Desenvolvimento de Software Multi Plataforma

Disciplina: Banco de Dados Não Relacional

Prof. Ms. Ricardo Leme

Peso: 25% da média final*

Entrega: 16/06/2025 até às 19h00

Apresentação para o professor: 16/06/2025

A ausência injustificada de um dos discentes no dia da apresentação, resultará na nota ZERO para o ausente.

Trabalho - Implementação de Autenticação JWT e Integração de UI com APIs RESTful

Neste projeto, você terá a tarefa de criar um sistema de autenticação JWT (JSON Web Tokens) e integrar uma interface de usuário (UI) com as APIs RESTful que foram desenvolvidas na entrega anterior. O objetivo é garantir que a UI seja capaz de consumir as APIs e exibir as mensagens geradas pela API de forma autenticada.

Requisitos do Projeto:

1. **Implementação do JWT:** Crie um sistema de autenticação JWT para proteger as rotas da sua API. Os usuários devem poder se autenticar, receber um token JWT válido e usá-lo para acessar as rotas protegidas.

2. **Criação da UI:** Efetue os ajustes apontados na correção anterior e permita aos usuários se cadastrarem, autenticarem e interagir com as operações da API. (incluir, editar, listar e apagar o registro)

3. **Integração com API:** A UI deve ser capaz de fazer solicitações às APIs para realizar operações básicas, incluindo GET, POST, PUT e DELETE. As solicitações devem OBRIGATORIAMENTE incluir o token JWT para autenticação.

4. **Exibição de Mensagens:** Garanta que a UI exiba as mensagens geradas pela API de forma clara e legível para o usuário.

5. **Exemplo de Solicitações:** Utilize a extensão REST Client no Visual Studio Code para criar exemplos de solicitações (usando token JWT) que interagem com as APIs. Documente essas chamadas de maneira clara no seu projeto.

6. **Documentação da API com Swagger:** Adicione documentação Swagger (https://swagger-autogen.github.io/docs/) à sua API para facilitar a compreensão e o uso por parte dos desenvolvedores. Isso inclui a descrição detalhada de cada rota, os parâmetros necessários, os tipos de resposta esperados e exemplos de solicitações.
   Dica:
   https://davibaltar.medium.com/documenta%C3%A7%C3%A3o-autom%C3%A1tica-de-apis-em-node-js-eb03041c643b

7. **Implementação dos testes da API**: Utilizando bibliotecas de testes com livre escolha, implemente no mínimo 5 diferentes testes unitários para a API desenvolvida. Exemplos de testes que podem ser implementados:

   Registro de Usuário:

   - Verificar se um novo usuário pode ser registrado com sucesso, fornecendo os dados obrigatórios (nome, email, senha).
   - Validar se o email do usuário já está registrado antes de permitir o registro.
   - Verificar se a senha do usuário atende aos requisitos mínimos de segurança (tamanho, caracteres especiais, etc.).
   - Garantir que o usuário registrado seja armazenado de forma segura no banco de dados.

   Autenticação de Usuário:

   - Verificar se um usuário registrado pode se autenticar com sucesso, fornecendo email e senha válidos.
   - Validar se os dados de autenticação (email e senha) estão corretos.
   - Garantir que um token JWT válido seja gerado para o usuário autenticado.
   - Verificar se o token JWT contém as informações do usuário e a data de expiração.

   Validação de Token JWT:

   - Validar se um token JWT válido permite o acesso a rotas protegidas.
   - Verificar se um token JWT inválido ou expirado é recusado.

- Garantir que o token JWT seja decodificado corretamente para extrair as informações do usuário.
- Validar se a data de expiração do token JWT ainda é válida no momento da solicitação.

Rotas Protegidas:

- Verificar se o acesso a rotas protegidas é restrito a usuários autenticados.
- Garantir que um usuário não autenticado seja redirecionado para a página de login.
- Validar se um usuário autenticado com token JWT válido tem acesso às rotas protegidas.

Tratamento de Erros:

- Verificar se erros de autenticação (email ou senha incorretos, token inválido ou expirado) são tratados de forma adequada.
- Garantir que mensagens de erro claras e informativas sejam exibidas ao usuário em caso de falha na autenticação.
- Validar se o sistema registra os erros de autenticação para fins de diagnóstico e auditoria.

Entrega:

Envie um link para o repositório do GitHub via a Tarefa do Microsoft Teams. Certifique-se de incluir o nome de todos os integrantes do grupo no README do projeto e o link da UI para facilitar a identificação dos colaboradores, assim como o link da API pública. (**Apenas um componente do grupo deve efetuar a entrega no Teams**)

Lembre-se de que o projeto deve ser funcional, bem documentado e seguir as melhores práticas de desenvolvimento web com Node.js, MongoDB e autenticação JWT.

Boa sorte! Happy coding! 🚀

- Os trabalhos com nota igual ou superior a **8.00** serão dispensados de fazer a segunda avaliação que será no dia 30/06. Neste caso, a nota da prova será a mesma do trabalho.