adminRoutes.js:

```
const express = require('express');
const router = express.Router();
const multer = require('multer');
const path = require('path');
const adminController = require('../controllers/adminController');

// Configuração do Multer
const storage = multer.diskStorage({
    destination: (req, file, cb) => cb(null, 'public/uploads/'),
    filename: (req, file, cb) => {
      const ext = path.extname(file.originalname).toLowerCase();
      cb(null, `${file.fieldname}-${Date.now()}${ext}`);
  }
});

function fileFilter(req, file, cb) {
  const isPDF = file.mimetype === 'application/pdf';
  const isFotoPerfil = file.fieldname === 'foto_perfil';

  if (isPDF || isFotoPerfil) {
    cb(null, true); // Aceita o upload
  } else {
    cb(new Error('Somente arquivos PDF ou imagem de perfil são
permitidos.'));
  }
}

const upload = multer({ storage, fileFilter });

router.get('/usuarios/index', adminController.renderUsuarios);
router.get('/usuarios/buscar', adminController.buscarUsuarios);
router.delete('/usuarios/deletar/:cod', adminController.deletarUsuarios);
router.get('/usuarios/editar/:cod', adminController.editarUsuario);
router.put('/usuarios/editar/:cod', upload.single('foto_perfil'),
adminController.salvarEdicaoUsuario);

router.get('/motoristas/index', adminController.renderMotoristas);
router.get('/motoristas/buscar', adminController.buscarMotoristas);
router.get('/motoristas/editar/:cod', adminController.editarMotorista);
```

```
router.delete('/motoristas/deletar/:cod',
adminController.deletarMotoristas);
router.put(
  '/motoristas/editar/:cod',
  upload.fields([
    { name: 'foto_perfil', maxCount: 1 },
    { name: 'carteira_trab', maxCount: 1 },
    { name: 'cursos', maxCount: 1 },
    { name: 'habilitacao', maxCount: 1 },
    { name: 'comprov_resid', maxCount: 1 },
    { name: 'comprov_escola', maxCount: 1 },
    { name: 'titulo_eleitor', maxCount: 1 },
    { name: 'ant_crim', maxCount: 1 },
    { name: 'exame_tox', maxCount: 1 },
  ]),
  adminController.salvarEdicaoMotorista
);

router.get('/viagens/index', adminController.renderViagens);
router.get('/viagens/buscar-eventos',
adminController.renderBuscarEventos);
router.get('/viagens/nova-viagem', adminController.renderNovaViagem);
router.post('/viagens/add-nova-viagem',
adminController.renderCadastrarViagem);
router.get('/viagens/ver-viagem/:cod', adminController.renderVerViagem);

router.get('/solicitacoes/index', adminController.renderSolicitacoes);

module.exports = router;
```

authRoutes.js:

```
const express = require('express');
const router = express.Router();
const multer = require('multer');
const path = require('path');
const authController = require('../controllers/authController');

// Configuração do Multer
const storage = multer.diskStorage({
```

```
    destination: (req, file, cb) => cb(null, 'public/uploads/'),
    filename: (req, file, cb) => {
    const ext = path.extname(file.originalname).toLowerCase();
    cb(null, `${file.fieldname}-${Date.now()}${ext}`);
  }
});

const upload = multer({ storage });

// Rotas
router.get('/entrada', authController.renderEntrada);
router.get('/cadastro', authController.renderCadastro);
router.get('/cadastro-sucesso', authController.renderCadastroSucesso);
router.post('/add-usuario', upload.single('foto_perfil'),
authController.cadastrarUsuario);

router.get('/cadastro-motorista', authController.renderCadastroMotorista);
router.post(
  '/add-motorista',
  upload.fields([
    { name: 'foto_perfil', maxCount: 1 },
    { name: 'carteira_trab', maxCount: 1 },
    { name: 'cursos', maxCount: 1 },
    { name: 'habilitacao', maxCount: 1 },
    { name: 'comprov_resid', maxCount: 1 },
    { name: 'comprov_escola', maxCount: 1 },
    { name: 'titulo_eleitor', maxCount: 1 },
    { name: 'ant_crim', maxCount: 1 },
    { name: 'exame_tox', maxCount: 1 },
  ]),
  authController.cadastrarMotorista
);

module.exports = router;
```

motoristaRoutes.js (rascunho):

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

```
const authController = require('../controllers/motoristaController');

// Configuração do Multer
const storage = multer.diskStorage({
    destination: (req, file, cb) => cb(null, 'public/uploads/'),
    filename: (req, file, cb) => cb(null, Date.now() + ".jpg")
});

const upload = multer({ storage });

router.get('/usuarios/index', motoristaController.renderUsuarios);

module.exports = router;*/
```

adminController.js:

```
const { Usuario, Endereco, Genero, Motorista, Documento, Viagem, Status }
= require('./models');
const { Op } = require('sequelize');

exports.renderUsuarios = async (req, res) => {
  try {
    const posts = await Usuario.findAll({
      include: [
        { model: Endereco },
        { model: Genero }
      ]
    });
    posts.sort((a, b) => b.cod - a.cod);
    res.render('admin/usuarios/index', {
      posts,
