1. OOP

A *class* is simply a representation of a type of *object*

In *OOP* the **encapsulation** is mainly achieved by creating classes, the classes expose public methods and properties

**Abstract** classes, which declared with the abstract keyword, cannot be instantiated. It can only be used as a super-class for other classes that extend the abstract class.

***Encapsulation*** means that a group of related properties, methods, and other members are treated as a single unit or object.

***Inheritance*** describes the ability to create new classes based on an existing class.

***Polymorphism*** means that you can have multiple classes that can be used interchangeably, even though each class implements the same properties or methods in different ways.

In one word, a reference of base class is assigned to a instance of sub class, and invoke the method of sub-class

1. *Authentication & Authorization*

***Authentication*** is knowing the identity of the user. For example, Alice logs in with her username and password, and the server uses the password to authenticate Alice.

***Authorization*** is deciding whether a user is allowed to perform an action. For example, Alice has permission to get a resource but not create a resource.

1. Overload & Override

**Overloading** is when you have multiple methods in the same scope, with the same name but different signatures.

**Overriding** is a principle that allows you to change the functionality of a method in a child class.

1. **What's difference between an Interface and Abstract class**

The same point is both of them cannot be instanced

The difference:

1. A class can implement **any number** of interfaces but a subclass can **at most use only one** abstract class
2. An abstract class can have non-abstract methods while in case of interface all the methods have to be abstract
3. An abstract can declare or use any variables, can have constructor and implementation methods, but interface just have implicitly public abstract method.

1. Keywords

Machine generated alternative text:
Keyword 
public 
private 
protected 
Internal 
Internal 
protected 
Description 
Public class is visible in the current and referencing 
assembly. 
Visible inside current class. 
Visible inside current and derived class. 
Visible inside containing assembly. 
Visible inside containing assembly and descendent of 
thecurrent class. 
Modifiers refine the declaration of a class. The list of all modifiers defined in the table are as follows; 
Modifier 
sealed 
static 
unsafe 
Abstract 
Description 
Class can't be inherited by a derived class. 
Class contains only static members. 
The class that has some unsafe construct likes pointers. 
The instance of the class is not created if the Class is 
abstract. 