



# **AGENDA 15: CRIPTOGRAFIA EM DADOS PHP**

**Desenvolvimento de Sistemas III**

Mônica Zungalo Quintal

# O QUE É CRIPTOGRAFIA DE DADOS?

- Trata-se de uma forma de traduzir dados no formato texto simples (não criptografados) para o formato texto cifrado (criptografados).
- Os usuários podem acessar os dados criptografados com uma chave de criptografia e os dados descriptografados com uma chave de descriptografia.
- Tem quatro objetivos principais:
  - ✓ **Confidencialidade:** disponibiliza informações somente para usuários autorizados.
  - ✓ **Integridade:** garante que as informações não tenham sido manipuladas.
  - ✓ **Autenticação:** confirma autenticidade das informações ou identidade do usuário.
  - ✓ **Não repúdio:** impede que um usuário negue compromissos ou ações anteriores.

# CRIPTOGRAFIA EM PHP

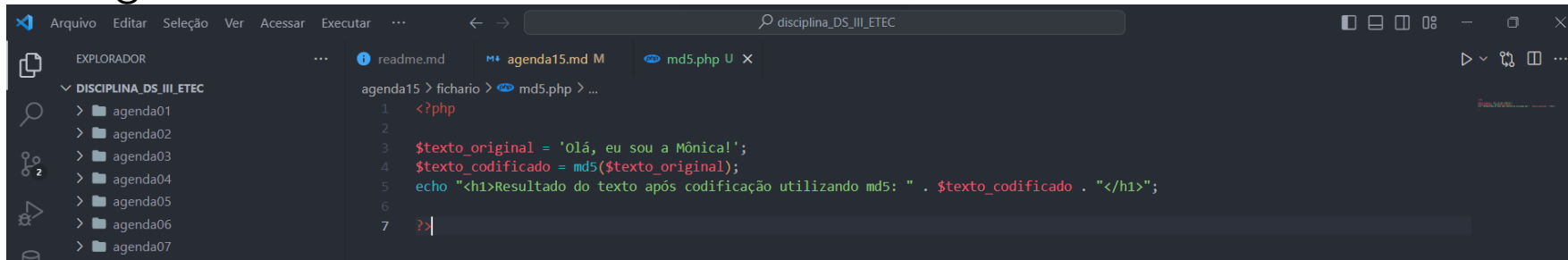
- Todas as linguagens de programação oferecem funções que trabalham com algum tipo de criptografia.
- Cada tipo de criptografia é mais indicado para um tipo de aplicação: as **menos complexas** normalmente são **mais rápidas**, no entanto **menos seguras**.
- Algumas das funções mais conhecidas no PHP para essa finalidade são:
  - md5.
  - sha1.
  - hash.

# MD5

- Este algoritmo foi muito utilizado no começo dos anos 2000 para fazer a criptografia das senhas armazenadas no banco de dados.
- Trata-se de uma hash de mão única (um conteúdo só pode ser codificado, e não o contrário.).
- Usa a função de hash criptográfica MD5 para **converter uma string de comprimento variável** em uma **string alfa-numérica de 32 caracteres**, que é uma representação de texto do valor hexadecimal de uma soma de verificação de 128 bits.
- Sintaxe: `md5(string);`