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'usuarios'
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao buscar usuários: ' + erro);
  }
};

exports.buscarUsuarios = async (req, res) => {
```

```
  try {
    const termo = req.query.q || '';
    const usuarios = await Usuario.findAll({
      where: {
        nome: {
          [Op.like]: `%${termo}%`
        }
      },
    });
    res.json(usuarios);
  } catch (erro) {
    console.error(erro);
    res.status(500).json({ erro: 'Erro ao buscar usuários' });
  }
};

exports.deletarUsuarios = async (req, res) => {
  try {
    const { cod } = req.params;
    const usuario = await Usuario.findByPk(cod);
    if (!usuario) {
      return res.status(404).send('Usuário não encontrado');
    }
    await usuario.destroy();
    res.redirect('/admin/usuarios/index');
  } catch (error) {
    console.error('Erro ao deletar usuário:', error);
    res.status(500).send('Erro interno no servidor');
  }
};

exports.editarUsuario = async (req, res) => {
  try {
    const cod = req.params.cod;
    const usuario = await Usuario.findOne({
      where: { cod },
      include: [
        { model: Endereco },
        { model: Genero }
      ]
```

```
    });
    if (!usuario) {
      return res.status(404).send('Usuário não encontrado');
    }
    res.render('admin/usuarios/editar', {
      usuario,
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'usuarios'
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao carregar usuário para edição');
  }
};

exports.salvarEdicaoUsuario = async (req, res) => {
  try {
    const cod = req.params.cod;
    const usuario = await Usuario.findByPk(cod);
    if (!usuario) {
      return res.status(404).send('Usuário não encontrado');
    }
    await Endereco.update({
      rua: req.body.rua,
      numero: req.body.numero,
      bairro: req.body.bairro,
      cidade: req.body.cidade,
      UF: req.body.uf,
      CEP: req.body.cep
    }, {
      where: { cod: usuario.enderecoID }
    });
    await Usuario.update({
      img: req.file ? req.file.filename : usuario.img,
      nome: req.body.nome,
      data_nasc: req.body.data_nasc,
      CPF: req.body.CPF,
      generoID: req.body.genero,
      email: req.body.email,
      fone: req.body.fone,
```

```
      SUS: req.body.SUS
    }, {
      where: { cod }
    });
    res.redirect('/admin/usuarios/index');
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao salvar edição: ' + erro);
  }
};

exports.renderMotoristas = async (req, res) => {
  try {
    const motoristas = await Motorista.findAll({
      include: [
        { model: Endereco },
        { model: Genero },
        { model: Documento}
      ]
    });
    motoristas.sort((a, b) => b.cod - a.cod);
    res.render('admin/motoristas/index', {
      posts: motoristas,
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'motoristas'
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao buscar motoristas: ' + erro);
  }
};

exports.buscarMotoristas = async (req, res) => {
  try {
    const termo = req.query.q || '';
    const motoristas = await Motorista.findAll({
      where: {
        nome: {
          [Op.like]: `%${termo}%`
        }
```

```
      },
      include: [
        { model: Endereco },
        { model: Genero }
      ]
    });
    res.json(motoristas);
  } catch (erro) {
    console.error(erro);
    res.status(500).json({ erro: 'Erro ao buscar motoristas' });
  }
};

exports.deletarMotoristas = async (req, res) => {
  try {
    const { cod } = req.params;
    const motorista = await Motorista.findByPk(cod);
    if (!motorista) {
      return res.status(404).send('Motorista não encontrado');
    }
    await motorista.destroy();
    res.redirect('/admin/motoristas/index');
  } catch (error) {
    console.error('Erro ao deletar motorista:', error);
    res.status(500).send('Erro interno no servidor');
  }
};

exports.editarMotorista = async (req, res) => {
  try {
    const cod = req.params.cod;
    const motorista = await Motorista.findOne({
      where: { cod },
      include: [
        { model: Endereco },
        { model: Genero }
      ]
    });
    if (!motorista) {
      return res.status(404).send('Motorista não encontrado');
```

```
    }
    res.render('admin/motoristas/editar', {
      usuario: motorista,
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'motoristas'
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao carregar motorista para edição');
  }
};

exports.salvarEdicaoMotorista = async (req, res) => {
  try {
    const cod = req.params.cod;
    const motorista = await Motorista.findByPk(cod);
    if (!motorista) {
      return res.status(404).send('Motorista não encontrado');
    }

    const cpfVerificando = req.body.CPF.replace(/\D/g, '');
    const foneVerificando = req.body.fone.replace(/\D/g, '');

    await Endereco.update({
      rua: req.body.rua,
      numero: req.body.numero,
