Missão Prática

Nível – 5

Tema: Software Sem Segurança Nao Serve

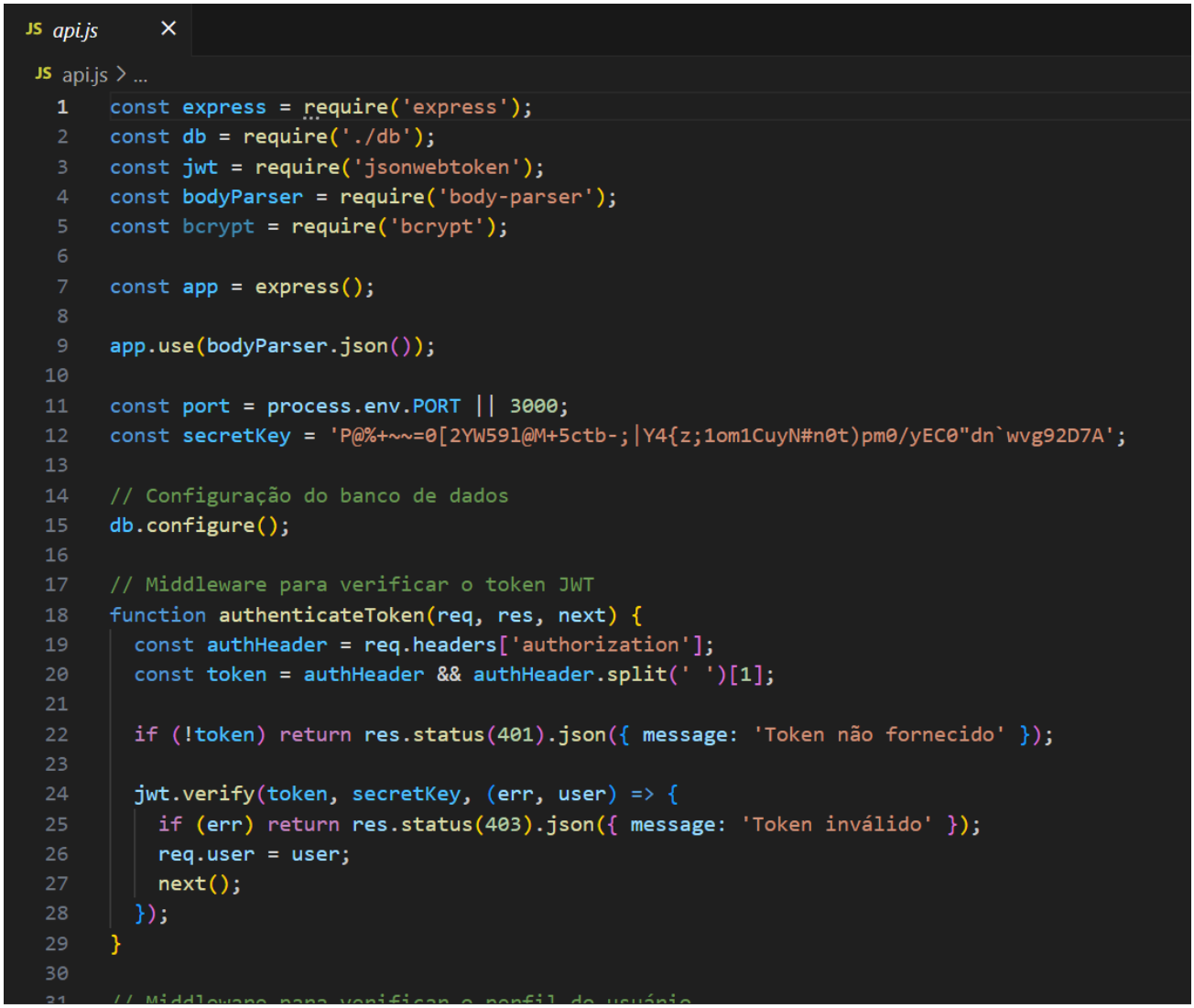
Aluno: Yvo Murilo Santos D’Albuquerque

Objetivos da prática

- Descrever o controle básico de acesso a uma API Rest;
- Descrever o tratamento de dados sensíveis e log de erros com foco em segurança;
- Descrever a prevenção de ataques de acesso não autorizado com base em tokens
- desprotegidos/desatualizados;
- Descrever o tratamento de SQL Injection em códigos-fonte; Descrever o tratamento
- de CRLF Injection em códigos-fonte;

- Descrever a prevenção a ataques do tipo CSRF em sistemas web;

