شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Web Application Programming Interface (API)

Tahaluf Training Center 2021

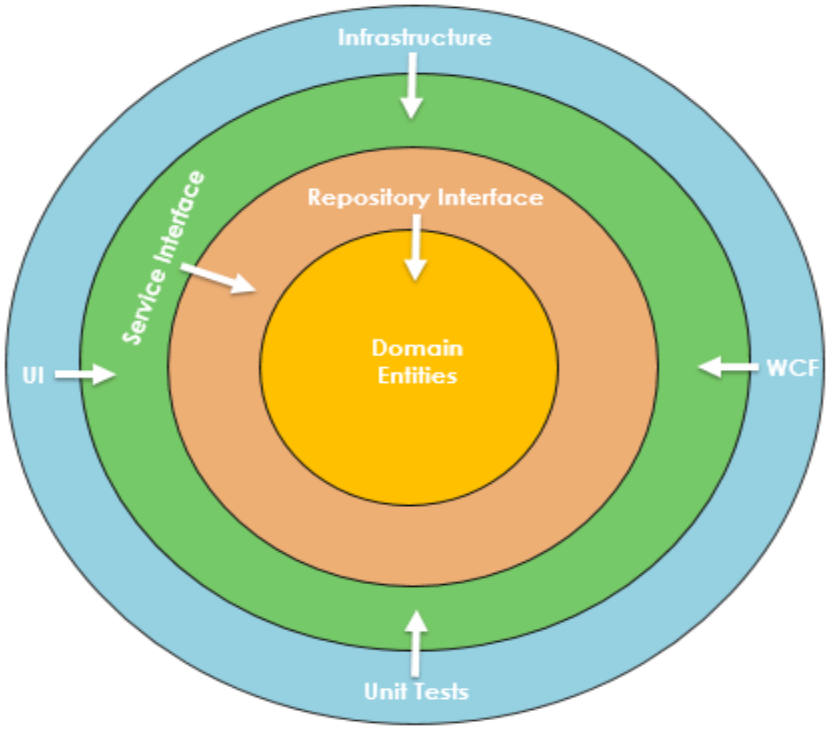
شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Chapter 03

Overview of Onion Architecture

Onion Architecture is used to overcome both separation of concern and tightly coupling issues in our application.



Overview of Onion Architecture

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Onion Architecture contains four layers:

- Domain Layer.
- Repository Layer.
- Service Layer.
- UI/Presentation Layer.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Domain/Data Access Layer

This is the core layer of the application, the classes in this layer are used to create a table in the database, basically this layer is used to build an entity.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Repository Layer

This is a generic repository layer which acts as an abstract layer between the data access and business layer of the application which makes a more loosely coupled approach to data access.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Service Layer

This layer is used communicate between UI and repository layers using Interface. We can call it as a business layer since it holds the business logic for an entity.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

UI Layer

The UI layer can be Web API or Web application or any other UI application. This layer contains the implementation of DI (Dependency Inversion) principle to build a loosely coupled application.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Advantages of Onion Architecture:

There are several advantages of the Onion Architecture, as listed below:

- ✓ It provides better maintainability as all the codes depend on layers or the center.
- ✓ It provides better testability as the unit test can be created for separate layers without an effect of other modules of the application.

Overview of Onion Architecture

- ✓ It develops a loosely coupled application as the outer layer of the application always communicates with the inner layer via interfaces.
- ✓ Any concrete implantation would be provided to the application at run time.
- ✓ Domain entities are core and center part. It can have access to both the database and UI layers.
- ✓ The internal layers never depend on the external layer. The code that may have changed should be part of an external layer.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Chapter 03

