**Co to jest SOLID:**

Jest to zbiór praktyk dobrego programowania, który sprawia, że kod który tworzymy jest bardziej elastyczny i w łatwiejszy sposób można go rozwijać. Jedną z zasad SOLID jest m.in. zasada pojedynczej odpowiedzialności – czyli klasa powinna posiadać jedną główną funkcjonalność.

**Co to MVC**

Jest to wzorzec projektowy, czyli sposób projektowania aplikacji, który zakłada oddzielenie trzech głównych składowych projektu: wartwy modelu, wartwy widoku i wartstwy kontrolera. Oddzielenie tych części ma wiele zalet m.in. umożliwia niezależną pracę nad projektem dla wielu zespołów np. niezależną pracę nad front-endem oraz back-endem. Ułatwia także łatwiejszy dalszy rozwój projektu – zmieniając np. jedną część mechaniki aplikacji nie musimy ingerować w warstwę widoku.

**Czym różni się GET od POST**

GET wysyła żądanie do serwera w sposób jawny w postaci tzw. Query parameters. POST wysyła żądanie w sposób ukryty. Powszechnie przyjęło się, że GET powinniśmy używać podczas pobierania danych z serwera (np. w trakcie filtrowania wyników), a POST do wysyłania danych (np. wysłanie danych z formularza)

**Czym różni się interfejs od klasy abstrakcyjnej:**

Klasa abstrakcyjna może zawierać oprócz metod abstrakcyjnych również metody nie abstrakcyjne czyli w pełni zaimplementowane (w interfejsie mogą znaleźć się tylko deklaracje) . Klasa abstrakcyjna może zawierać właściwości i może używać modyfikatorów dostępu.

Klasa potomna może dziedziczyć tylko po jednej klasie abstrakcyjnej, natomiast może implementować wiele interfejsów.

**Co to jest elvis operator:**

To operator, który zwraca pierwszy operand jeżeli jego wartość jest prawdziwa, w przeciwnym wypadku zwraca drugi operand.

foo = bar ? bar : baz;

w tym przypadku zwrócona zostanie wartość ‘baz’ ponieważ ‘foo’ != ‘bar’

**Czym różni się self od $this**

$this odnosi się do pola w obiekcie danej konkretnej instancji klasy, self – odnosi się do pola w kontekście klasy.

**Jak aktualizować autoloader w composerze:**

composer dump-autoload

**Co to bundle w Symfony:**

Jest to tzw. plugin – paczka zawierająca dodatkowe funkcjonalności, które można zaimplementować w projekcie. Podstawowy projekt w Symfony zawiera kilka podstawowych Bundles niezbędnych do prawidłowego funkcjonowania, jeśli potrzebujemy w naszej aplikacji rozszerzyć jakąś funkcjonalność, to wgrywamy dodatkowy pakiet. Podstawowe Bundle z jakimi najczęściej pracuję to m.in. SecurityBundle do zarządzania systemem logowania użytkowników, lub FixtureBundle do wypełniania bazy danych jakimiś danymi.

**Co to dependency Injection:**

Jest to tzw. wstrzykiwanie zależności – wzorzec projektowy, który zakłada tworzenie powiązań między klasami. Jeżeli nasza klasa potrzebuje do swojego utworzenia jakiegoś obiektu to możemy przykładowo wstrzyknąć taki obiekt w konstruktorze, a framework sam utworzy instancję tego obiektu w tzw kontenerze.

**Co to Doctrine:**

Jest to tzw system ORM – Mapowanie Obiektowo Relacyjne - czyli system, który ma za zadanie przyspieszyć, usprawnić pracę z bazami danych. Doctrine zapewnia działania na bazach, tak jakbyśmy pracowali z obiektami – odwzorowuje strukturę danych z bazy danych (tabele, relacje) na obiekty.