🔐 Adicionando Segurança em Aplicações

Prof. Gabriel Caixeta Silva

🔐 Adicionando Segurança em Aplicações SlimPHP com `.env`

Em aplicações modernas, é fundamental **proteger credenciais sensíveis** como senhas de banco de dados, chaves de API e dados de SMTP. No SlimPHP, isso pode ser feito facilmente com o uso de variáveis de ambiente via `.env`.

✅ Por que usar `.env`?

Arquivos `.env` permitem manter **as configurações separadas do código-fonte**, evitando que dados sensíveis sejam versionados no Git ou expostos em ambientes públicos.

Exemplos de informações protegidas:

- Dados do banco de dados
- Chaves de API (ex: Mailtrap, Stripe)
- Configurações de e-mail SMTP
- Modo de debug

📁 Estrutura do Projeto

Exemplo de organização recomendada:

```
meu-projeto/
├── public/
│   └── index.php             # Ponto de entrada da aplicação
├── src/
│   └── App/Database/         # Código fonte (seguindo PSR-4)
├── .env                      # Variáveis reais (NÃO deve ir para o Git)
├── .env.example              # Modelo para outros desenvolvedores
├── composer.json
```

📦 1. Instalando o `phpdotenv`

Vamos usar a biblioteca vlucas/phpdotenv, que é amplamente usada em projetos PHP modernos.

Execute:

```
composer require vlucas/phpdotenv
```

🛠️ 2. Criando o arquivo `.env`

Importante: nunca versionar esse arquivo.

Crie um arquivo `.env` na raiz do projeto com os dados reais (exemplo):

```
APP_ENV=development
APP_DEBUG=true

DB_HOST=localhost
DB_NAME=my_tarefas
DB_USER=root
DB_PASS=123456

MAIL_HOST=sandbox.smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=seu_usuario_mailtrap
MAIL_PASSWORD=sua_senha_mailtrap
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=no-reply@seudominio.com
MAIL_FROM_NAME="Minha API"
```

📄 3. Criando `.env.example`

Esse arquivo serve como modelo para outros desenvolvedores:

```
APP_ENV=production
APP_DEBUG=false

DB_HOST=
DB_NAME=
DB_USER=
DB_PASS=

MAIL_HOST=sandbox.smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=
MAIL_PASSWORD=
MAIL_ENCRYPTION=null
MAIL_FROM_ADDRESS=no-reply@example.com
MAIL_FROM_NAME="Minha API"
```

🚫 4. Ignorando o `.env` no Git

No arquivo `.gitignore`, adicione:

```
.env
```

🔃 5. Carregando as variáveis no `index.php`

No início do arquivo `public/index.php`, adicione:

```php
use Dotenv\Dotenv;

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

$dotenv = Dotenv::createImmutable(__DIR__ . '/../');
$dotenv->load();
```

🧩 6. Usando variáveis no código (exemplo com banco de dados)

```php
namespace App\Database;

class Mariadb {
    private ?\PDO $connection = null;

    public function __construct() {
        $host     = $_ENV['DB_HOST']     ?? throw new \RuntimeException('DB_HOST não definido');
        $dbname   = $_ENV['DB_NAME']     ?? throw new \RuntimeException('DB_NAME não definido');
        $username = $_ENV['DB_USER']     ?? throw new \RuntimeException('DB_USER não definido');
        $password = $_ENV['DB_PASS']     ?? throw new \RuntimeException('DB_PASS não definido');

        try {
            $this->connection = new \PDO(
                "mysql:host=$host;dbname=$dbname;charset=utf8",
                $username,
                $password,
                [
                    \PDO::ATTR_ERRMODE => \PDO::ERRMODE_EXCEPTION,
                    \PDO::ATTR_DEFAULT_FETCH_MODE => \PDO::FETCH_ASSOC,
                    \PDO::ATTR_EMULATE_PREPARES => false,
                ]
            );
        } catch (\PDOException $e) {
            die("Erro ao conectar: " . $e->getMessage());
        }
    }

    public function getConnection(): ?\PDO {
        return $this->connection;
    }
}
```

✉️ 7. Usando variáveis para envio de e-mails

Se for usar Mailtrap com PHPMailer, por exemplo:

```
$mail->Host = $_ENV['MAIL_HOST'];
$mail->Port = $_ENV['MAIL_PORT'];
$mail->Username = $_ENV['MAIL_USERNAME'];
$mail->Password = $_ENV['MAIL_PASSWORD'];
$mail->SMTPSecure = $_ENV['MAIL_ENCRYPTION'] ?? 'tls';
$mail->setFrom($_ENV['MAIL_FROM_ADDRESS'], $_ENV['MAIL_FROM_NAME']);
```

🔐 Conclusão

O uso de `.env`:

✅ Protege dados sensíveis
✅ Facilita a configuração em diferentes ambientes (dev, prod, homolog)
✅ Segue boas práticas de segurança e manutenção

📚 Referências

- vlucas/phpdotenv
- Slim Framework
- Mailtrap SMTP