**✅ Project Title: "Simple Logger with Reflection-based Inspector"**

**🎯 Project Goal:**

Create a system where multiple classes use a shared Logger trait to log actions, and then use PHP **Reflection** to inspect the classes and show:

* What methods they have
* What traits they use

**🧩 Project Structure**

pgsql

CopyEdit

/project

│

├── LoggerTrait.php

├── User.php

├── Admin.php

├── Inspector.php

└── index.php

**🧱 Step-by-Step Code**

**🟦 1. LoggerTrait.php**

php

CopyEdit

<?php

trait LoggerTrait {

public function log($msg) {

echo "[LOG]: $msg" . PHP\_EOL;

}

}

**🟦 2. User.php**

php

CopyEdit

<?php

require\_once 'LoggerTrait.php';

class User {

use LoggerTrait;

public function login($name) {

$this->log("User $name has logged in.");

}

public function logout($name) {

$this->log("User $name has logged out.");

}

}

**🟦 3. Admin.php**

php

CopyEdit

<?php

require\_once 'LoggerTrait.php';

class Admin {

use LoggerTrait;

public function createUser($name) {

$this->log("Admin created user: $name");

}

public function deleteUser($name) {

$this->log("Admin deleted user: $name");

}

}

**🟦 4. Inspector.php**

php

CopyEdit

<?php

class Inspector {

public static function inspectClass($className) {

$ref = new ReflectionClass($className);

echo "📦 Inspecting Class: $className\n";

echo "\n✅ Methods:\n";

foreach ($ref->getMethods() as $method) {

echo "- " . $method->getName() . "\n";

}

echo "\n🧩 Traits Used:\n";

$traits = $ref->getTraitNames();

foreach ($traits as $trait) {

echo "- " . $trait . "\n";

}

echo str\_repeat("=", 30) . "\n\n";

}

}

**🟦 5. index.php**

php

CopyEdit

<?php

require\_once 'User.php';

require\_once 'Admin.php';

require\_once 'Inspector.php';

$user = new User();

$user->login("Alice");

$user->logout("Alice");

$admin = new Admin();

$admin->createUser("Bob");

$admin->deleteUser("Bob");

Inspector::inspectClass('User');

Inspector::inspectClass('Admin');

**🧪 Output Example**

markdown

CopyEdit

[LOG]: User Alice has logged in.

[LOG]: User Alice has logged out.

[LOG]: Admin created user: Bob

[LOG]: Admin deleted user: Bob

📦 Inspecting Class: User

✅ Methods:

- login

- logout

- log

🧩 Traits Used:

- LoggerTrait

==============================

📦 Inspecting Class: Admin

✅ Methods:

- createUser

- deleteUser

- log

🧩 Traits Used:

- LoggerTrait

==============================

**📘 Learning Outcome**

* 🧠 Understand how to **reuse code using traits**
* 🔍 Learn how to use **ReflectionClass** to analyze code at runtime
* 🛠️ See a basic use case of traits and reflection working **together**