      bairro: req.body.bairro,
      cidade: req.body.cidade,
      UF: req.body.uf,
      CEP: req.body.cep
    }, {
      where: { cod: motorista.enderecoID }
    });

    await Motorista.update({
      img: req.file ? req.file.filename : motorista.img,
      nome: req.body.nome,
      data_nasc: req.body.data_nasc,
      CPF: cpfVerificando,
      generoID: req.body.genero,
```

```
      email: req.body.email,
      fone: foneVerificando,
      // docsID e senha não alterados
    }, {
      where: { cod }
    });

    if (req.files) {
      const camposDocumentos = [
        'carteira_trab',
        'cursos',
        'habilitacao',
        'comprov_resid',
        'comprov_escola',
        'titulo_eleitor',
        'ant_crim',
        'exame_tox'
      ];

      const novosDados = {};

      camposDocumentos.forEach((campo) => {
        if (req.files[campo]) {
          novosDados[campo] = req.files[campo][0].filename;
        }
      });

      if (Object.keys(novosDados).length > 0 && motorista.docsID) {
        await Documento.update(novosDados, {
          where: { cod: motorista.docsID }
        });
      }
    }


    res.redirect('/admin/motoristas/index');
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao salvar edição: ' + erro);
  }
```

```
};

exports.renderSolicitacoes = async (req, res) => {
  try {
    res.render('admin/solicitacoes/index', {
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'solicitacoes'
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao carregar solicitações');
  }
};

exports.renderViagens = (req, res) => {
    res.render('admin/viagens/index', {
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'viagens',
    });
};

exports.renderBuscarEventos = async (req, res) => {
    const viagens = await Viagem.findAll();

    const eventos = viagens.map(v => ({
      title: v.destino_cid,
      start: v.data_viagem,
      url: `/admin/viagens/ver-viagem/${v.cod}`
    }));
    res.json(eventos);
}

exports.renderNovaViagem = async (req, res) => {
  try {
    const motoristas = await Motorista.findAll();
    const dataSelecionada = req.query.data_viagem || '';
    res.render('admin/viagens/nova-viagem', {
      layout: 'layouts/layoutAdmin',
      paginaAtual: 'viagens',
      dataSelecionada,
```

```
      motoristas
    });
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao carregar motoristas: ' + erro);
  }
};

exports.renderCadastrarViagem = async (req, res) => {
  try {
    await Viagem.create({
      destino_cid: req.body.destino_cid,
      data_viagem: req.body.data_viagem,
      horario_saida: req.body.horario_saida,
      lugares_dispo: req.body.lugares_dispo,
      modelo_car: req.body.modelo_car,
      placa: req.body.placa,
      motoristaID: req.body.motoristaID,
      statusID: 1
    });
    res.redirect('/admin/viagens/index');
  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao cadastrar viagem: ' + erro);
  }
};

exports.renderVerViagem = async (req, res) => {
    const cod = req.params.cod;
    const viagem = await Viagem.findOne({
      where: { cod },
      include: [
        { model:  Motorista, as: 'Motorista' },
        { model: Status }
      ]
    });

    res.render('admin/viagens/ver-viagem', {
      usuario: viagem,
      layout: 'layouts/layoutAdmin',
```

```
      paginaAtual: 'viagens'
    });
}
```

authController.js:

```
const Usuario = require('../models/Usuario');
const Endereco = require('../models/Endereco');
const Motorista = require('../models/Motorista');
const Documento = require('../models/Documento');

exports.renderEntrada = (req, res) => {
  res.render('auth/entrada', { layout: 'layouts/layoutAuth' });
};

exports.renderCadastro = (req, res) => {
  res.render('auth/cadastro', {
    layout: 'layouts/layoutAuth',
    erroCPF: null,
    erroSUS: null,
    preenchido: {}
  });
};

exports.renderCadastroSucesso = (req, res) => {
  res.render('auth/cadastro-sucesso', { layout: 'layouts/layoutAuth' });
};

exports.cadastrarUsuario = async (req, res) => {
  try {
    const cpfVerificando = req.body.CPF.replace(/\D/g, '');
    const susVerificando = req.body.SUS.replace(/\D/g, '');
    const foneVerificando = req.body.fone.replace(/\D/g, '');

    const cpfExistente = await Usuario.findOne({ where: { CPF:
cpfVerificando } });
    const susExistente = await Usuario.findOne({ where: { SUS:
susVerificando } });

    const erros = {
      erroCPF: null,
```

```
      erroSUS: null,
      erroFone: null,
    };

    if (cpfExistente) erros.erroCPF = 'CPF já cadastrado.';
    else if (cpfVerificando.length !== 11) erros.erroCPF = 'CPF
inválido.';

    if (susExistente) erros.erroSUS = 'Número do SUS já cadastrado.';
    else if (susVerificando.length !== 15) erros.erroSUS = 'Número do SUS
inválido.';

    if (foneVerificando.length < 11) erros.erroFone = 'Telefone inválido.
