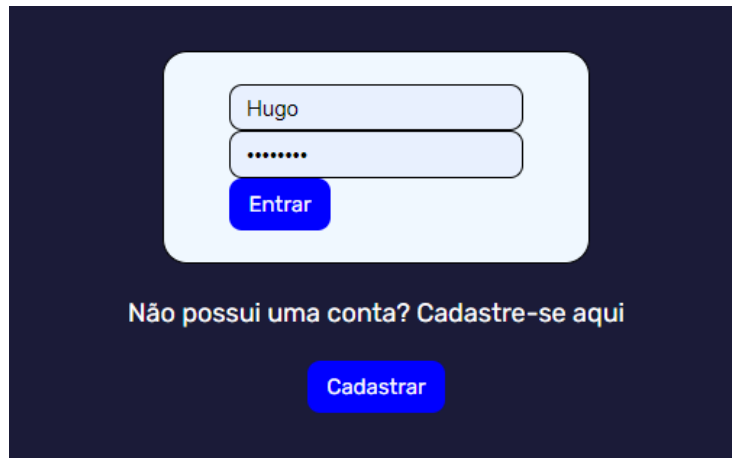
input{
  padding: 5px 10px;
  font-weight: 400;
  font-size: 14px;
  border: 1px solid black;
  border-radius: 8px;
}

select{  padding: 5px 10px;
  font-weight: 400;
  font-size: 14px;
  border: 1px solid black;
  border-radius: 8px; }

h5 {
  font-size: 34PX;
  color: #948d8d;
}
```

## PRINTS

### Login



A login form on a dark blue background. The form is a light blue rounded rectangle containing two input fields: the first contains the text 'Hugo' and the second contains seven dots. Below the fields is a blue button with the text 'Entrar'. Below the form, the text 'Não possui uma conta? Cadastre-se aqui' is displayed, followed by a blue button with the text 'Cadastrar'.

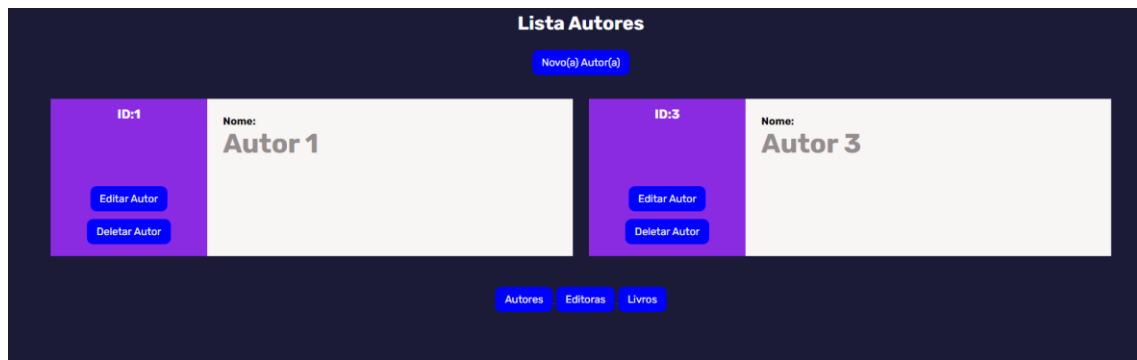
### SingUp



A registration form on a dark blue background. The form is a light blue rounded rectangle with the title 'Registro' in bold. It contains three input fields: 'Nome de Usuário', 'E-mail', and 'Senha'. Below the fields is a blue button with the text 'Registrar'. Below the form, the text 'Já tem uma conta? Faça login aqui' is displayed.

### Autor





**addAutor**

**Novo Autor:**

Nome:

Cadastrar Autor(a)

Voltar

**EditarAutor**

## Editar Autor:

ID:

1

Nome:

Autor 1

Editar Autor(a)

## Editora

## Lista Editoras

Nova Editora

ID:3

Editar Editora

Deletar Editora

Nome:

Editora C

Estado:

MG

Cidade:

Belo Horizonte

Logradouro:

Praça C

Autores

Editoras

Livros

## Add Editora

**Nova editora:**

Nome:

Estado:

Cidade:

Logradouro:

Cadastrar editora

Voltar

**Edit Editora**

Editar Editora:

ID:

3

Nome:

Editora C

Estado:

MG

Cidade:

Belo Horizonte

Logradouro:

Praça C

Editar Editora

Livro

Lista de Livros

Novo Livro

ID: 11

Editar Livro (Editora)

Editar Livro (Autor)

Deletar Livro

Titulo:

Livro A

Editora:

Editora C

Autor:

Autor 1

Autores

Editoras

Livros

Add Livro

### Novo Livro:

Título

Editora:

Editora C ▾

Autor(a):

Autor 1 ▾

Criar Livro

Voltar

Edit Livro (editora)

### Editar Livro:

ID:

Título:

Editora C ▾

Editar Livro

Edit Livro (Autor)

## Editar Livro:

ID:

11

Título:

Livro A

Autor 1 ▾

Editar Livros