1 Overview of Onion Architecture

2 Overview of Class Library

3 Create a Class Library

4 Create a Services in ASP.Net Web API

Overview of Class Library

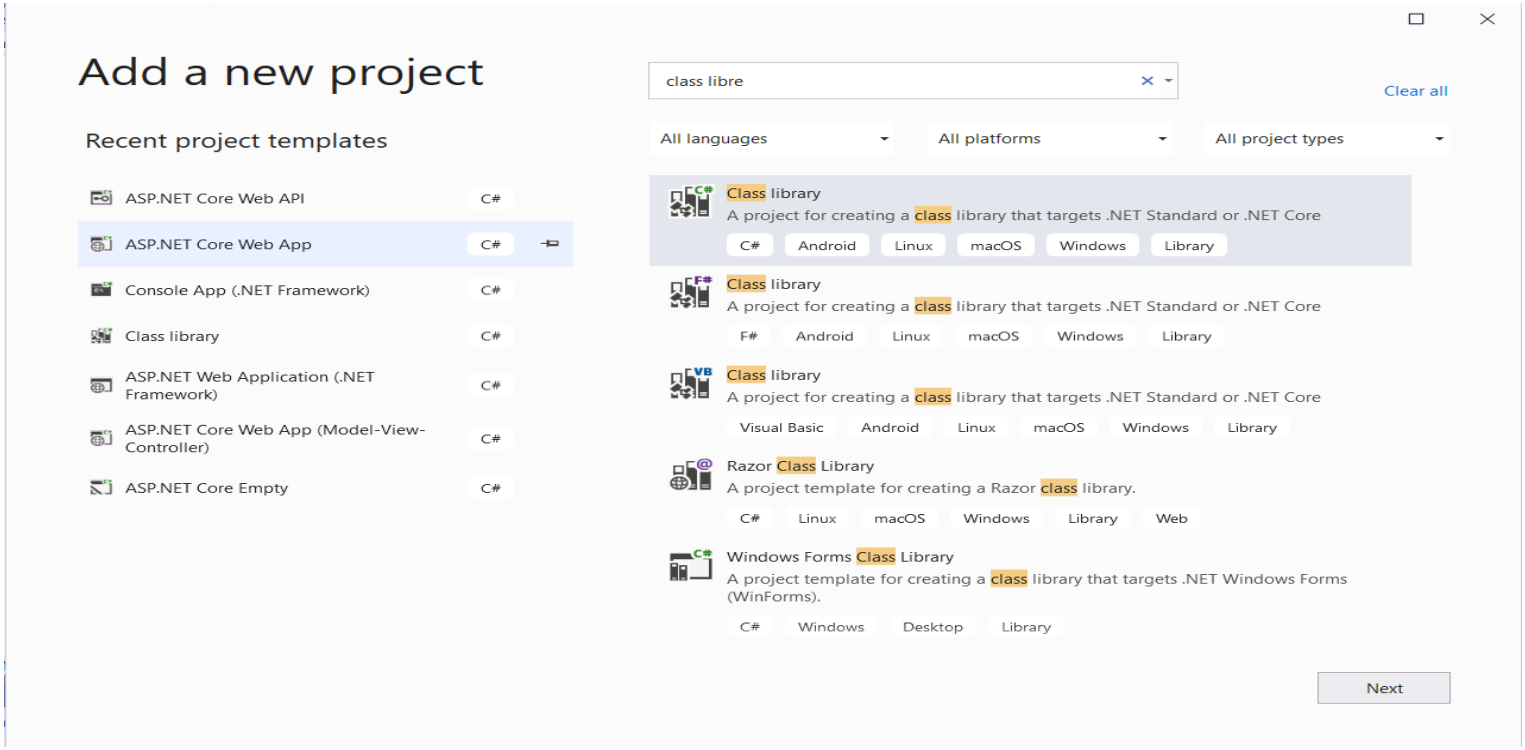
A class library is a package of programs (code that has classes, types, interfaces, and other program elements) that is easily distributable, shareable, and reusable by other developers who want to implement the same functionality. Physically, a class library is a .dll (dynamic link library) file.

