**Brendon Harrisson Avelino**

**Professor Marcos – SENAI**

**Instruções da minha API**

**index.html:**

<!DOCTYPE *html*>

<html *lang*="pt-BR">

<head>

    <meta *charset*="UTF-8">

    <meta *name*="viewport" *content*="width=device-width, initial-scale=1.0">

    <title>Criptografia de Mensagens</title>

    <link *rel*="stylesheet" *href*="styles.css">

</head>

<body>

    <div *class*="container">

        <h1>Criptografar e Descriptografar Mensagens</h1>

        <div *id*="box">

            <textarea *id*="message" *placeholder*="Digite sua mensagem aqui..."></textarea>

            <button *onclick*="*encryptMessage*()">Criptografar</button>

            <button *onclick*="*decryptMessage*()">Descriptografar</button>

        </div>

        <div *id*="result"></div>

    </div>

    <script>

        const *apiUrl* ='https://82874cb7-7cf0-4525-99ad-b92eccae1aaa-00-3bqgt2l7n7jni.picard.replit.dev';

*async* function *encryptMessage*() {

*try* {

                const *message* = *document.getElementById*('message')*.value*;

                const *response* = *await fetch*(`${apiUrl}*/encrypt*`,{

*method*:'POST',

*headers*:{'Content-Type':'application/json'},

*body*: *JSON.stringify*({message})

});

                if (!*response.*ok) *throw* new *Error*('Erro na requisição');

                const *data* = *await response.json*();

*document.getElementById*('result')*.*innerText = `Criptografado: ${*data.*encrypted}\nIV: ${*data.*iv}`;

            } *catch* (error) {

*console.error*('Erro:', error);

*alert*('Ocorreu um erro ao criptografar a mensagem.');

            }

        }

*async* function *decryptMessage*() {

*try* {

                const[*encrypted*, *iv*]= *document.getElementById*('result')*.innerText.split*('\n')*.map*(*line* => *line.split*(': ')[1]);

                const *response* = *await fetch*(`${apiUrl}*/decrypt*`,{

*method*:'POST',

*headers*:{'Content-Type':'application/json'},

*body*: *JSON.stringify*({encrypted,iv})

});

                if (!*response.*ok) *throw* new *Error*('Erro na requisição');

                const *data* = *await response.json*();

*document.getElementById*('result')*.*innerText = `Descriptografado: ${*data.*decrypted}`;

            } *catch* (error) {

*console.error*('Erro:', error);

*alert*('Ocorreu um erro ao descriptografar a mensagem.');

            }

        }

    </script>

</body>

</html>

**index.js ( o mesmo código também é usado no index.js no replit ) :**

const *express* = *require*('express');

const *cors* = *require*('cors');

const *crypto* = *require*('crypto');

const *app* = *express*();

*app.use*(*cors*());

*app.use*(*express.json*());

const *algorithm* ='aes-256-cbc';

const *key* = *crypto.randomBytes*(32); // *Chave aleatória para criptografia*

const *iv* = *crypto.randomBytes*(16);  // *Vetor de inicialização aleatório*

// *Rota para criptografar*

*app.post*('/encrypt', (req, res) => {

    const{ *message* }= *req.body*;

    if (!message) *return* *res.status*(400)*.json*({ error: 'Mensagem não fornecida' });

    const *cipher* = *crypto.createCipheriv*(algorithm,key,iv);

    letencrypted= *cipher.update*(message,'utf-8','hex');

    encrypted += *cipher.final*('hex');

*res.json*({ encrypted, iv: *iv.toString*('hex') });

});

// *Rota para descriptografar*

*app.post*('/decrypt', (req, res) => {

    const{ *encrypted*, *iv* }= *req.body*;

    if (!encrypted || !iv) *return* *res.status*(400)*.json*({ error: 'Dados incompletos' });

    const *decipher* = *crypto.createDecipheriv*(algorithm,key, *Buffer.from*(iv,'hex'));

    letdecrypted= *decipher.update*(encrypted,'hex','utf-8');

    decrypted += *decipher.final*('utf-8');

*res.json*({ decrypted });

});

// *Verificar funcionamento*

*app.get*('/', (req, res) => {

*res.send*('API funcionando!');

});

const *PORT* = *process.env.PORT* ||8080;

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

👇

👇

Estrutura do código:

**Estrutura Geral**

* **Frontend (HTML e JavaScript):**
  + O código HTML contém um campo de texto (textarea) onde o usuário pode inserir uma mensagem.
  + Existem dois botões: um para criptografar a mensagem e outro para descriptografá-la.
  + Quando um botão é clicado, uma função JavaScript é chamada para interagir com a API.
* **Backend (Node.js com Express):**
  + A API é construída usando o Express, que facilita a criação de rotas e manipulação de requisições HTTP.
  + A criptografia é feita utilizando o módulo crypto do Node.js.

**Funcionamento das Funções**

1. **Criptografar Mensagem:**
   * Quando o usuário clica no botão "Criptografar":
     + A função encryptMessage() é chamada.
     + A mensagem é enviada para a rota /encrypt da API via uma requisição POST.
     + A API gera um texto criptografado usando o algoritmo AES-256-CBC, uma chave aleatória e um vetor de inicialização (IV).
     + O resultado (texto criptografado e IV) é retornado e exibido na página.
2. **Descriptografar Mensagem:**
   * Quando o usuário clica no botão "Descriptografar":
     + A função decryptMessage() é chamada.
     + A função extrai o texto criptografado e o IV da exibição anterior.
     + Esses dados são enviados para a rota /decrypt da API via uma requisição POST.
     + A API usa o mesmo algoritmo e IV para descriptografar o texto, retornando a mensagem original.
     + O resultado (mensagem descriptografada) é exibido na página.

**Erros e Tratamento:**

* O código inclui tratamento de erros para lidar com falhas de requisições, exibindo alertas ao usuário em caso de problemas.

Passos para o funcionamento correto do código:

**Iniciar o Servidor:**

* Inicie o servidor no Replit pressionando o botão “Run”

**Abrir o HTML:**

* Abra o arquivo index.html no navegador.

**Inserir Mensagem:**

* Digite a mensagem no campo de texto (textarea).

**Criptografar:**

* Clique no botão **"Criptografar"** para enviar a mensagem à API. A mensagem criptografada e o IV aparecerão abaixo.

**Descriptografar:**

* Clique no botão **"Descriptografar"** para reverter a mensagem criptografada para seu formato original.

**Verificar Resultados:**

* Confira a área de resultados para confirmar se a criptografia e a descriptografia funcionaram corretamente.

Fim 👍👊