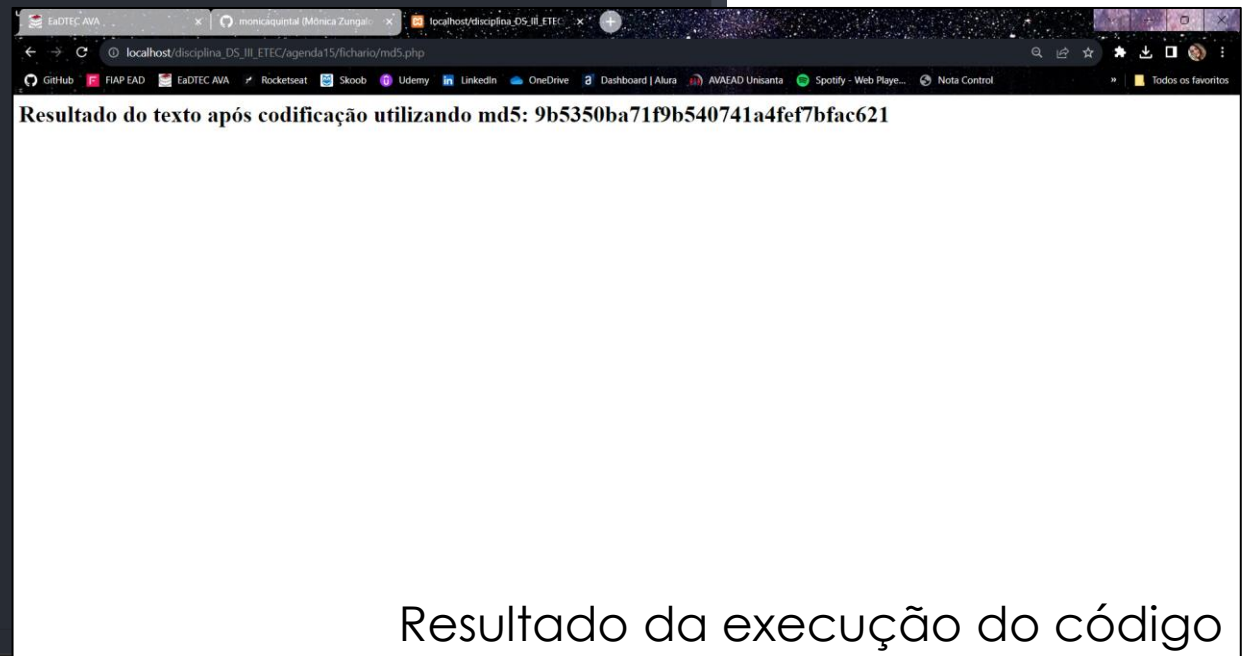
# EXEMPLO: MD5

Código desenvolvido:



The screenshot shows a code editor with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure with folders 'agenda01' through 'agenda14', a 'project' folder, and a 'agenda15' folder containing a 'fichario' folder and a 'md5.php' file. The code editor shows the following PHP code:

```
1 <?php
2
3 $texto_original = 'Olá, eu sou a Mônica!';
4 $texto_codificado = md5($texto_original);
5 echo "<h1>Resultado do texto após codificação utilizando md5: " . $texto_codificado . "</h1>";
6
7 >>
```



