Campus: Santa Cruz da Serra - Duque de Caxias/RJ
Curso: Desenvolvimento full stack
Disciplina: RPG0035 - SOFTWARE SEM SEGURANÇA NÃO SERVE!
Número da Turma: 9001
Semestre letivo: Mundo 5
Integrantes: Felipe Freaza Fidalgo

SOFTWARE SEM SEGURANÇA NÃO SERVE! (JSON Web Tokens)

Visão Geral

Este projeto tem como objetivo refatorar uma API para melhorar sua segurança. Originalmente, a API apresentava vulnerabilidades críticas, como criptografia insegura para identificação de sessões e falta de controle de acesso adequado. Através da refatoração, buscamos implementar boas práticas de segurança e garantir que a aplicação esteja protegida contra ataques comuns, como força bruta e injeção de código.

Contexto do Problema

A aplicação web existente possuía uma API que utilizava um sistema de geração de identificadores de sessão (`session-id`) baseado em criptografia simples. Esse método não só era suscetível a ataques de força bruta, mas também usava uma chave de criptografia fraca, que era o próprio nome da empresa. Além disso, a API não realizava validações adequadas nos parâmetros recebidos, o que abria portas para ataques de injeção.

Componentes do Projeto

1. Servidor Express

O projeto utiliza o framework **Express.js** para gerenciar o servidor e criar rotas para a API. O servidor é configurado para escutar em uma porta específica e usa o **body-parser** para interpretar os dados das requisições.

2. Autenticação e Criptografia

Originalmente, o código gerava um `session-id` usando uma criptografia simples com a chave derivada do nome da empresa. Após a refatoração, substituímos essa abordagem por tokens JWT (JSON Web Tokens), que são mais seguros e contêm informações criptografadas sobre o usuário e a validade do token.

3. Endpoints da API

Os principais endpoints da API foram modificados para melhorar a segurança:

- **Login (`/api/auth/login`)**: Recebe as credenciais do usuário, valida o login e gera um token JWT. O token substitui o `session-id` inseguro.
- **Recuperação de Usuários (`/api/users`)**: Agora protegido por autenticação JWT, este endpoint só pode ser acessado por usuários com o perfil 'admin'. Antes da refatoração, qualquer pessoa com um `session-id` válido poderia acessar este recurso.

4. Validação e Controle de Acesso

Após a refatoração, implementamos validações rigorosas para o token JWT em cada requisição. Isso garante que apenas usuários autenticados e autorizados possam acessar recursos sensíveis. A validação inclui a verificação da expiração do token e a identidade do usuário.

Funcionalidades e Métodos

Função de Login

A função '`doLogin`' valida as credenciais fornecidas e, se corretas, gera um token JWT que é retornado ao cliente. Este token é então usado para autenticação em futuras requisições.

Função de Autenticação de Token

O middleware `authenticateToken` verifica a presença e validade do token JWT nas requisições. Se o token for válido, ele adiciona as informações do usuário à requisição, permitindo o acesso aos recursos protegidos.

Controle de Acesso

Adicionamos controle de acesso baseado em perfil para proteger endpoints críticos. Apenas usuários com o perfil 'admin' podem acessar informações sensíveis, como dados de todos os usuários e contratos.

Sanitização de Parâmetros

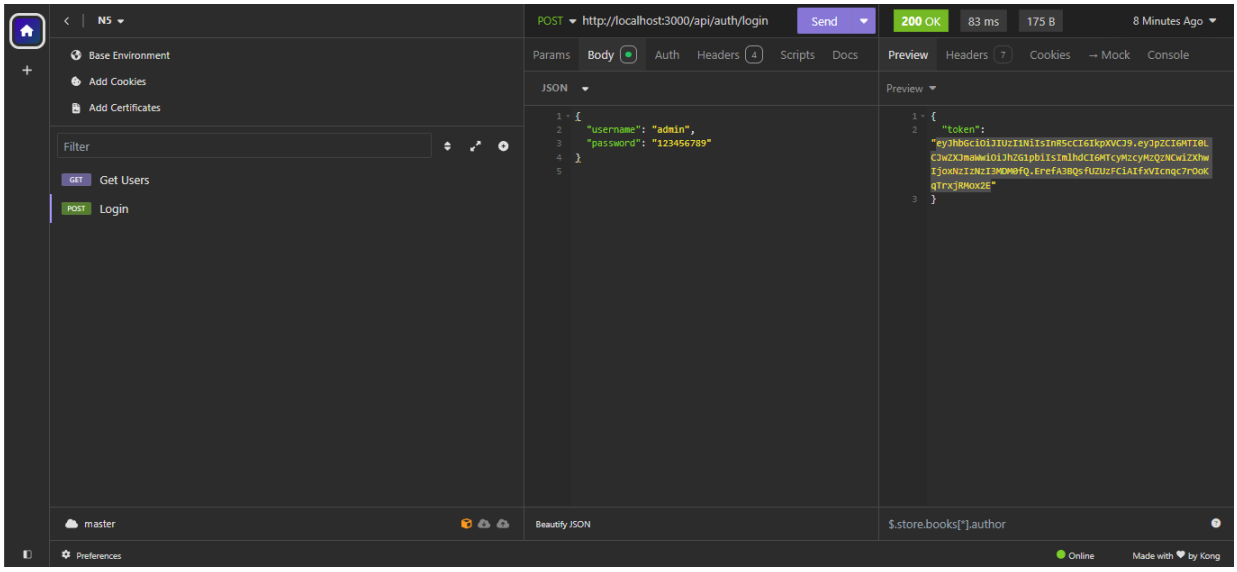
Refatoramos a função `getContracts` para garantir que os parâmetros recebidos sejam sanitizados, prevenindo ataques de injeção de código. A sanitização é realizada para remover caracteres potencialmente perigosos dos parâmetros de entrada.

