UNIT TESTING

1. **Controller**:

private Mock<ICourseService<Course>> CourseServiceObj;

        private Mock<IEnrollmentService<Enrollment>> EnrollmentServiceObj;

        private CoursesController courseCtrlr;

        private Course c = new Course

        {

            CourseId = 119,

            CourseTitle= "MPP",

            CourseCredits = 2

        };

        private List<Course> courses = new List<Course>() {

            new Course

            {

                CourseId = 119,

                CourseTitle= "MPP",

                CourseCredits = 2

            },

            new Course

            {

                CourseId = 120,

                CourseTitle= "Algebra",

                CourseCredits = 3

            }

        };

        [SetUp]

        public void Setup()

        {

            CourseServiceObj = new Mock<ICourseService<Course>>();

            EnrollmentServiceObj = new Mock<IEnrollmentService<Enrollment>>();

            courseCtrlr = new CoursesController(CourseServiceObj.Object, EnrollmentServiceObj.Object);

        }

        [Test]

        public void GetAllCourses\_Test()

        {

            CourseServiceObj.Setup(x=>x.GetAll()).Returns(courses);

            var result = courseCtrlr.GetAllCourses();

            Assert.That(result, Is.InstanceOf<Task<ActionResult<IEnumerable<Course>>>>());

        }

        [TestCase(119)]

        public void GetGetCourseById\_Test(int id)

        {

            CourseServiceObj.Setup(\_ => \_.GetById(id)).Returns(c);

            Task result = courseCtrlr.GetCourseById(119);

            Assert.That(result,Is.InstanceOf<Task<ActionResult<Course>>>());

            Task<ActionResult<Course>> taskResult = (Task<ActionResult<Course>>)result;

            System.Diagnostics.Debug.WriteLine(taskResult.Result.Value);

            Assert.That(taskResult.Result.Value,Is.AssignableTo<Course>());

        }

1. **Service:**

private Mock<ICourseRepo<Course>> CourseRepoObj;

        private CourseService courseServ;

        private Course c = new Course

        {

            CourseId = 119,

            CourseTitle = "MPP",

            CourseCredits = 2

        };

        private List<Course> courses = new List<Course>() {

            new Course

            {

                CourseId = 119,

                CourseTitle= "MPP",

                CourseCredits = 2

            },

            new Course

            {

                CourseId = 120,

                CourseTitle= "Algebra",

                CourseCredits = 3

            }

        };

        [SetUp]

        public void Setup()

        {

            CourseRepoObj = new Mock<ICourseRepo<Course>>();

            CourseRepoObj.Setup(x => x.GetAll()).Returns(courses);

            courseServ = new CourseService(CourseRepoObj.Object);

        }

        [Test]

        public void GetAll\_Test()

        {

            var result = courseServ.GetAll();

            // To print something to debug window

            //System.Diagnostics.Debug.WriteLine(result);

            Assert.AreEqual(2,result.Count);

        }

    }

1. **Repository:**

private List<Course> courses = new List<Course>();

        IQueryable<Course> courseData;

        Mock<DbSet<Course>> mockSet;

        Mock<API\_CEContext> mockAPIContext;

        CourseRepo courseRepo;

        [SetUp]

        public void Setup()

        {

            courses = new List<Course>() {

                new Course

                {

                    CourseId = 119,

                    CourseTitle= "MPP",

                    CourseCredits = 2

                },

                new Course

                {

                    CourseId = 120,

                    CourseTitle= "Algebra",

                    CourseCredits = 3

                }

            };

            courseData = courses.AsQueryable();

            mockSet = new Mock<DbSet<Course>>();

            mockSet.As<IQueryable<Course>>().Setup(m => m.Provider).Returns(courseData.Provider);

            mockSet.As<IQueryable<Course>>().Setup(m => m.Expression).Returns(courseData.Expression);

            mockSet.As<IQueryable<Course>>().Setup(m => m.ElementType).Returns(courseData.ElementType);

            mockSet.As<IQueryable<Course>>().Setup(m => m.GetEnumerator()).Returns(courseData.GetEnumerator());

            var p = new DbContextOptions<API\_CEContext>();

            mockAPIContext = new Mock<API\_CEContext>(p);

            mockAPIContext.Setup(x => x.Courses).Returns(mockSet.Object);

            courseRepo = new CourseRepo(mockAPIContext.Object);

        }

        [Test]

        public void GetCourseById\_Test()

        {

            var result = courseRepo.GetById(119);

            string title = result.CourseTitle;

            Assert.AreEqual("MPP", title);

        }

    }