# Code Review

public class DuplicateChecker

{

public List<People> FindDuplicates(List<People> originals, List<People> newPeople)

{

List<People> d = new List<People>();

for (int x = 1; x < originals.Count; x++)

{

foreach(var np in newPeople)

{

if (np.Forename == originals[x].Forename &&

np.Surname == originals[x].Surname)

{

d.Add(np);

break;

}

}

}

return d;

}

}

public class People

{

public string Surname { get; set; }

public string Forename { get; set; }

public int Id { get; set; }

}

public class DuplicateCheckerTests

{

[TestMethod()]

public void FindDuplicatesTestFinds()

{

// arrange

People Kiko = new People() { Id = 1, Forename = "Kiko", Surname = "Cassilla" };

People Luke = new People() { Id = 2, Forename = "Luke", Surname = "Ayling" };

People Gani = new People() { Id = 3, Forename = "Gianni", Surname = "Allioski" };

People Liam = new People() { Id = 4, Forename = "Liam", Surname = "Cooper" };

People Ben = new People() { Id = 5, Forename = "Ben", Surname = "White" };

List<People> originals = new List<People>() { Kiko,Luke,Gani,Liam,Ben };

List<People> newPeople = new List<People>() { Gani };

var checker = new DuplicateChecker();

// act

var duplicates = checker.FindDuplicates(originals, newPeople);

// assert

Assert.IsTrue(duplicates.Contains(Gani));

}

}

CodeReview Points

1.Not followed proper tier architecture

2.Main() function is not implemented to call the method to execute

3.Null check not happened for List<people> and List<Original>.

4.Logs are not implemented

5.XML Comments not followed

6.Method Names should be Camel Cased

7.Interface not implemented

8.Dependency injection not followed

9.try and catch blocks not implemented

10.Name declaration for list in both method and test method is not proper.

11.using namespaces are not included

12.Model required fields check not implemented.

13.Process is not efficient. Processing time is very long

14.codecoverage not covered fully.