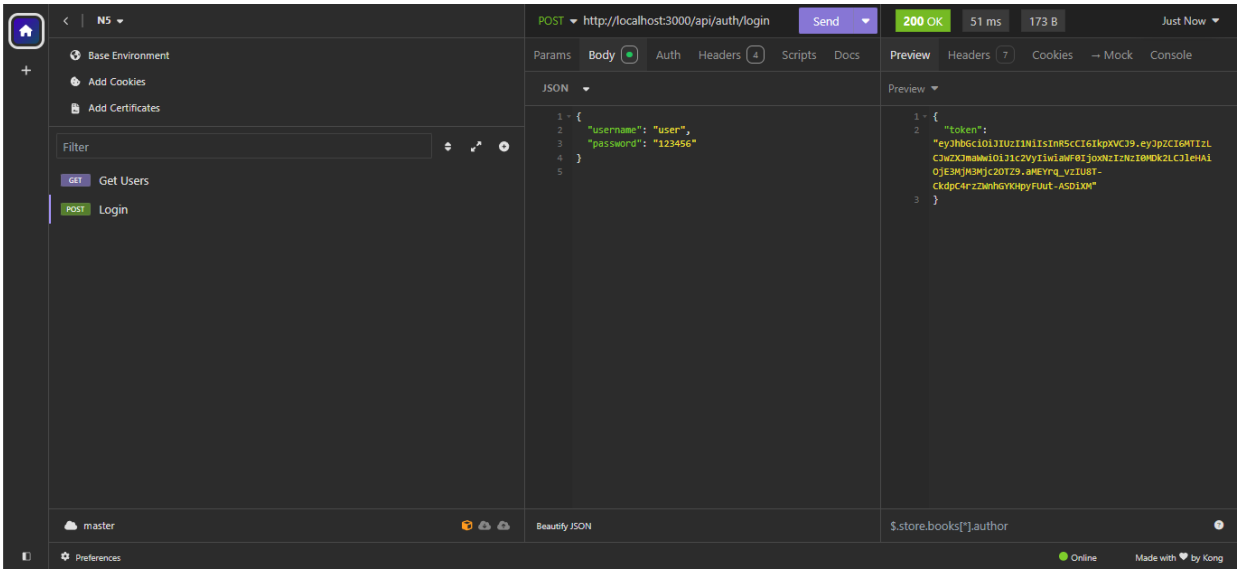
Testes e Imagens

Para validar as mudanças e garantir que todas as medidas de segurança estejam funcionando conforme esperado, foram realizados testes abrangentes. As imagens a seguir ilustram os testes realizados em cada endpoint da API. Estes testes foram executados utilizando a ferramenta Insomnia.

