https://stackoverflow.blog/2023/02/13/coding-102-writing-code-other-people-can-read/?cb=1&_ga=2.147656567.1980597688.1676268773-1320968652.1671520452

Variable

1) **Descriptive** - Names given to variables should make sense and relate to what it really stores.
2) Use a lower Camel case for names with more than one word.

Constant Variable

Two types

1) Global 2) Class

Global Constant related to particular one entity can be defined like this for Ex Database
Ex. DB_HOST, DB_USER, DB_PASS, DB_NAME

Class Constant for Circle Class can be
Const PI = 3.14

1) All Caps
2) Descriptive

Function Name

Two of them are majorly used: 1) lowercase with underscore 2) lower camelcase

1) The setter function name should start with the **verb** and **entity** being affected by the function and for the getter verb entity then Properties ex. getPostId().
2) The class function can be defined without an entity because the class itself represents an entity.
3) Methods returning boolean values should start with “has” or “is”. Other getters should start with “get“. Ex . isUserLogged(), hasKey()
4) Setter methods should start with “set”.

Class

Class names are always written in UpperCamelCase.
Class names must be nouns never adjectives.
Class name should
Private and protected properties in class must be prefixed with a single underscore for Ex

```
Class Name{
        private  $_radius;
}
```

Controller Naming Convention

Ex. ArticleController
1) Uppercase
2) Use a Singular Name

Route

Ex. articles/1
1) Plural

Model

Ex. User

1) Singular

Best Practices

- A class and method should have only one responsibility.
- For Ex below program check auth user, client, and isVerfucation. In a single method

```
public function getFullNameAttribute(): string
{
   if (auth()->user() && auth()->user()->hasRole('client') && auth()->user()->isVerified()) {
      return 'Mr. ' . $this->first_name . ' ' . $this->middle_name . ' ' . $this->last_name;
   } else {
      return $this->first_name[0] . '. ' . $this->last_name;
   }
}
```

- The good practice for this program should be an individual method for each work like isverfiedClient, and getFullNameLong().

```
public function getFullNameAttribute(): string
{
   return $this->isVerifiedClient() ? $this->getFullNameLong() : $this->getFullNameShort();
}

public function isVerifiedClient(): bool
{
   return auth()->user() && auth()->user()->hasRole('client') && auth()->user()->isVerified();
}

public function getFullNameLong(): string
{
   return 'Mr. '. $this->first_name . ' '. $this->middle_name . ' '. $this->last_name;
}

public function getFullNameShort(): string
{
   return $this->first_name[0] . '. '. $this->last_name;
}
```

- Avoid using the Environment variable directly in code using config both are the same but config gets data from the configuration.
-
- Use the request class for validation.
-
- Use timeout HTTP request.
-

- Use the chunk() function to break down heavy tasks.
-
- To specify `strict` we need to set `declare(strict_types=1);`

Null Safe operator

```php
<?php

// As of PHP 8.0.0, this line:
$result = $repository?->getUser(5)?->name;

// Is equivalent to the following code block:
if (is_null($repository)) {
    $result = null;
} else {
    $user = $repository->getUser(5);
    if (is_null($user)) {
        $result = null;
    } else {
        $result = $user->name;
    }
}
?>
```

- The null safe operator is best used when null is considered a valid and expected possible value for a property or method return. For indicating an error, a thrown exception is preferable.

GOTO

The goto operator can be used to jump to another section in the program. The target point is specified by a case-sensitive label followed by a colon, and the instruction is given as goto followed by the desired target label. This is not a full unrestricted goto. The target label must be within the same file and context, meaning that you cannot jump out of a function or method, nor can you jump into one. You also cannot jump into any sort of loop or switch structure. You may jump out of these, and a common use is to use a goto in place of a multi-level break.

```php
<?php
goto a;
echo 'Foo';
```

```
a:
echo 'Bar';
?>
```

NameSpace

PHP Namespaces provide a way in which to group related classes, interfaces, functions, and constants. Namespace names are case-insensitive

```
<?php
namespace myspace;
function hello() {
   echo "Hello World\n";
 }
Include namespace_file.php
use myspace;
myspace\hello();
?>
```

Enumerations: Enumerations or "Enums" allow a developer to define a custom type that is limited to one of a discrete number of possible values. That can be especially helpful when defining a domain model, as it enables "making invalid states unrepresentable."

```
<?php
enum Suit
{
   case Hearts;
   case Diamonds;
   case Clubs;
   case Spades;
}
?>
```

```
<?php
enum Suit: string
{
    case Hearts = 'H';
    case Diamonds = 'D';
```

```
    case Clubs = 'C';
    case Spades = 'S';
}
?>
```