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Chapter 03

Create a Class Library

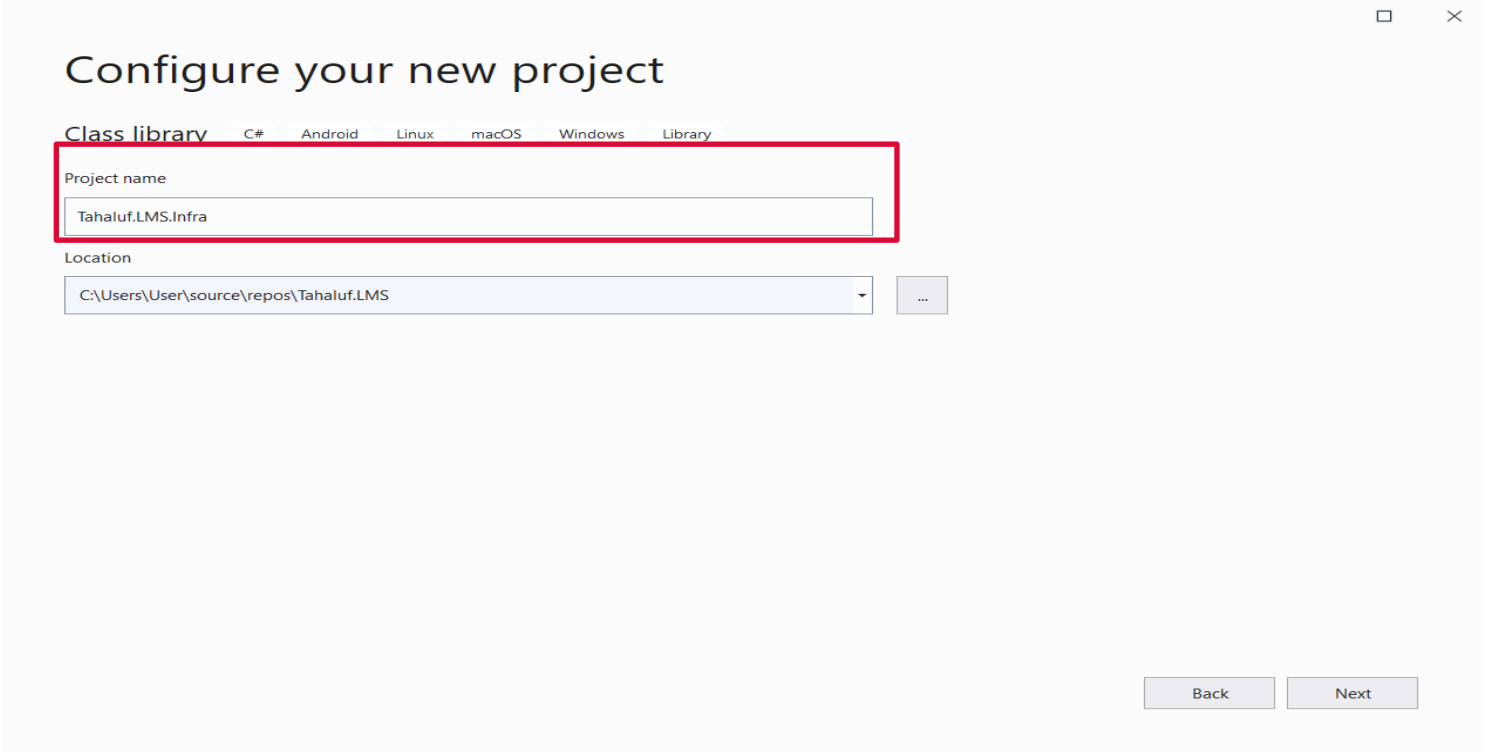
To Create .NET Core Class Library => Right Click on Solution Name (Tahaluf.LMS) => Add => New Project => Class library