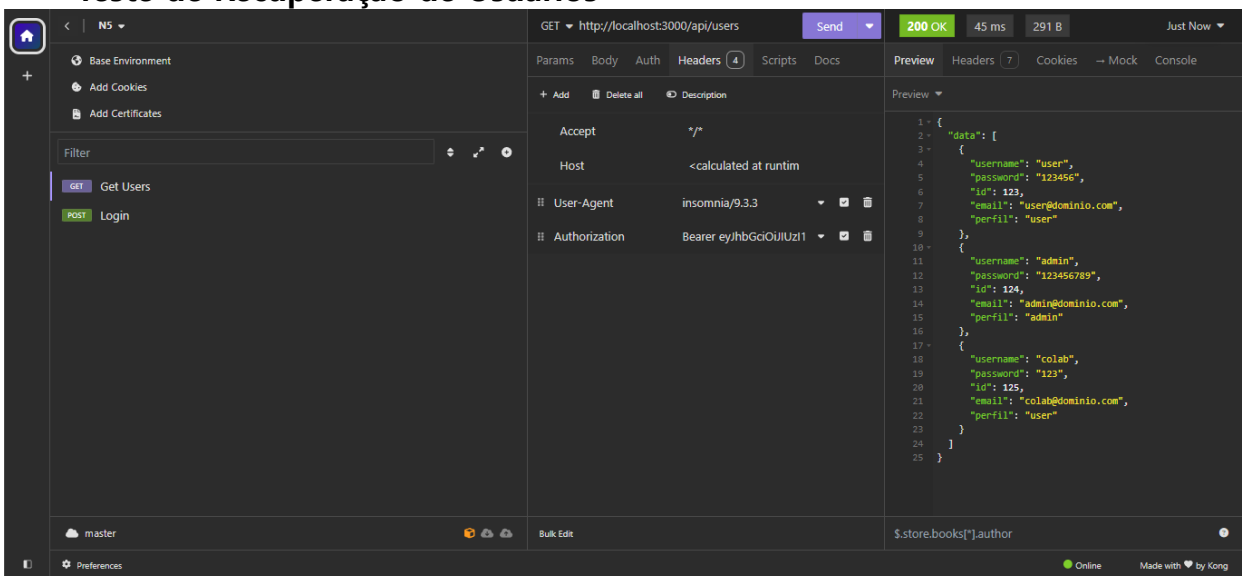
Imagens de Teste

- **Teste de Login**

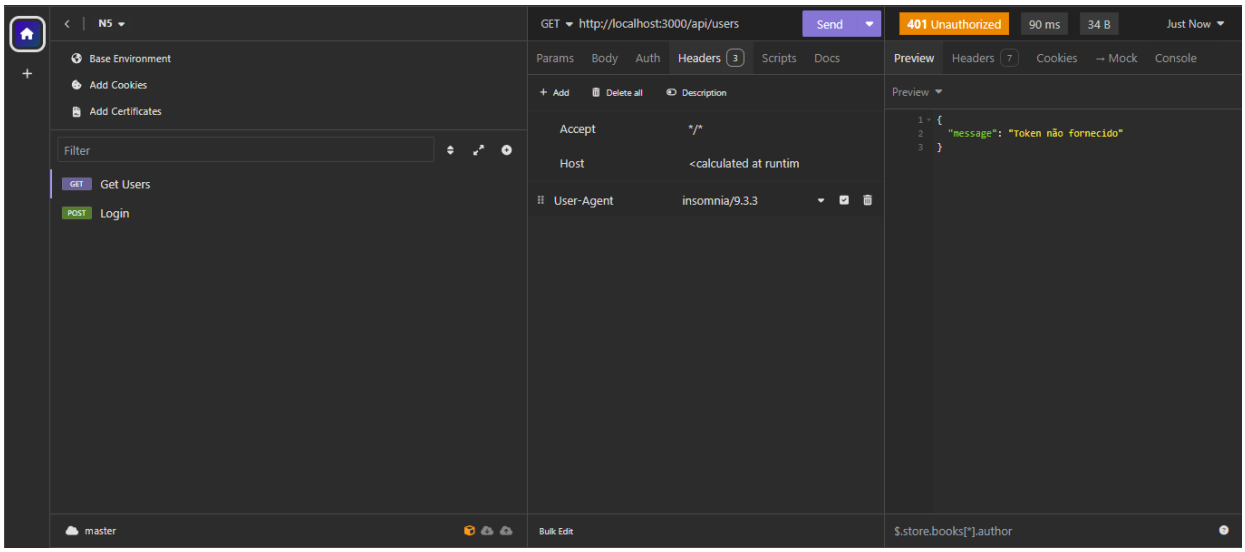


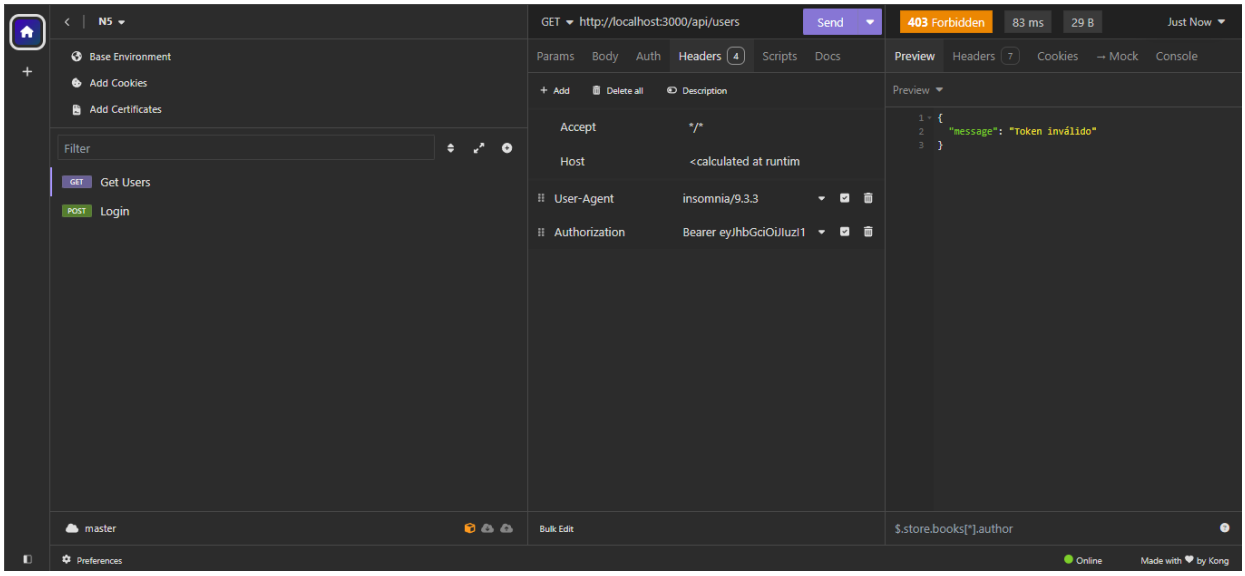


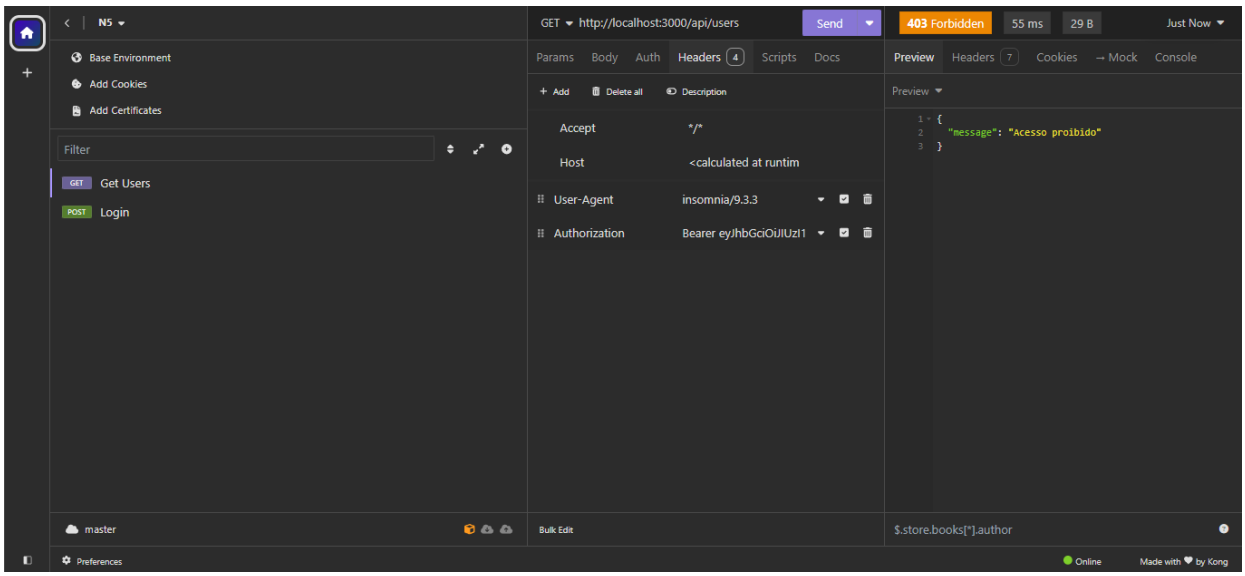
- **Teste de Recuperação de Usuários**



- **Teste de Controle de Acesso**







As imagens acima mostram os resultados dos testes, incluindo requisições e respostas da API. Elas comprovam que os endpoints estão funcionando corretamente e que as medidas de segurança foram efetivamente implementadas.

Conclusão

A refatoração da API melhorou significativamente sua segurança, abordando vulnerabilidades críticas e implementando melhores práticas de autenticação e controle de acesso. A transição para tokens JWT fortaleceu a segurança da autenticação, e a sanitização dos parâmetros de entrada protege contra injeções de código. Com essas mudanças, a aplicação está agora melhor preparada para resistir a ataques e garantir a integridade e a confidencialidade dos dados.

Esta prática não só reforça o conhecimento sobre segurança em aplicações web, mas também ilustra a importância de aplicar medidas corretivas para proteger sistemas contra ameaças comuns.