Setor9 - Negócios Online
www.setor9.com.br
Tel. 12-981924785
E-mail. contato@setor9.com.br

Guia: Proteger credenciais em projetos PHP com vlucas/phpdotenv

Problema

Credenciais (banco de dados, API keys) hardcoded no código-fonte são commitadas no repositório e ficam expostas no histórico do Git.

Solução

Mover todas as credenciais para um arquivo `.env` que fica **fora** do repositório, usando a biblioteca `vlucas/phpdotenv` para carregá-las como variáveis de ambiente.

Passo a passo

1. Instalar a dependência

Na raiz do projeto (onde está o `composer.json`):

```shell
composer require vlucas/phpdotenv
```

2. Criar o arquivo `.env` na raiz do projeto

Preencher com os valores reais do ambiente:

```
DB_HOST=127.0.0.1
DB_PORT=3306
DB_NAME=meu_banco
DB_USER=meu_usuario
DB_PASSWORD=minha_senha
OPENAI_API_KEY=sk-proj-minha-chave
```

Este arquivo nunca vai para o repositório. Cada ambiente (local, produção, staging) tem o seu próprio `.env`.

3. Criar o `.env.example` na raiz

Modelo sem valores sensíveis — este sim vai para o repositório:

```
None
DB_HOST=127.0.0.1
DB_PORT=3306
DB_NAME=
DB_USER=
DB_PASSWORD=
OPENAI_API_KEY=
```

4. Criar ou atualizar o `.gitignore`

Adicionar:

```
None
.env
vendor/
```

5. Criar o `bootstrap.php` na raiz

```php
<?php

require_once __DIR__ . '/vendor/autoload.php';

$dotenv = Dotenv\Dotenv::createImmutable(__DIR__);
$dotenv->load();

$dotenv->required([
    'DB_HOST',
    'DB_PORT',
    'DB_NAME',
    'DB_USER',
    'DB_PASSWORD',
    'OPENAI_API_KEY',
])->notEmpty();
```

O `required()->notEmpty()` faz a aplicação parar com erro claro se alguma variável estiver ausente ou vazia.

6. Incluir o bootstrap no ponto de entrada da aplicação

No `public/index.php` (ou equivalente), substituir:

```php
// ANTES
require '../Application/autoload.php';
require '../vendor/autoload.php';
```

Por:

```php
// DEPOIS
require '../bootstrap.php';
require '../Application/autoload.php';
```

O `bootstrap.php` já carrega o `vendor/autoload.php`, então não precisa duplicar.

7. Substituir credenciais hardcoded pelo `$_ENV`

Banco de dados — onde antes tinha:

```php
private $DB_NAME = 'meu_banco';
private $DB_USER = 'root';
private $DB_PASSWORD = 'minha_senha';
```

Trocar por:

```php
private $DB_NAME;
private $DB_USER;
private $DB_PASSWORD;
private $DB_HOST;
private $DB_PORT;
```

```
public function __construct()
{
    $this->DB_NAME     = $_ENV['DB_NAME'];
    $this->DB_USER     = $_ENV['DB_USER'];
    $this->DB_PASSWORD = $_ENV['DB_PASSWORD'];
    $this->DB_HOST     = $_ENV['DB_HOST'];
    $this->DB_PORT     = $_ENV['DB_PORT'];

    $this->conn = new PDO(

"mysql:dbname={$this->DB_NAME};host={$this->DB_HOST};port={$this->DB_PORT}",
        $this->DB_USER,
        $this->DB_PASSWORD
    );
}
```

API keys — onde antes tinha:

```
PHP
$apiKey = 'sk-proj-chave-hardcoded';
```

Trocar por:

```
PHP
$apiKey = $_ENV['OPENAI_API_KEY'];
```

8. Testar localmente

Acessar a aplicação normalmente. Se alguma variável estiver faltando no `.env`, o dotenv mostra um erro claro dizendo qual é.

9. Commitar

```
Shell
git add .gitignore .env.example bootstrap.php public/index.php
Application/core/Database.php Application/models/Redacoes.php
git commit -m "refactor: mover credenciais para .env via phpdotenv"
```

Antes do `git add`, rodar `git status` e confirmar que o `.env` **não aparece** na lista.

> Se o `.env` aparecer, significa que ele já foi rastreado antes. Resolver com:

```shell
git rm --cached .env
```

10. Configurar produção

Criar o `.env` manualmente no servidor com os valores de produção. Este arquivo é criado **uma única vez** e não é gerenciado pelo Git.

Deploy: preservar o `.env` no servidor

Se o script de deploy apaga o diretório antes de clonar (como `rm -rf`), o `.env` será apagado junto. A solução é salvar e restaurar:

```shell
# Salvar antes de apagar
cp /caminho/do/projeto/.env /tmp/.env.backup

# Apagar e clonar
rm -rf /caminho/do/projeto
git clone git@github.com:usuario/repo.git /caminho/do/projeto

# Restaurar
mv /tmp/.env.backup /caminho/do/projeto/.env
```

Checklist rápido para novos projetos

- [ ] `composer require vlucas/phpdotenv`
- [ ] Criar `.env` com valores reais (local)
- [ ] Criar `.env.example` sem valores (vai pro repo)
- [ ] Adicionar `.env` ao `.gitignore`
- [ ] Criar `bootstrap.php` com `Dotenv::createImmutable`

- [ ] Incluir `bootstrap.php` no ponto de entrada
- [ ] Trocar credenciais hardcoded por `$_ENV['VARIAVEL']`
- [ ] Verificar com `git status` que `.env` não será commitado
- [ ] Criar `.env` no servidor de produção
- [ ] Ajustar script de deploy para preservar o `.env`