Resultado da execução do código

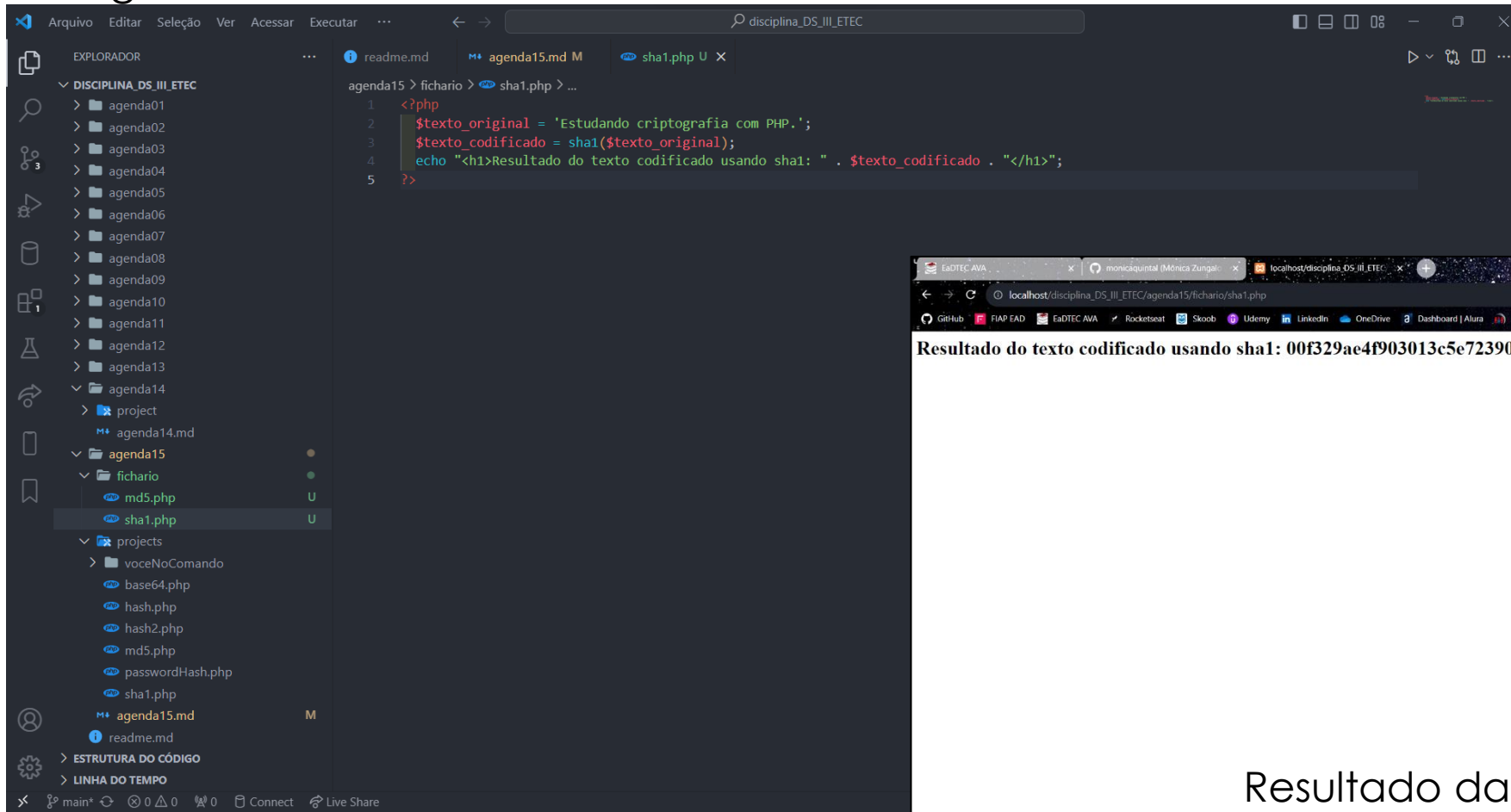
# SHA1

- Usa a função de hash criptográfica SHA1 para converter uma **string de comprimento variável** em uma **string de 40 caracteres** que é uma representação de texto do valor hexadecimal de uma soma de verificação de 160 bits.
  - Ou seja, segue o proposto pelo MD5, porém possui 160 bits, resultando em uma string maior, formada por 40 caracteres alfa-numéricos.
- Trata-se de uma hash de mão única (um conteúdo só pode ser codificado, e não o contrário).
- Sintaxe: `sha1(string);`