Deve conter 11 dígitos.';

    // Verifica se tem algum erro com valor (não null, não vazio)
    const temErro = Object.values(erros).some(erro => erro && erro.length
> 0);

    if (temErro) {
      return res.render('auth/cadastro', {
        layout: 'layouts/layoutAuth',
        ...erros,
        preenchido: req.body
      });
    }

    // Criação do endereço
    const enderecoCriado = await Endereco.create({
      rua: req.body.rua,
      numero: req.body.numero,
      bairro: req.body.bairro,
      cidade: req.body.cidade,
      UF: req.body.uf,
      CEP: req.body.cep
    });

    // Criação do usuário
    await Usuario.create({
      img: req.file ? req.file.filename : null,
```

```
      nome: req.body.nome,
      data_nasc: req.body.data_nasc,
      CPF: cpfVerificando,
      generoID: req.body.genero,
      email: req.body.email,
      fone: foneVerificando,
      enderecoID: enderecoCriado.cod,
      SUS: susVerificando,
      senha: req.body.senha
    });

    // Sucesso
    res.redirect('/auth/cadastro-sucesso');

  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao cadastrar: ' + erro);
  }
};

exports.renderCadastroMotorista = (req, res) => {
  res.render('auth/cadastro-motorista', {
    layout: 'layouts/layoutAuth',
    erroCPF: null,
    erroMatricula: null,
    preenchido: {},
    erroFone: null
  });
};

exports.cadastrarMotorista = async (req, res) => {
  try {
    const cpfVerificando = req.body.CPF.replace(/\D/g, '');
    const matriculaVerificando = req.body.matricula.replace(/\D/g, '');
    const foneVerificando = req.body.fone.replace(/\D/g, '');

    const cpfExistente = await Motorista.findOne({ where: { CPF:
cpfVerificando } });
    const matriculaExistente = await Motorista.findOne({ where: {
matricula: matriculaVerificando } });
```

```
    const erros = {
      erroCPF: null,
      erroMatricula: null,
      erroFone: null,
    };

    if (cpfExistente) erros.erroCPF = 'CPF já cadastrado.';
    else if (cpfVerificando.length !== 11) erros.erroCPF = 'CPF
inválido.';

    if (matriculaExistente) erros.erroMatricula = 'Matrícula já
cadastrada.';
    else if (!/^\d{4}$/.test(matriculaVerificando)) erros.erroMatricula =
'Matrícula inválida. Deve conter exatamente 4 dígitos.';

    if (foneVerificando.length < 11) erros.erroFone = 'Telefone inválido.
