Trainee Code Challenge

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace TraineeCodeChallenge
{
    //Fill your code here
    public class CodeChallenge{

        [DatabaseGenerated(DatabaseGeneratedOption.None)]
        [Key]
        public int CodeChallengeId {get; set;}
        public string CodeChallengeName {get; set;}
        public ICollection<Trainee> TraineeList{get; set;}

        public CodeChallenge(){
            this.TraineeList = new List<Trainee>();
        }

    }
}
```

Program:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TraineeCodeChallenge
{
    class Program
    {
        static void Main(string[] args)
        {
            TraineeCodeChallengeRepository traineeCodeChallengeRepository = new
TraineeCodeChallengeRepository();

            int menu = 0;
            string menuRepeatLoopInput = string.Empty;
            do
            {
                Console.WriteLine("Menu:\nEnter 1 to insert New Trainees and Code
Challenge detail to database\n" +
                "Enter 2 to display all Trainees and all Code Challenges present in the
database\n" +
                "Enter 3 to enroll Trainees to Code Challenges\n" +
                "Enter 4 to search by trainee id and display the Trainee and his/her
enrolled Code Challenges");

                Console.WriteLine("Enter your menu choice:");
                menu = int.Parse(Console.ReadLine());
```

```
                switch (menu)
                {
                    case 1:
                        IList<Trainee> traineeInfo = new List<Trainee>
                        {
                            new Trainee{TraineeId=1001,TraineeName="Sam"},
                            new Trainee{TraineeId=1002,TraineeName="Merry"},
                            new Trainee{TraineeId=1003,TraineeName="Meghna"}
                        };
                        IList<Trainee> traineeListCheck =
traineeCodeChallengeRepository.DisplayTrainees();
                        if (traineeListCheck.Count == 0)
                        {
                            traineeCodeChallengeRepository.AddTrainees(traineeInfo);
                            Console.WriteLine("Trainee Details successfully saved in
database");
                        }
                        else
                        {
                            Console.WriteLine("Trainee Details are already present in
database");
                        }
                        IList<CodeChallenge> codeChallengeInfo = new
List<CodeChallenge>
                        {
                            new CodeChallenge{CodeChallengeId=1,CodeChallengeName="Web
UI"},
                            new CodeChallenge{CodeChallengeId=2,CodeChallengeName="C#"},
                            new CodeChallenge{CodeChallengeId=3,CodeChallengeName="MVC
5"}
                        };
                        IList<CodeChallenge> codeChallengeListCheck =
traineeCodeChallengeRepository.DisplayCodeChallenges();
                        if (codeChallengeListCheck.Count == 0)
                        {

traineeCodeChallengeRepository.AddCodeChallenges(codeChallengeInfo);
                            Console.WriteLine("Code Challenge Details successfully
saved in database");
                        }
                        else
                        {
                            Console.WriteLine("Code Challenge Details are already
present in database");
                        }
                        break;

                    case 2:

                        Console.WriteLine("Display list of Code Challenges:");
                        Console.WriteLine("{0,-20}{1}", "CC Id", "CC Name");
                        IList<CodeChallenge> codeChallengeList =
traineeCodeChallengeRepository.DisplayCodeChallenges();
                        if (codeChallengeList.Count > 0)
                        {
                            foreach (var codeChallenge in codeChallengeList)
                            {
                                Console.WriteLine("{0,-20}{1}",
codeChallenge.CodeChallengeId, codeChallenge.CodeChallengeName);
                            }
                        }
```

```
                        else
                        {
                            Console.WriteLine("No Code Challenge information is present
in the database");
                        }

                        Console.WriteLine("\nDisplay list of Trainees:");
                        Console.WriteLine("{0,-20}{1}", "Trainee Id", "Trainee Name");
                        IList<Trainee> traineeList =
traineeCodeChallengeRepository.DisplayTrainees();
                        if (traineeList.Count > 0)
                        {
                            foreach (var trainee in traineeList)
                            {
                                Console.WriteLine("{0,-20}{1}", trainee.TraineeId,
trainee.TraineeName);
                            }
                        }
                        else
                        {
                            Console.WriteLine("No Trainee Information is present in the
database");
                        }

                        break;

                    case 3:

                        string loopInput = "yes";
                        do
                        {
                            try
                            {
                                Console.WriteLine("Enroll Trainees to the Code
Challenges:");
                                Console.WriteLine("Enter Trainee Id: ");
                                int traineeId = int.Parse(Console.ReadLine());
                                Console.WriteLine("Enter Code Challenge Id in which the
trainee has to be enrolled: ");
                                int codeChallengeId = int.Parse(Console.ReadLine());
                                int i =
traineeCodeChallengeRepository.EnrollTraineeToCodeChallenge(traineeId, codeChallengeId);
                                if (i > 0)
                                {
                                    Console.WriteLine("Trainee Successfully enrolled to
Code Challenge");
                                }

                            }
                            catch
                            {
                                Console.WriteLine("The Trainee could not be registered
to Code Challenge. Check Trainee Id and Code Challenge Id");
                            }
                            Console.WriteLine("Press yes to continue..No to terminate");
                            loopInput = Console.ReadLine();
                        }
                        while (loopInput.Equals("yes",
StringComparison.InvariantCultureIgnoreCase));

                        break;
```

```
                    case 4:
                        Console.WriteLine("\nDisplay Trainee and the Code Challenges
the trainee have been enrolled");
                        Console.WriteLine("Enter Trainee Id whose information you want
to search:");
                        int searchTraineeId = int.Parse(Console.ReadLine());
                        Console.WriteLine("{0,-20}{1,-20}{2,-20}{3}", "Trainee Id",
"Trainee Name","CC Id", "CC Name");
                        IList<Trainee> traineeListInfo =
traineeCodeChallengeRepository.SearchTraineeById(searchTraineeId);
                        if (traineeListInfo.Count > 0)
                        {
                            //Fill your code here to display the trainee and his/her
enrolled code challenges

traineeCodeChallengeRepository.SearchTraineeById(searchTraineeId);
                        }
                        else
                        {
                            Console.WriteLine("The searched Trainee has not been
enrolled to any Code Challenges");
                        }
                        break;
                }
                Console.WriteLine("Press Yes to repeat menu..Any other key to
terminate");
                menuRepeatLoopInput = Console.ReadLine();
            }
            while
(menuRepeatLoopInput.Equals("yes",StringComparison.InvariantCultureIgnoreCase));

            Console.WriteLine("Thank you. Have a nice day");
        }
    }
}
```