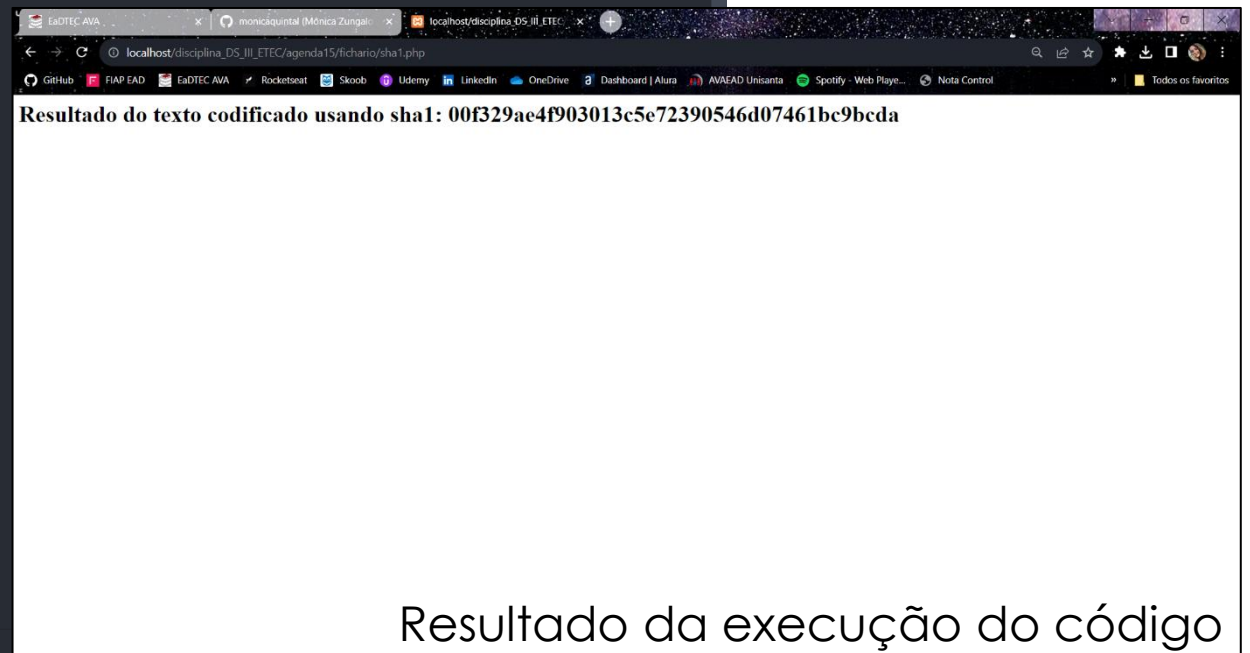
# EXEMPLO: SHA1

Código desenvolvido:



The screenshot shows a code editor with a file explorer on the left. The file explorer shows a directory structure with folders 'agenda01' through 'agenda14', a 'project' folder, and a 'agenda15' folder. Inside 'agenda15', there is a 'fichario' folder containing 'md5.php' and 'sha1.php'. The 'sha1.php' file is selected. The code editor shows the following PHP code:

```
1 <?php
2 $texto_original = 'Estudando criptografia com PHP.';
3 $texto_codificado = sha1($texto_original);
4 echo "<h1>Resultado do texto codificado usando sha1: " . $texto_codificado . "</h1>";
5 ?>
```