Deve conter 11 dígitos.';

    const temErro = Object.values(erros).some(erro => erro);

    if (temErro) {
      return res.render('auth/cadastro-motorista', {
        layout: 'layouts/layoutAuth',
        ...erros,
        preenchido: req.body
      });
    }

    const enderecoCriado = await Endereco.create({
      rua: req.body.rua,
      numero: req.body.numero,
      bairro: req.body.bairro,
      cidade: req.body.cidade,
      UF: req.body.uf,
      CEP: req.body.cep
    });

    const arquivos = req.files;
```

```
    const docsCriado = await Documento.create({
      carteira_trab: arquivos?.carteira_trab?.[0]?.filename,
      cursos: arquivos?.cursos?.[0]?.filename,
      habilitacao: arquivos?.habilitacao?.[0]?.filename,
      comprov_resid: arquivos?.comprov_resid?.[0]?.filename,
      comprov_escola: arquivos?.comprov_escola?.[0]?.filename,
      titulo_eleitor: arquivos?.titulo_eleitor?.[0]?.filename,
      ant_crim: arquivos?.ant_crim?.[0]?.filename,
      exame_tox: arquivos?.exame_tox?.[0]?.filename
    });

    await Motorista.create({
      img: req.files['foto_perfil']?.[0].filename || null,
      nome: req.body.nome,
      data_nasc: req.body.data_nasc,
      CPF: cpfVerificando,
      fone: foneVerificando,
      email: req.body.email,
      generoID: req.body.genero,
      enderecoID: enderecoCriado.cod,
      docsID: docsCriado.cod,
      senha: req.body.senha,
      matricula: matriculaVerificando
    });

    res.redirect('/admin/motoristas/index');

  } catch (erro) {
    console.error(erro);
    res.status(500).send('Erro ao cadastrar motorista: ' + erro);
  }
};
```

views/admin/viagens/index.ejs:

```
<div id="calendario"></div>

  <script
src="https://cdn.jsdelivr.net/npm/fullcalendar@6.1.8/index.global.min.js">
</script>
```

```
  <script>
    document.addEventListener('DOMContentLoaded', function () {
      const calendarEl = document.getElementById('calendario');
      if (!calendarEl) return;

      const calendar = new FullCalendar.Calendar(calendarEl, {
        initialView: 'dayGridMonth',
        locale: 'pt-br',
        headerToolbar: {
          left: 'prev,next today',
          center: 'title',
          right: 'dayGridMonth,timeGridWeek,timeGridDay',
        },
        buttonText: {
            today: 'Hoje',
            month: 'Mês',
            week: 'Semana',
            day: 'Dia',
        },
        events: '/admin/viagens/buscar-eventos',

        dateClick: function(info) {
          const dataSelecionada = info.dateStr; // formato: 'YYYY-MM-DD'
          // redireciona para a página de nova viagem, passando a data
como parâmetro
          window.location.href =
`/admin/viagens/nova-viagem?data_viagem=${dataSelecionada}`;
        },

      });

      calendar.render();
    });
  </script>
```

views/admin/viagens/ver-viagem.ejs:

```
<div class="col-12 d-flex justify-content-center espaco-card-edit">
  <div class="card p-3 shadow-sm mt-2" style="width: 100%; max-width:
720px; background-color: #d1e3fd;">
```

```
    <h2 class="mb-4 titulo">
      <%= usuario.destino_cid %> - <%=
usuario.data_viagem.split('-').reverse().join('/') %>
    </h2>

    <p><strong>Horário de saída:</strong> <%= usuario.horario_saida %></p>

    <hr>

    <p><strong>Lugares disponíveis:</strong> <%= usuario.lugares_dispo
%></p>

    <hr>

    <p><strong>Modelo do carro:</strong> <%= usuario.modelo_car %></p>

    <hr>

    <p><strong>Placa:</strong> <%= usuario.placa %></p>

    <hr>

    <p><strong>Motorista:</strong> <%= usuario.Motorista.nome %></p>

    <hr>

    <p><strong>Status:</strong> <%= usuario.status.descricao %></p>

    <hr>
  </div>
</div>
```

views/admin/viagens/nova-viagem.ejs:

```
<div class="col-12 d-flex justify-content-center espaco-card-edit">
  <div class="card p-3 shadow-sm mt-2" style="width: 100%; max-width:
600px; background-color: #ffffff;">

    <form action="/admin/viagens/add-nova-viagem" method="POST"
class="container mt-4">
      <h2 class="mb-4 titulo">Cadastrar Viagem</h2>
```

```
      <div class="mb-3">
        <label for="destino_cid" class="form-label">Destino</label>
        <input type="text" name="destino_cid" class="form-control"
maxlength="30" required placeholder="Digite a cidade de destino">
      </div>

      <div class="mb-3">
        <label for="data_viagem" class="form-label">Data da Viagem</label>
        <input type="date" name="data_viagem" class="form-control"
value="<%= dataSelecionada %>" required placeholder="Selecione a data">
      </div>

      <div class="mb-3">
