UNIVERSIDADE ESTÁCIO DE SÁ
Polo Nova Ourinhos - Ourinhos/SP
Tecnologia em Desenvolvimento Full Stack
RPG0035 - Software sem Segurança não Serve

Aluno: Rogério Benedito Geraldo Matrícula: 202110027921.

Github: https://github.com/Rogeriobg/MPN5M5-RPG0035.git

RELATÓRIO DA MISSÃO PRÁTICA

1- Controle básico de acesso a uma API Rest

A autenticação é feita com JWT. O login gera um token válido por uma hora. Rotas protegidas usam middleware para validar o token. Há uma verificação adicional para garantir que apenas administradores acessem rotas específicas.

2- Tratamento de dados sensíveis e log de erros com foco em segurança

As senhas são armazenadas de forma segura usando bcrypt com hash. Durante o login, a senha informada é comparada com o hash. O código evita exibir informações sensíveis. Não há sistema de log robusto, mas o tratamento de senha é adequado.

3- Prevenção de ataques com tokens desprotegidos ou desatualizados

O sistema rejeita requisições sem token ou com token inválido. Tokens têm tempo de expiração e só são aceitos se forem válidos. Isso evita o uso de tokens antigos ou comprometidos.

4- Tratamento de SQL Injection no código-fonte

As entradas passadas para consultas são sanitizadas por uma função que remove caracteres perigosos. Isso reduz o risco de comandos maliciosos. Idealmente, seria melhor usar prepared statements no lugar de strings concatenadas.

5- Tratamento de CRLF Injection

A sanitização de entradas também evita injeções de quebras de linha que poderiam alterar a estrutura de cabeçalhos HTTP ou gerar logs falsos. A função remove caracteres que poderiam causar esse tipo de ataque.

6- Prevenção a ataques CSRF em sistemas web

Como a autenticação usa JWT no cabeçalho Authorization, o navegador não envia automaticamente o token. Isso impede que requisições maliciosas sejam executadas sem o conhecimento do usuário. O uso de token fora de cookies mitiga ataques CSRF.

7- CÓDIGO UTILIZADO

```
const express = require('express');
const bodyParser = require('body-parser');
const cors = require('cors');
const jwt = require('jsonwebtoken');
const bcrypt = require('bcrypt');
const { body, validationResult } = require('express-validator');

const app = express();
const port = process.env.PORT || 3000;
const SECRET_KEY = 'chave_super_secreta';

app.use(cors());
app.use(bodyParser.json());



const users = [
  {
    username: "user",
    password: bcrypt.hashSync("123456", 10),
    id: 123,
    email: "user@dominio.com",
    perfil: "user"
  },
  {
    username: "admin",
    password: bcrypt.hashSync("123456789", 10),
    id: 124,
    email: "admin@dominio.com",
    perfil: "admin"
  },
  {
    username: "colab",
    password: bcrypt.hashSync("123", 10),
    id: 125,
    email: "colab@dominio.com",
    perfil: "user"
  }
];


function sanitize(input) {
  if (typeof input !== 'string') return '';
  return input.replace(/[^a-zA-Z0-9-_@.]/g, '');
}
```

```
function doLogin(credentials) {
  return users.find(u => u.username === credentials.username);
}


function generateToken(user) {
  const payload = {
    id: user.id,
    perfil: user.perfil,
    email: user.email
  };
  return jwt.sign(payload, SECRET_KEY, { expiresIn: '1h' });
}


function authenticateToken(req, res, next) {
  const authHeader = req.headers['authorization'];
  const token = authHeader?.split(' ')[1];
  if (!token) return res.status(401).json({ message: 'Token não fornecido'
});

  jwt.verify(token, SECRET_KEY, (err, user) => {
    if (err) return res.status(403).json({ message: 'Token inválido' });
    req.user = user;
    next();
  });
}


function onlyAdmin(req, res, next) {
  if (req.user.perfil !== 'admin') {
    return res.status(403).json({ message: 'Acesso restrito a
administradores' });
  }
  next();
}


class Repository {
  execute(query) {
    console.log("Query executada (fake):", query);

    return [
      {
        contrato_id: 1,
        empresa: 'AcmeCorp',
        data_inicio: '2024-01-01',
        descricao: 'Contrato de prestação de serviços'
      }
    ];
  }
}
```

```
function getContracts(empresa, inicio) {
  const sanitizedEmpresa = sanitize(empresa);
  const sanitizedInicio = sanitize(inicio);
  const query = `SELECT * FROM contracts WHERE empresa =
'${sanitizedEmpresa}' AND data_inicio = '${sanitizedInicio}'`;
  const repository = new Repository();
  return repository.execute(query);
}


app.post('/api/auth/login',
  body('username').notEmpty().withMessage('Usuário obrigatório'),
  body('password').notEmpty().withMessage('Senha obrigatória'),
  (req, res) => {
    const errors = validationResult(req);
    if (!errors.isEmpty()) return res.status(400).json({ errors:
errors.array() });

    const credentials = req.body;
    const user = doLogin(credentials);
    if (!user) return res.status(401).json({ message: 'Credenciais inválidas'
});

    const passwordValid = bcrypt.compareSync(credentials.password,
user.password);
    if (!passwordValid) return res.status(401).json({ message: 'Credenciais
inválidas' });

    const token = generateToken(user);
    res.json({ token });
  }
);


app.get('/api/auth/profile', authenticateToken, (req, res) => {
  res.status(200).json({ user: req.user });
});


app.get('/api/users', authenticateToken, onlyAdmin, (req, res) => {
  res.status(200).json({ data: users });
});


app.get('/api/contracts/:empresa/:inicio', authenticateToken, onlyAdmin,
(req, res) => {
  const { empresa, inicio } = req.params;
  const result = getContracts(empresa, inicio);
  if (result.length > 0)
    res.status(200).json({ data: result });
```

```
  else
    res.status(404).json({ message: 'Dados não encontrados' });
});


app.listen(port, () => {
  console.log(`Servidor rodando na porta ${port}`);
});
```

