Capgemini

C# Code Review Checklist

Pune, Abhilasha Mishra

Folders should be created to separate layers, components etc	
Unit Test cases are written and code coverage is > 70	√
Minimum of 3 tiers are created and data access code is not part of controller classes	
Interfaces are created for all classes	
If dependency injection is used, there should be no “new” objects getting created in code	
Generics are used and code is not repeated	
Methods are testable	
There are no build warnings	
Deprecated methods are not used	
Visual Studio Static code analysis has been run	
Unused “using” statements have been removed	
Null checks have been added for all nullable parameters	
Classes are following Single Responsibility Principle	
Method should not contains more than 30 lines of code	
Code should not extend beyond 100th column	
Boilerplate comments have been added to all public methods	
Try-catch has been implemented for all Controller(outermost layer) methods	

Nested if conditions should not be present	√
Lambda expressions are used	
Security checks to avoid 10 OWASP security issues are implemented	
Passwords are encrypted	
Variables are declared in config file and not hard coded in code	
Enums are created whenever applicable	
Any methods which have not been implemented yet, should throw “NotImplementedException”	
String.Concat & String.Format are used instead of “+”	
String.Empty is used in place of “”	
All third party packages are referred from Nuget. There should be no local references to packages from unclear origins	
There should not be any decision making based on “null” values	
No method should return null	
Logging has been implemented	
variable and parameter names must follow Camel casing and for class, function, and method names Pascal casing must be used	
Variable and method names should be self explanatory	