Resultado da execução do código



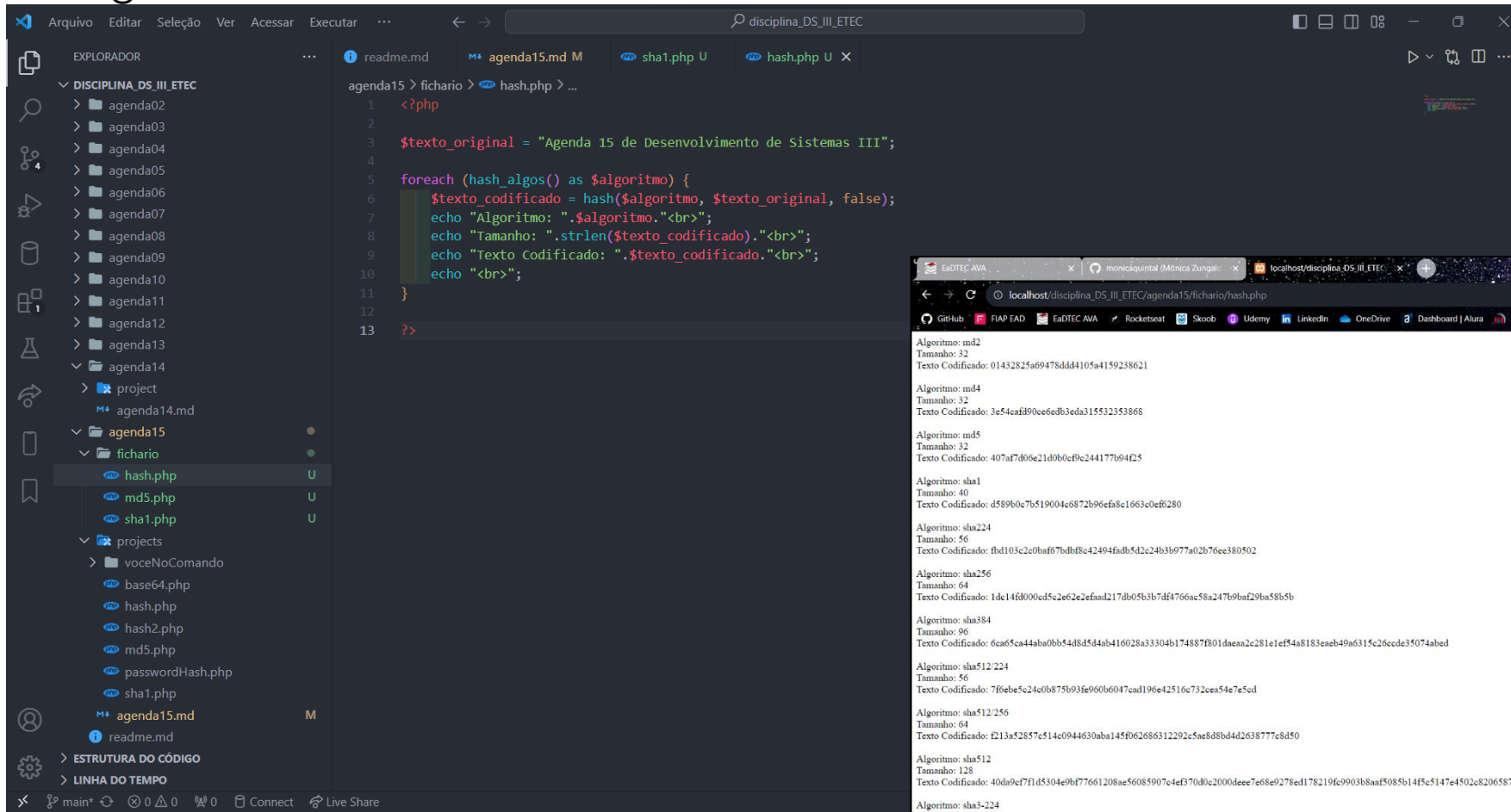
# HASH

- A função hash também é de mão única e **retorna o hash calculado de um valor**, no entanto nesta **função é possível definir qual algoritmo será usado** para transformar o valor em hash.
- Trata-se de uma função que possibilita a escolha do algoritmo hash de mão única.