        <label for="horario_saida" class="form-label">Horário de
Saída</label>
        <input type="time" name="horario_saida" class="form-control"
required placeholder="Escolha o horário de saída">
      </div>

      <div class="mb-3">
        <label for="lugares_dispo" class="form-label">Lugares
Disponíveis</label>
        <input type="number" name="lugares_dispo" class="form-control"
required placeholder="Informe a quantidade de lugares">
      </div>

      <div class="mb-3">
        <label for="modelo_car" class="form-label">Modelo do
Veículo</label>
        <input type="text" name="modelo_car" class="form-control"
maxlength="30" required placeholder="Ex: Corolla, Uno, Civic">
      </div>

      <div class="mb-3">
        <label for="placa" class="form-label">Placa</label>
        <input type="text" name="placa" class="form-control" maxlength="7"
required placeholder="Digite a placa do carro">
      </div>
```

```
      <div class="mb-3">
        <label for="motoristaID" class="form-label">Motorista</label>
        <select name="motoristaID" class="form-control" required>
          <option value="" disabled selected>Selecione um
motorista</option>
          <% motoristas.forEach(motorista => { %>
            <option value="<%= motorista.cod %>"><%= motorista.nome
%></option>
          <% }) %>
        </select>
      </div>

      <div class="d-flex justify-content-between">
        <a href="/admin/viagens/index" class="botao">Cancelar</a>
        <button type="submit" class="botao">Cadastrar</button>
      </div>
    </form>
  </div>
</div>
```

views/admin/usuarios/editar.ejs:

```
<div class="col-12 d-flex justify-content-center espaco-card-edit">
  <div class="card p-3 shadow-sm mt-2" style="width: 100%; max-width:
720px; background-color: #d1e3fd;">
    <form action="/admin/usuarios/editar/<%= usuario.cod %>?_method=PUT"
method="POST" enctype="multipart/form-data" class="form-bold">

      <div class="d-flex flex-row align-items-center mb-3">
        <label for="foto_perfil" class="me-3 ">
          <img id="preview_foto" src="<%= usuario.img ? '/uploads/' +
usuario.img : '/assets/default-profile.jpg' %>"  alt="Foto de perfil"
class="foto-perfil" />
        </label>
        <input type="file" id="foto_perfil" name="foto_perfil"
accept="image/*" class="d-none" onchange="previewImagem(event)">
        <div class="flex-grow-1">
          <div class="d-flex align-items-center">
```

```
            <input type="text" id="nome" name="nome" maxlength="40"
required class="form-control input-transparente" value="<%= usuario.nome
%>" />
          </div>
        </div>
      </div>

      <div class="mb-3 d-flex align-items-center">
        <label for="data_nasc" class="me-2" >Data de nascimento:</label>
        <input type="date" id="data_nasc" name="data_nasc" required
class="form-control input-transparente" style="width: 110px;" value="<%=
usuario.data_nasc %>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="CPF" class="me-2" >CPF:</label>
        <input type="text" id="CPF" name="CPF" maxlength="14" required
class="form-control input-transparente" value="<%=
usuario.CPF.replace(/^(\d{3})(\d{3})(\d{3})(\d{2})$/, "$1.$2.$3-$4")  %>"
/>
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="genero" class="me-2" >Gênero:</label>
        <select id="genero" name="genero" required class="form-select
input-transparente" style="width: auto;">
          <option disabled>Não selecionado</option>
          <option value="1" <%= usuario.generoID == 1 ? 'selected' : ''
%>>Masculino</option>
          <option value="2" <%= usuario.generoID == 2 ? 'selected' : ''
%>>Feminino</option>
          <option value="3" <%= usuario.generoID == 3 ? 'selected' : ''
%>>Não-binário</option>
          <option value="4" <%= usuario.generoID == 4 ? 'selected' : ''
%>>Outro</option>
        </select>
```

```
      </div>

      <hr>

      <!-- Campo: Email -->
      <div class="mb-3 d-flex align-items-center">
        <label for="email" class="me-2" >Email:</label>
        <input type="email" id="email" name="email" maxlength="30"
required class="form-control input-transparente"  value="<%= usuario.email
%>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="fone" class="me-2" >Telefone:</label>
        <input type="text" id="fone" name="fone" maxlength="16"
class="form-control input-transparente"  value="<%=
usuario.fone.replace(/^(\d{2})(\d{5})(\d{4})$/, "($1) $2-$3") %>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="rua" class="me-2" >Rua:</label>