CÓDIGO DO FRONTEND (Apenas para testes da api)

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <title>Software sem segurança não serve</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css
" rel="stylesheet">
  <style>
    body {
      padding: 2rem;
    }
    .token-info {
      word-break: break-all;
    }
  </style>
</head>
<body>
  <div class="container">
   <h1 class="mb-4 text-center">Software Sem Segurança Não Serve</h1>


    <div class="card mb-4">
      <div class="card-header">Login</div>
      <div class="card-body">
        <form id="loginForm">
          <div class="mb-3">
            <label for="username" class="form-label">Usuário</label>
            <input type="text" class="form-control" id="username" required
value="admin">
          </div>
          <div class="mb-3">
            <label for="password" class="form-label">Senha</label>
            <input type="password" class="form-control" id="password"
required value="123456789">
          </div>
          <button type="submit" class="btn btn-primary">Login</button>
        </form>
        <div class="mt-3">
          <strong>Token:</strong>
          <pre class="token-info" id="tokenDisplay"></pre>
        </div>
      </div>
    </div>


    <div class="card mb-4">
      <div class="card-header">Perfil do Usuário (GET
/api/auth/profile)</div>
```

```
      <div class="card-body">
        <button class="btn btn-success" onclick="getProfile()">Ver
Perfil</button>
        <pre id="profileResult" class="mt-3"></pre>
      </div>
    </div>


    <div class="card mb-4">
      <div class="card-header">Lista de Usuários (somente admin)</div>
      <div class="card-body">
        <button class="btn btn-info" onclick="getUsers()">Ver
Usuários</button>
        <pre id="usersResult" class="mt-3"></pre>
      </div>
    </div>


    <div class="card mb-4">
      <div class="card-header">Consultar Contratos (somente admin)</div>
      <div class="card-body">
        <form id="contractForm">
          <div class="mb-3">
            <label for="empresa" class="form-label">Empresa</label>
            <input type="text" class="form-control" id="empresa"
value="AcmeCorp" required>
          </div>
          <div class="mb-3">
            <label for="inicio" class="form-label">Data de Início</label>
            <input type="text" class="form-control" id="inicio" value="2024-
01-01" required>
          </div>
          <button type="submit" class="btn btn-warning">Buscar
Contratos</button>
        </form>
        <pre id="contractsResult" class="mt-3"></pre>
      </div>
    </div>
  </div>

  <script>
    let token = '';


    document.getElementById('loginForm').addEventListener('submit', async (e)
=> {
      e.preventDefault();
      const username = document.getElementById('username').value;
      const password = document.getElementById('password').value;

      const res = await fetch('http://localhost:3000/api/auth/login', {
```

```
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ username, password })
      });

      const data = await res.json();
      if (data.token) {
        token = data.token;
        document.getElementById('tokenDisplay').textContent = data.token;
        alert('Login bem-sucedido!');
      } else {
        alert(data.message || 'Erro no login');
      }
    });


    async function getProfile() {
      const res = await fetch('http://localhost:3000/api/auth/profile', {
        headers: { 'Authorization': 'Bearer ' + token }
      });
      const data = await res.json();
      document.getElementById('profileResult').textContent =
JSON.stringify(data, null, 2);
    }


    async function getUsers() {
      const res = await fetch('http://localhost:3000/api/users', {
        headers: { 'Authorization': 'Bearer ' + token }
      });
      const data = await res.json();
      document.getElementById('usersResult').textContent =
JSON.stringify(data, null, 2);
    }


    document.getElementById('contractForm').addEventListener('submit', async
(e) => {
      e.preventDefault();
      const empresa = document.getElementById('empresa').value;
      const inicio = document.getElementById('inicio').value;

      const res = await
fetch(`http://localhost:3000/api/contracts/${empresa}/${inicio}`, {
        headers: { 'Authorization': 'Bearer ' + token }
      });

      const data = await res.json();
      document.getElementById('contractsResult').textContent =
JSON.stringify(data, null, 2);
    });
```

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