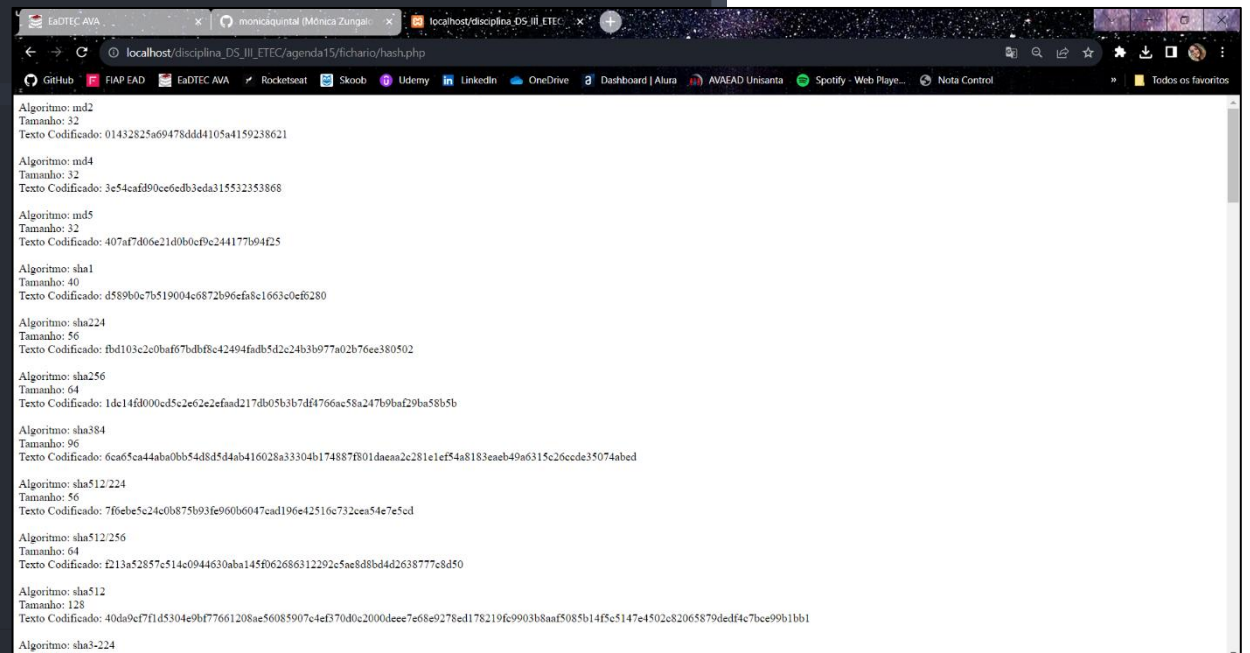


# EXEMPLO: HASH

Código desenvolvido:



```
1 <?php
2
3 $texto_original = "Agenda 15 de Desenvolvimento de Sistemas III";
4
5 foreach (hash_algos() as $algoritmo) {
6     $texto_codificado = hash($algoritmo, $texto_original, false);
7     echo "Algoritmo: ".$algoritmo."<br>";
8     echo "Tamanho: ".strlen($texto_codificado)."<br>";
9     echo "Texto Codificado: ".$texto_codificado."<br>";
10    echo "<br>";
11 }
12
13 ?>
```



Resultado da execução do código

# PASSWORD\_HASH

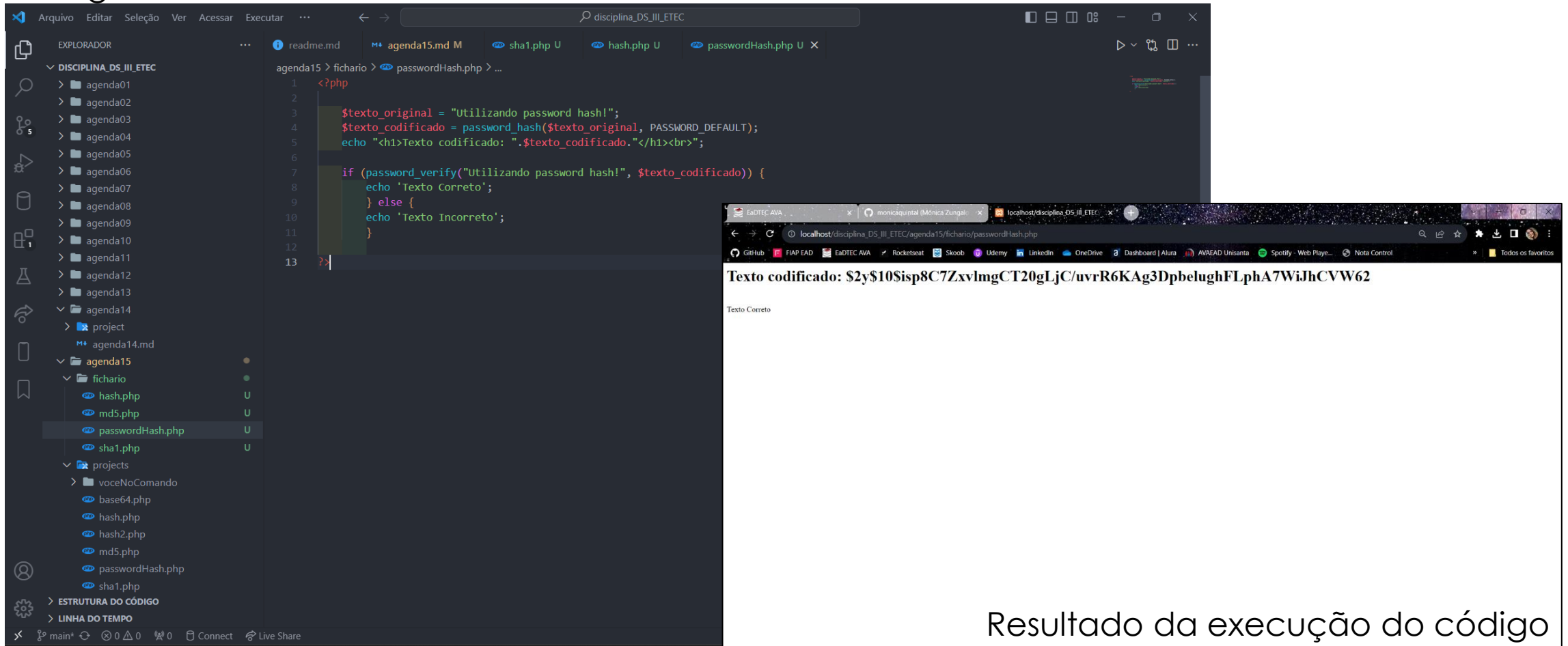
- Trata-se de uma **API** que fornece um conjunto de funções que facilitam a utilização de hash focado em senhas, ajudando o desenvolvedor a criar hashes seguros.
- Há duas funções: **password\_hash** (criará a hash) e **password\_verify** (fará a verificação da hash).
- O **password\_hash()** é usado para criar o hash uma determinada string usando o algoritmo mais forte atualmente disponível e **password\_verify()** verifica se a senha fornecida corresponde ao hash armazenado no banco de dados.