        <input type="text" id="rua" name="rua" required
class="form-control input-transparente" style="max-width: 300px;"
value="<%= usuario.endereco?.rua || '' %>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="numero" class="me-2" >Número:</label>
        <input type="text" id="numero" name="numero" required
class="form-control input-transparente"  value="<%=
usuario.endereco?.numero || '' %>" />
      </div>

      <hr>
```

```
      <div class="mb-3 d-flex align-items-center">
        <label for="bairro" class="me-2" >Bairro:</label>
        <input type="text" id="bairro" name="bairro" required
class="form-control input-transparente"  value="<%=
usuario.endereco?.bairro || '' %>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="cidade" class="me-2" >Cidade:</label>
        <input type="text" id="cidade" name="cidade" required
class="form-control input-transparente" value="<%=
usuario.endereco?.cidade || '' %>" />
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="uf" class="me-2" >Estado (UF):</label>
        <select id="uf" name="uf" required class="form-select
input-transparente" style="width: 60px;">
          <option disabled>Não selecionado</option>
          <% ['SC', 'PR', 'RS', 'SP', 'RJ', 'MG', 'ES', 'DF', 'GO', 'MT',
'MS', 'AL', 'BA', 'CE', 'MA', 'PB', 'PE', 'PI', 'RN', 'SE', 'AC', 'AP',
'AM', 'PA', 'RO', 'RR', 'TO'].forEach(estado => { %>
            <option value="<%= estado %>" <%= usuario.endereco?.UF ===
estado ? 'selected' : '' %>><%= estado %></option>
          <% }) %>
        </select>
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="cep" class="me-2">CEP:</label>
        <input type="text" id="cep" name="cep" required
class="form-control input-transparente" value="<%=
usuario.endereco.CEP.replace(/^(\d{5})(\d{3})$/, '$1-$2') %>" />
```

```
      </div>

      <hr>

      <div class="mb-3 d-flex align-items-center">
        <label for="SUS" class="me-2">SUS:</label>
        <input type="text" id="SUS" name="SUS" maxlength="15" required
class="form-control input-transparente" value="<%= usuario.SUS%>" />
      </div>

      <hr>

      <div class="d-flex justify-content-between">
        <a href="/admin/usuarios/index" class="botao">Cancelar</a>
        <button type="submit" class="botao">Salvar</button>
      </div>
    </form>
  </div>
</div>


<script>
  function previewImagem(event) {
  const input = event.target;
  const reader = new FileReader();

  reader.onload = function () {
    const preview = document.getElementById('preview_foto');
    preview.src = reader.result;
  };

  reader.readAsDataURL(input.files[0]);
}

 document.querySelector('form').addEventListener('submit', function (e) {
    const cpfInput = document.getElementById('CPF');
    const foneInput = document.getElementById('fone');
    const cepInput = document.getElementById('cep');

    cpfInput.value = cpfInput.value.replace(/\D/g, '');
```

```
    foneInput.value = foneInput.value.replace(/\D/g, '');
    cepInput.value = cepInput.value.replace(/\D/g, '');
  });

</script>
```

views/admin/usuarios/index.ejs:

```
<div class="container mt-4">
  <div class="row justify-content-center mb-5">
    <div class="d-flex justify-content-center align-items-center"
style="gap: 10px;">
      <input type="search"
      id="pesquisa"
      placeholder="Pesquisar usuário por nome ou CPF"
      class="form-control pesquisa-custom mb-3">
      <a href="/auth/cadastro" class="botao-cadastro">Cadastrar
Usuário</a>
    </div>

    <% posts.forEach(function(post) { %>

      <div class="col-12 d-flex justify-content-center">
        <div class="card p-3 shadow-sm mt-2" style="width: 100%;
max-width: 720px; background-color: #d1e3fd;">
          <div class="d-flex flex-row align-items-center">

            <img
              src="<%= post.img ? '/uploads/' + post.img :
'/assets/default-profile.jpg' %>"
              alt="Foto de perfil"
              class="foto-perfil me-3">

            <div class="flex-grow-1">
              <h5 class="nome-perfil mb-0"><%= post.nome %></h5>
            </div>

            <button class="botao ver-mais-btn">Ver mais</button>
            </div>
```

```
            <!--VER MAIS-->
            <div class="detalhes">
            <p><strong>Data de Nascimento: </strong><%=
post.data_nasc.split('-').reverse().join('/') %></p>
            <hr>
            <p><strong>CPF: </strong><span class="cpf-perfil"><%=
post.CPF.replace(/^(\d{3})(\d{3})(\d{3})(\d{2})$/, "$1.$2.$3-$4")
%></span></p>
            <hr>
            <p><strong>Gênero: </strong> <%= post.genero.descricao %></p>