Trainee:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace TraineeCodeChallenge
{
   //Fill your code here
   public class Trainee{

       [DatabaseGenerated(DatabaseGeneratedOption.None)]
       [Key]
       public int TraineeId{get; set;}
       public string TraineeName{get; set;}
       public ICollection<CodeChallenge> CodeChallengeList {get; set;}

      public Trainee(){
          this.CodeChallengeList = new List<CodeChallenge>();
      }
```

```
   }
}
```

TraineeCodeChallengeContext:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data.Entity;

namespace TraineeCodeChallenge
{
    public class TraineeCodeChallengeContext:DbContext
    {
        public TraineeCodeChallengeContext():base("DataConnection")
        {

        }

          //Fill your code here to create the DbSets
          public DbSet<Trainee> Trainees{get; set;}
          public DbSet<CodeChallenge> CodeChallenges {get; set;}

    }
}
```

TraineeCodeChallengeRepository:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;


namespace TraineeCodeChallenge
{

    //Fill your code here
    public class TraineeCodeChallengeRepository{
        TraineeCodeChallengeContext ctx = new TraineeCodeChallengeContext();

        public int AddCodeChallenges(IList<CodeChallenge> CodeChallengeList){
            foreach(CodeChallenge Cs in CodeChallengeList){
                ctx.CodeChallenges.Add(Cs);
            }

            ctx.SaveChanges();
            return CodeChallengeList.Count;
        }

        public int AddTrainees(IList<Trainee> traineeList){
            foreach(var trainee in traineeList){
                ctx.Trainees.Add(trainee);
            }
            ctx.SaveChanges();

            return traineeList.Count;
```

```
        }

        public IList<CodeChallenge> DisplayCodeChallenges(){
            List<CodeChallenge> details = ctx.CodeChallenges.ToList<CodeChallenge>();

            return details;
        }

        public IList<Trainee> DisplayTrainees(){
            IList<Trainee> allTrainees = ctx.Trainees.ToList<Trainee>();
            return allTrainees;
        }

        public int EnrollTraineeToCodeChallenge(int traineeId, int codeChallengeId){
            CodeChallenge obj = ctx.CodeChallenges.Find(codeChallengeId);
            Trainee tr = ctx.Trainees.Find(traineeId);
            tr.CodeChallengeList.Add(obj);

            if(ctx.SaveChanges()>0){
                return 1;
            }
            else{
                return 0;
            }
        }

        public List<Trainee> SearchTraineeById(int traineeid){
            List<Trainee> T = new List<Trainee>();
            Trainee Tr = ctx.Trainees.Find(traineeid);
            if(Tr==null){
                Console.WriteLine("The searched Trainee has not been enrolled to any
Code Challenges");
            }


            return T;

        }
    }
}
```