UNIVERSIDADE DO OESTE DE SANTA CATARINA

UNOESC – CAMPUS SÃO MIGUEL DO OESTE

CURSO DE CIÊNCIA DA COMPUTAÇÃO

DOCUMENTAÇÃO DOS RESULTADOS DOS TESTES UNITÁRIOS

UTILIZANDO PHPUnit

ENGENHARIA DE SOFTWARE II – SMO11-8

Cauana Rosin Ghizzi

Natani Gayardo

São Miguel do Oeste – SC

2025

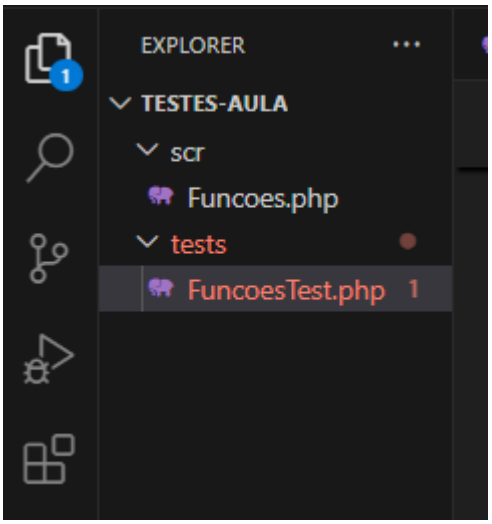
1. Ambiente de Desenvolvimento:

Sistema operacional: Windows 11 Home Single Language

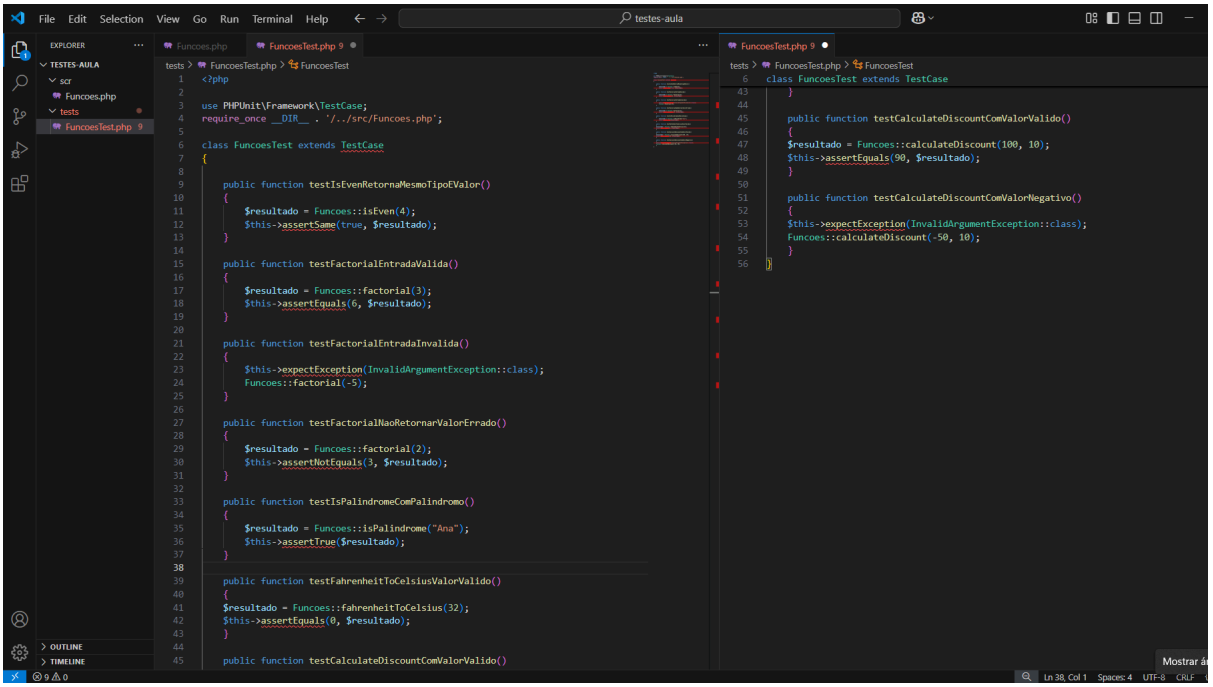
Versão do PHP: PHP 8.2.28

Versão do PHPUnit: 9

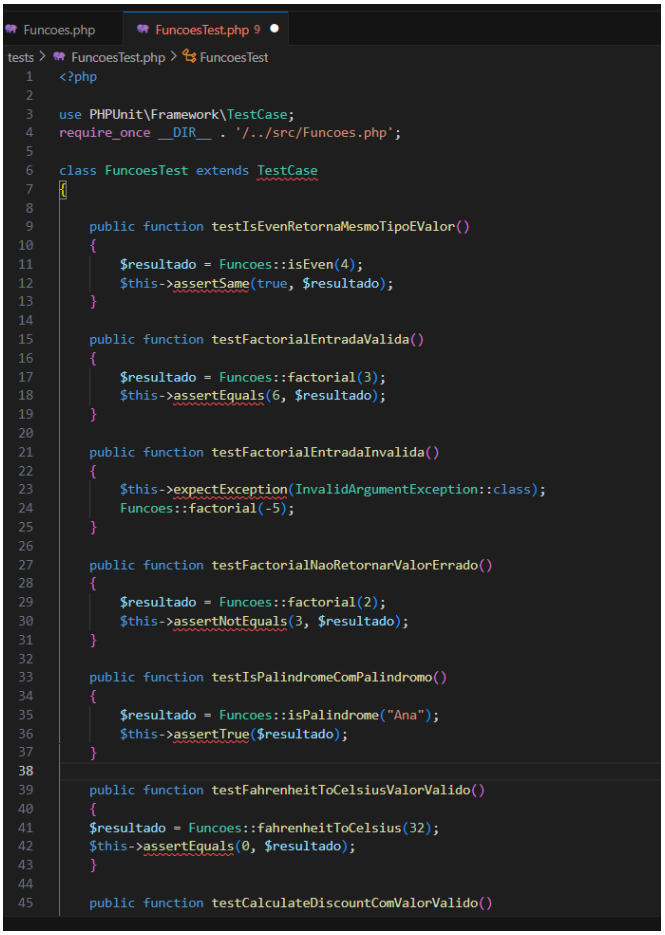
2.Estrutura do projeto:

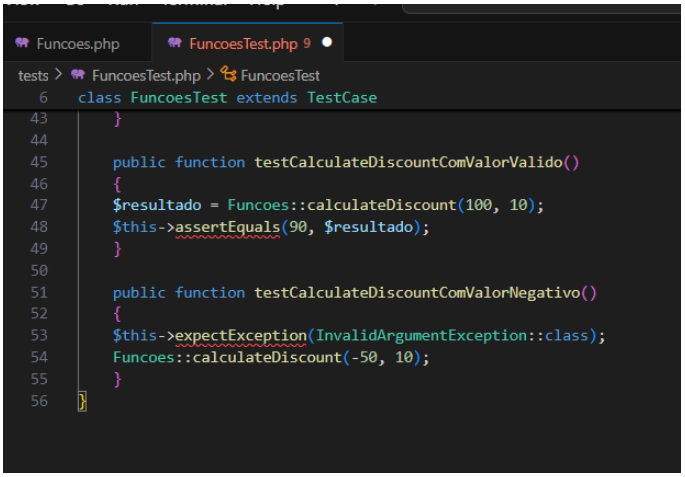


3.Implementação dos Testes Unitários (tests/FuncoesTest.php):



(PrtSc do código com os testes unitários das funções/Tela toda)





(PrtSc do código com os testes unitários das funções/Com zoom)

4. Descrição dos testes

Foram adicionados docblocks no próprio código recentemente para melhor praticidade:

L8 - L11 =

```
/**
    * Testa se a função isEven retorna true para um número par
    * Garantindo que o tipo e o valor retornado são iguais a true
    */
```