Create a Class Library

شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

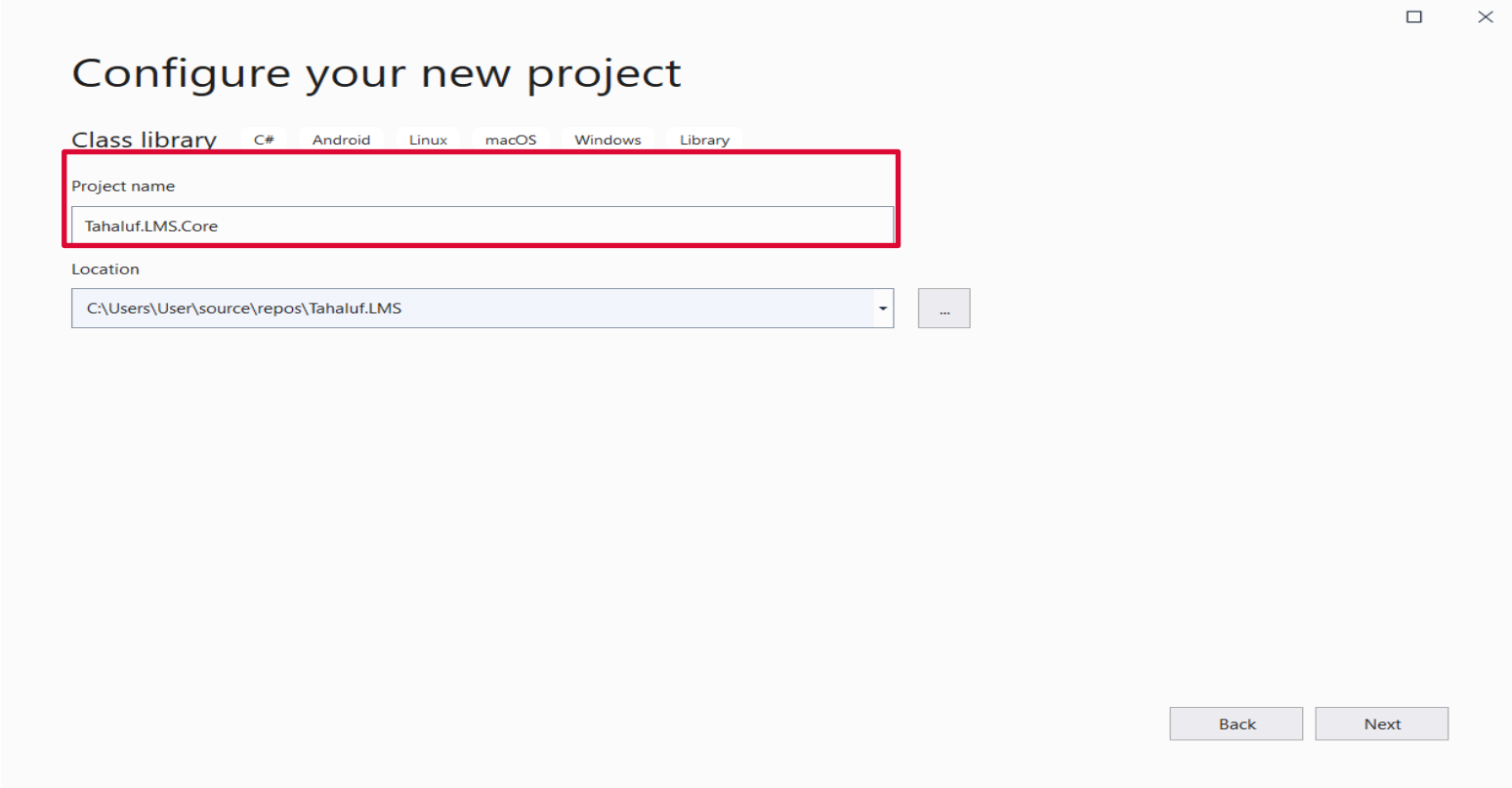
Enter Project Name



Create a Class Library

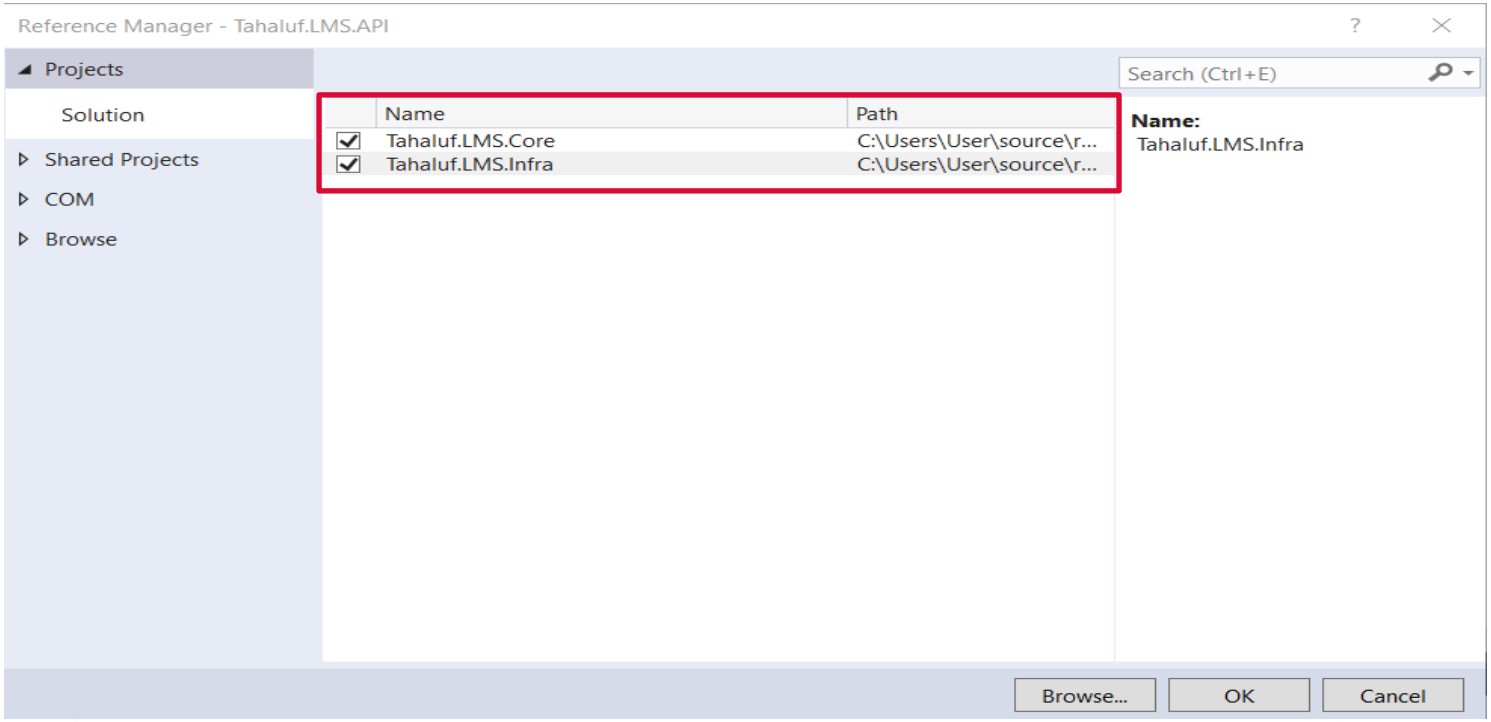
شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Enter Project Name