# PASSWORD\_HASH

- Sintaxe: `password_hash ("texto", PASSWORD_DEFAULT);`
  - primeiro parâmetro: texto a ser codificado e
  - segundo parâmetro: as configurações de custo e salt.
- O *salt* é utilizado para evitar que duas senhas idênticas produzam hashes idênticos, o que facilita a descoberta de senhas.
- O *custo* é o valor que determina o quão complexo o algoritmo deve ser e, portanto, o quanto demorará a geração do hash.

# EXEMPLO: PASSWORD\_HASH

Código desenvolvido:



The image shows a development environment with a code editor and a web browser. The code editor displays the following PHP code in `passwordHash.php`:

```
1 <?php
2
3 $texto_original = "Utilizando password hash!";
4 $texto_codificado = password_hash($texto_original, PASSWORD_DEFAULT);
5 echo "<h1>Texto codificado: ".$texto_codificado."</h1><br>";
6
7 if (password_verify("Utilizando password hash!", $texto_codificado)) {
8     echo 'Texto Correto';
9 } else {
10     echo 'Texto Incorreto';
11 }
12
13 ?>
```

The web browser shows the result of the script execution at `localhost/disciplina_DS_III_ETEC/agenda15/fichario/passwordHash.php`. The output is:

Texto codificado: \$2y\$10\$isp8C7ZxvImgCT20gLjC/uvrR6KAg3DpbelughFLphA7WiJhCVW62

Texto Correto

Resultado da execução do código

# CONCLUSÃO

- Uma maneira de **proteger a integridade dos dados** é através da criptografia, que assegura o conteúdo de uma mensagem ou arquivo através de algoritmos avançados.
- Logo, é possível **manter o sigilo** das informações, bem como **garantir que nenhum dado seja acessado** por um terceiro não autorizado!
- Além disso, a criptografia atinge vários objetivos relacionados à **segurança da informação**, incluindo **confidencialidade, integridade e autenticação**.

# BIBLIOGRAFIA

- IBM. **O que é criptografia? Definição de criptografia de dados.** Disponível em: <<https://www.ibm.com/br-pt/topics/encryption>>. Acesso em: 16 de novembro de 2023.
- AWS. **O que é criptografia?** Disponível em: <<https://aws.amazon.com/pt/what-is/cryptography/>>. Acesso em: 16 de novembro de 2023.
- AWS. **Função MD5.** Disponível em: <[https://docs.aws.amazon.com/pt\\_br/redshift/latest/dg/r\\_MD5.html](https://docs.aws.amazon.com/pt_br/redshift/latest/dg/r_MD5.html)>. Acesso em: 16 de novembro de 2023.
- AWS. **Função SHA1.** Disponível em: <[https://docs.aws.amazon.com/pt\\_br/clean-rooms/latest/sql-reference/SHA1.html](https://docs.aws.amazon.com/pt_br/clean-rooms/latest/sql-reference/SHA1.html)>. Acesso em: 16 de novembro de 2023.
- Diogo Bemfica. **Criptografia com PHP.** 2019. Disponível em: <<https://diogobemfica.com.br/criptografia-com-php>>. Acesso em: 16 de novembro de 2023.
- PHP. Documentação PHP: **Modelo de Armazenamento Criptografado.** Disponível em: <[https://www.php.net/manual/pt\\_BR/security.database.storage.php](https://www.php.net/manual/pt_BR/security.database.storage.php)>. Acesso em: 16 de novembro de 2023.