L18 - L21=

```
/**
    * Testa o cálculo do fatorial se a entrada válida ou seja número positivo
    * Verificando se o resultado para o fatorial de 3 é igual a 6 nesse caso
    */
```

L28 -L30=

```
/**
    * Testando lançamento de InvalidArgumentException para entrada inválida com número negativo na função factorial
    */
```

L37 - L40=

```
/**
    * Garantindo que a função factorial não vai retornar um valor incorreto sendo na entrada válida
    * Aqui nessa função verificando se o fatorial de 2 não é igual a 3
    */
```

L47 - L50=

```
 /**
    * Testando se a função isPalindrome identifica uma string palíndroma ("Ana")
    * A comparação precisa ser case-insensitive e ignorar caracteres não alfanuméricos se tiver
    */
```

L57 - L59=

```
/**
    * Testando a conversão de Fahrenheit para Celsius com um valor válido aqui
    */
```

L66 - L68=

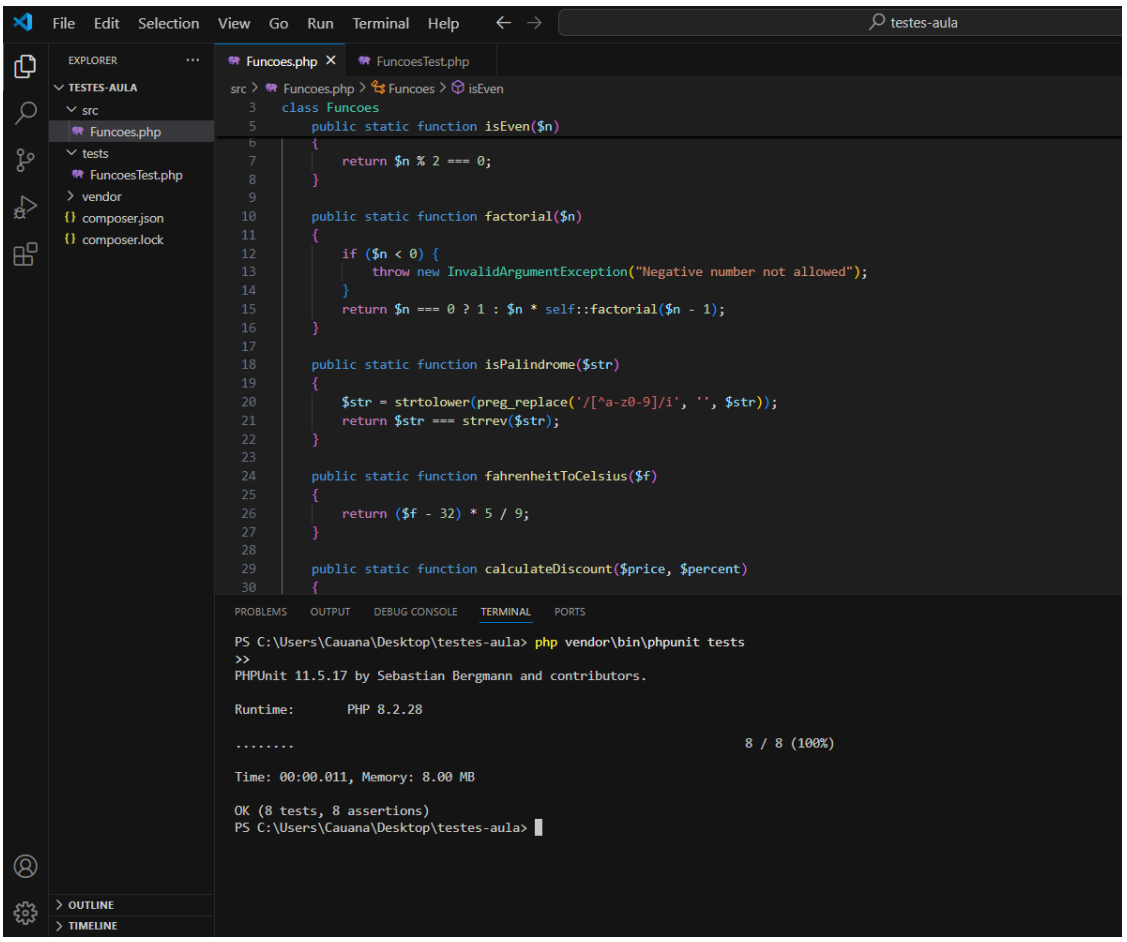
```
/**
    * Testando o cálculo do desconto com valores de preço e porcentagem válidos
    */
```