            <hr>
            <p><strong>Email: </strong> <%= post.email %></p>
            <hr>
            <p><strong>Telefone: </strong><%=
post.fone.replace(/^(\d{2})(\d{5})(\d{4})$/, "($1) $2-$3")
            %></p>
            <hr>
            <p><strong>Endereço: </strong>
              <%= post.endereco?.rua || '' %>,
              <%= post.endereco?.numero || '' %> -
              <%= post.endereco?.bairro || '' %> -
              <%= post.endereco?.cidade || '' %>
              (<%= post.endereco?.UF || '' %>)
              CEP: <%= post.endereco?.CEP
                          ? post.endereco.CEP.replace(/^(\d{5})(\d{3})$/,
"$1-$2")
                          : ''
                    %>
            </p>
            <hr>
            <p><strong>SUS: </strong> <%= post.SUS %></p>
            <hr>
<div class="d-flex justify-content-end gap-3">
  <a href="/admin/usuarios/editar/<%= post.cod %>"
class="botao">Editar</a>
  <form action="/admin/usuarios/deletar/<%= post.cod %>?_method=DELETE"
method="POST">
    <button type="submit" class="botao bg-vermelho" onclick="return
confirm('Tem certeza que deseja deletar este usuário?')">
      Deletar
```

```
    </button>
  </form>
</div>

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

    <% }) %>

  </div>
</div>

<script>
  const inputPesquisa = document.getElementById('pesquisa');
  const cards = Array.from(document.querySelectorAll('.card')); //
colocando os cards dentro de uma array
  const row = document.querySelector('.row');


  cards.forEach((card, index) => {
    card.setAttribute('data-original-index', index);            // salva o
index(posição) original dos cards
  });

  inputPesquisa.addEventListener('input', function () {
    const termo = this.value.trim().toLowerCase();              //
padronizando: trim = remove espaços toLowerCase = deixa tudo minusculo

    if (termo.length === 0) {                                   // se a
barra de pesquisa ta vazia mostra todos os cards
      cards.forEach(card => {
        card.style.display = 'block';
        card.setAttribute('data-relevancia', '0');              // tira a
relevancia
      });

   const fragment = document.createDocumentFragment();

      cards
```

```
        .sort((a, b) => a.getAttribute('data-original-index') -
b.getAttribute('data-original-index'))    // .sort serve pra reordenar a
array
        .forEach(card => fragment.appendChild(card));           // pega os
cards e coloca dentro do fragment

      row.appendChild(fragment);                                //
colocando o fragment na row

    } else {
      cards.forEach(card => {
        const nomeELemento = card.querySelector('.nome-perfil');
        const nome = nomeELemento ? nomeELemento.textContent.toLowerCase()
: '';  // se o nomeELemento existe transforma em lowercase e guarda no
nome se não guarda ' '

        const cpfElemento = card.querySelector('.cpf-perfil');
        const cpf = cpfElemento ?
cpfElemento.textContent.replace(/[.\-]/g, '') : '';

        if (nome.startsWith(termo) || cpf.startsWith(termo)) {
          card.style.display = 'block';
          card.setAttribute('data-relevancia', '1');
        } else if (nome.includes(termo) || cpf.includes(termo)) {
          card.style.display = 'block';
          card.setAttribute('data-relevancia', '2');
        } else {
          card.style.display = 'none';
          card.setAttribute('data-relevancia', '99');
        }

      });

      const fragment = document.createDocumentFragment();

      cards
        .filter(card => card.style.display === 'block')
        .sort((a, b) => a.getAttribute('data-relevancia') -
b.getAttribute('data-relevancia'))
        .forEach(card => fragment.appendChild(card));
```

```
      row.appendChild(fragment);
    }
  });

  // VER MAIS / VER MENOS
  document.querySelectorAll('.ver-mais-btn').forEach(btn => {
    btn.addEventListener('click', function () {
      const card = this.closest('.card');
      const detalhes = card.querySelector('.detalhes');
      const estaExpandido = detalhes.classList.contains('expandido');

      if (estaExpandido) {
        detalhes.style.height = '0px';
        detalhes.style.marginTop = '0';
        detalhes.classList.remove('expandido');
        this.textContent = 'Ver mais';
      } else {
        detalhes.style.height = detalhes.scrollHeight + 'px';
        detalhes.style.marginTop = '16px';
        detalhes.classList.add('expandido');
        this.textContent = 'Ver menos';
      }
    });
  });
</script>
```