Códigos



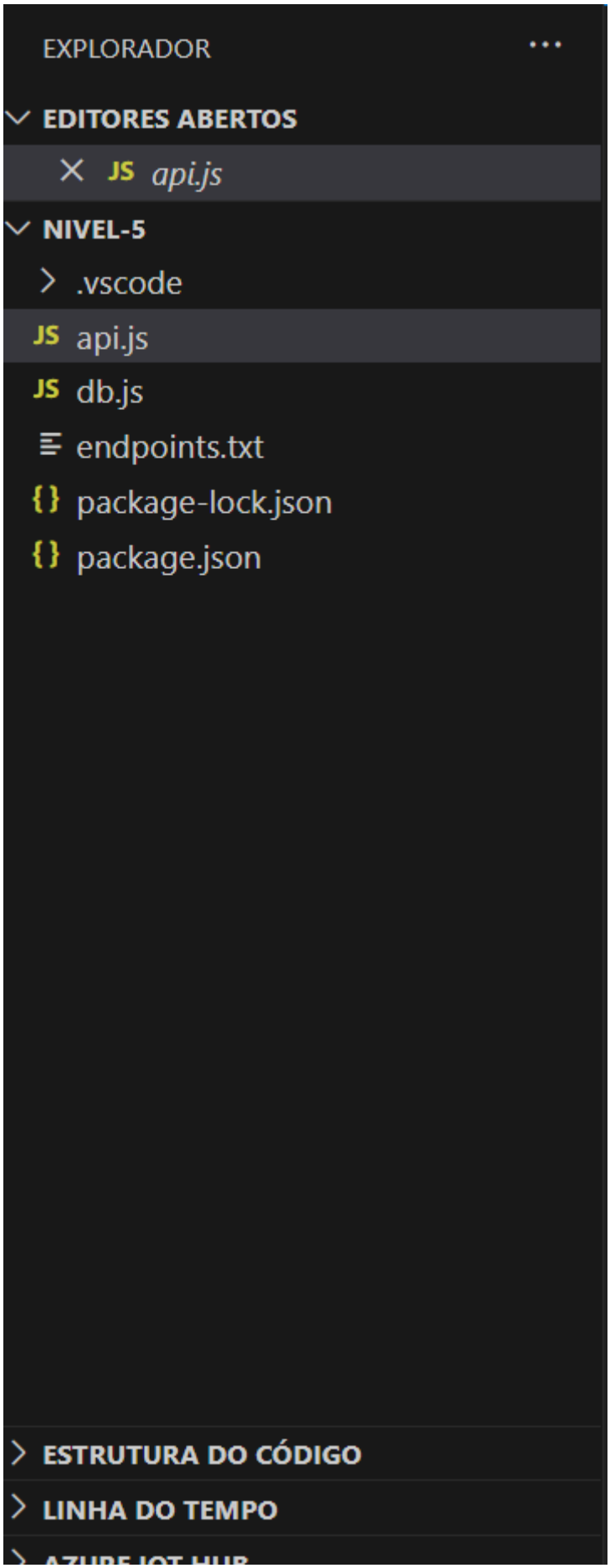
```
JS api.js
JS api.js > ...
1   const express = require('express');
2   const db = require('./db');
3   const jwt = require('jsonwebtoken');
4   const bodyParser = require('body-parser');
5   const bcrypt = require('bcrypt');
6
7   const app = express();
8
9   app.use(bodyParser.json());
10
11  const port = process.env.PORT || 3000;
12  const secretKey = 'P@%+~~=0[2YW59l@M+5ctb-;|Y4{z;1om1CuyN#n0t)pm0/yEC0"dn`wvg92D7A';
13
14  // Configuração do banco de dados
15  db.configure();
16
17  // Middleware para verificar o token JWT
18  function authenticateToken(req, res, next) {
19    const authHeader = req.headers['authorization'];
20    const token = authHeader && authHeader.split(' ')[1];
21
22    if (!token) return res.status(401).json({ message: 'Token não fornecido' });
23
24    jwt.verify(token, secretKey, (err, user) => {
25      if (err) return res.status(403).json({ message: 'Token inválido' });
26      req.user = user;
27      next();
28    });
29  }
30
31  // Middleware para verificar o perfil do usuário
```

db.js

```js
const sqlite3 = require('sqlite3').verbose();

const db = new sqlite3.Database(':memory:');

db.serialize(() => {
  db.run(`CREATE TABLE contracts (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    empresa TEXT,
    data_inicio TEXT
  )`);

  db.run(`CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    username TEXT,
    password TEXT,
    email TEXT,
    perfil TEXT
  )`);

  // Inserção de dados de exemplo
  db.run(`INSERT INTO contracts (empresa, data_inicio) VALUES ('empresa1', '2023-01-01')`);
  db.run(`INSERT INTO contracts (empresa, data_inicio) VALUES ('empresa2', '2023-02-01')`);

  db.run(`INSERT INTO users (username, password, email, perfil) VALUES ('user', '123456', 'user@dominio.com', 'user')`);
  db.run(`INSERT INTO users (username, password, email, perfil) VALUES ('admin', '123456789', 'admin@dominio.com', 'admin')`);
  db.run(`INSERT INTO users (username, password, email, perfil) VALUES ('colab', '123', 'colab@dominio.com', 'user')`);
});

module.exports = db;
```

endpoints.txt

```
Endpoints Disponíveis:
POST /api/auth/register - Registrar um novo usuário
POST /api/auth/login - Autenticar um usuário e obter um token JWT
GET /api/profile - Obter os dados do usuário autenticado
GET /api/users - Obter todos os usuários (apenas para administradores)
POST /api/contracts - Criar um novo contrato (apenas para administradores)
GET /api/contracts - Obter contratos existentes (apenas para administradores)
```


EXPLORADOR
EDITORES ABERTOS
api.js
NIVEL-5
.vscode
api.js
db.js
endpoints.txt
package-lock.json
package.json
ESTRUTURA DO CÓDIGO
LINHA DO TEMPO