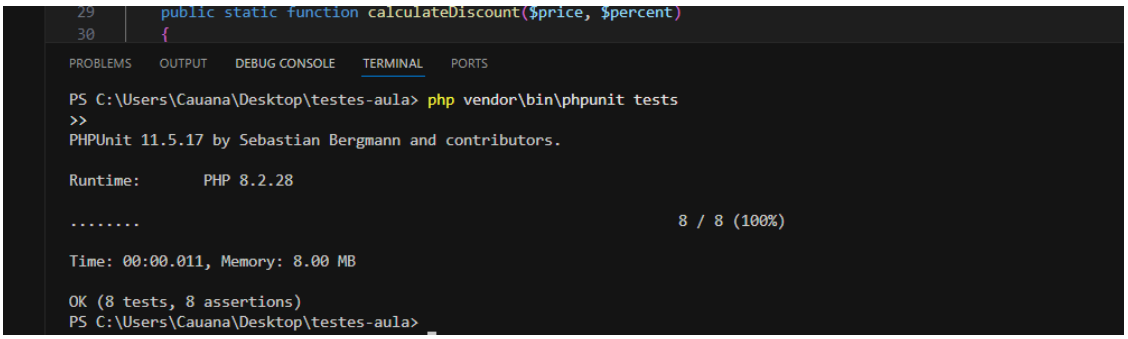
L75- L77 =

```
/**
    * Testando o lançamento de InvalidArgumentException quando o preço é negativo na função no caso entrada invalida
    */
```

Dicionário : (L) é a linha do código em que o comentário está.
(-) é até qual linha este comentário vai

5. Resultado da execução dos testes utilizando o comando: `./vendor/bin/phpunit test





6. Reflexão final nos testes

- Os testes ajudaram a identificar comportamentos inesperados?

Sim, foi fundamental. Podemos concluir que tiveram alguns problemas que talvez passariam despercebidos se fossem validados manualmente. Durante a configuração, ocorreram alguns erros evidenciados antes da execução dos testes em si, como uma falha no carregamento da classe Funcoes por caso de um erro (de escrita humana) no caminho do arquivo que invés de src estava scr, acredito que esse problema estrutural poderia

comprometer todo o desenvolvimento se não identificado. Após essa correção e executar adequadamente os testes, foi validado corretamente o comportamento das funções.

- Algum teste falhou? Por quê?

No período inicial os testes demoraram para ter sucesso na execução por conta de erros de estrutura e de configuração como classe de teste não reconhecida, arquivo não encontrado por ter nome incorreto, dependência não compatível com a versão do PHP onde foi preciso fazer um upgrade. As falhas no mais, foram mais problemas no ambiente de dev e organização do projeto do que de falhas lógicas nas funções. Quando foram corrigidos estes pontos citados, os testes foram foram executados corretamente e todos passaram, tendo como resultado as funções implementadas corretamente de acordo com os critérios definidos nos testes.

- Como os testes podem ajudar na evolução segura do código?

Eles são como uma rede de segurança durante o desenvolvimento do software. Quando o projeto começa a crescer e surgem mudanças nas funções, por exemplo, os testes têm o papel de garantir que o funcionamento do que já existe continue estável mesmo com mais implementações no código posteriormente. Facilita para o dev entender se alguma alteração causou algum bug como no caso do scr (acima) mesmo de exemplo, assim é muito mais prático fazer alterações e adições sem estar quebrando o código. Também assim como solicitado aqui na atividade para que fosse documentado, os testes servem como documentação prática, mostrando como as funções devem se comportar e quais são as situações esperadas para cada uma delas, por exemplo, onde qualquer dev que pega a documentação está a nivel de entender e fazer a manutenção nele, entre outros.