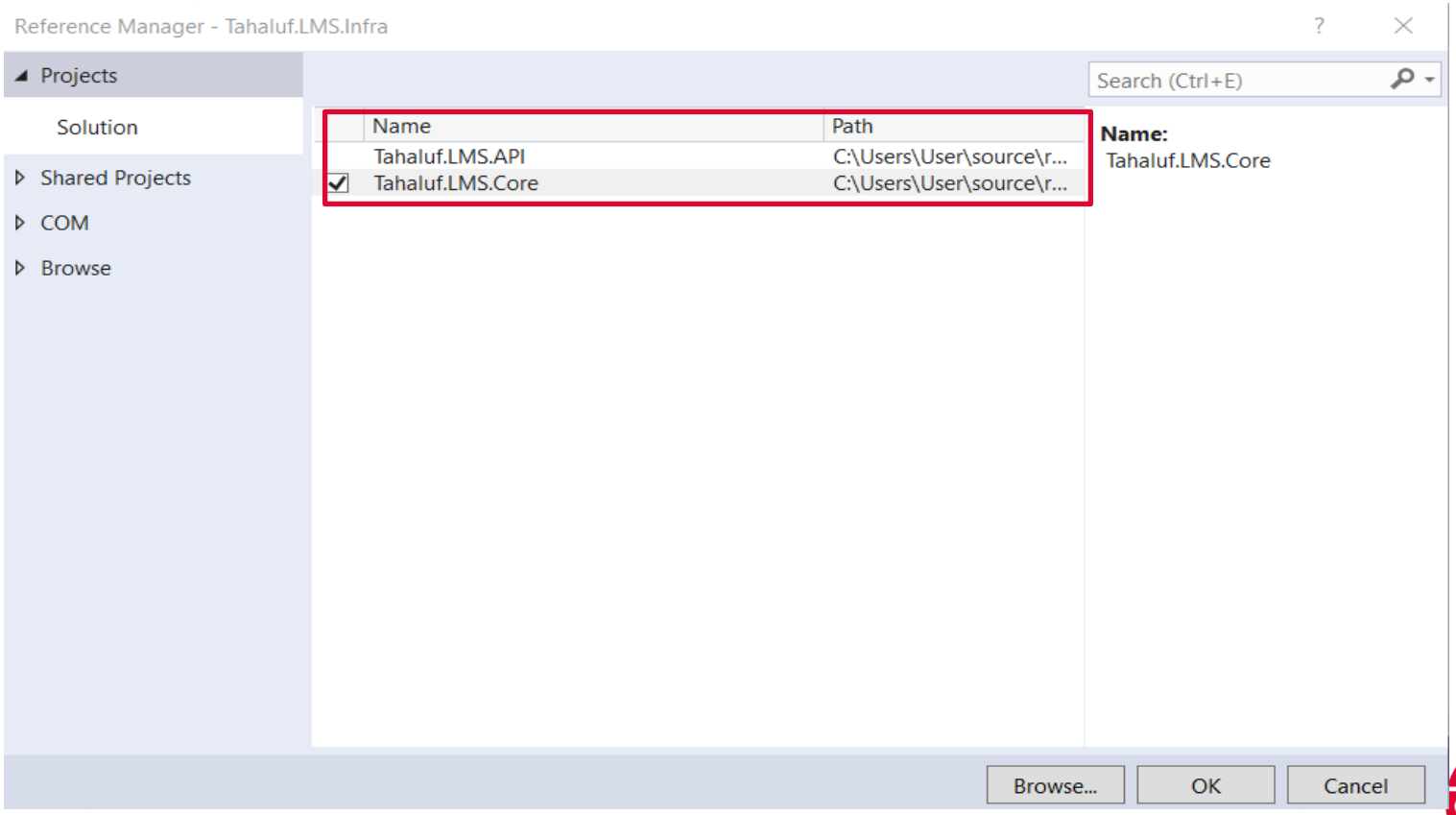
Create a Class Library

Right Click on Dependencies => Add Project Reference => Solution => Select check.

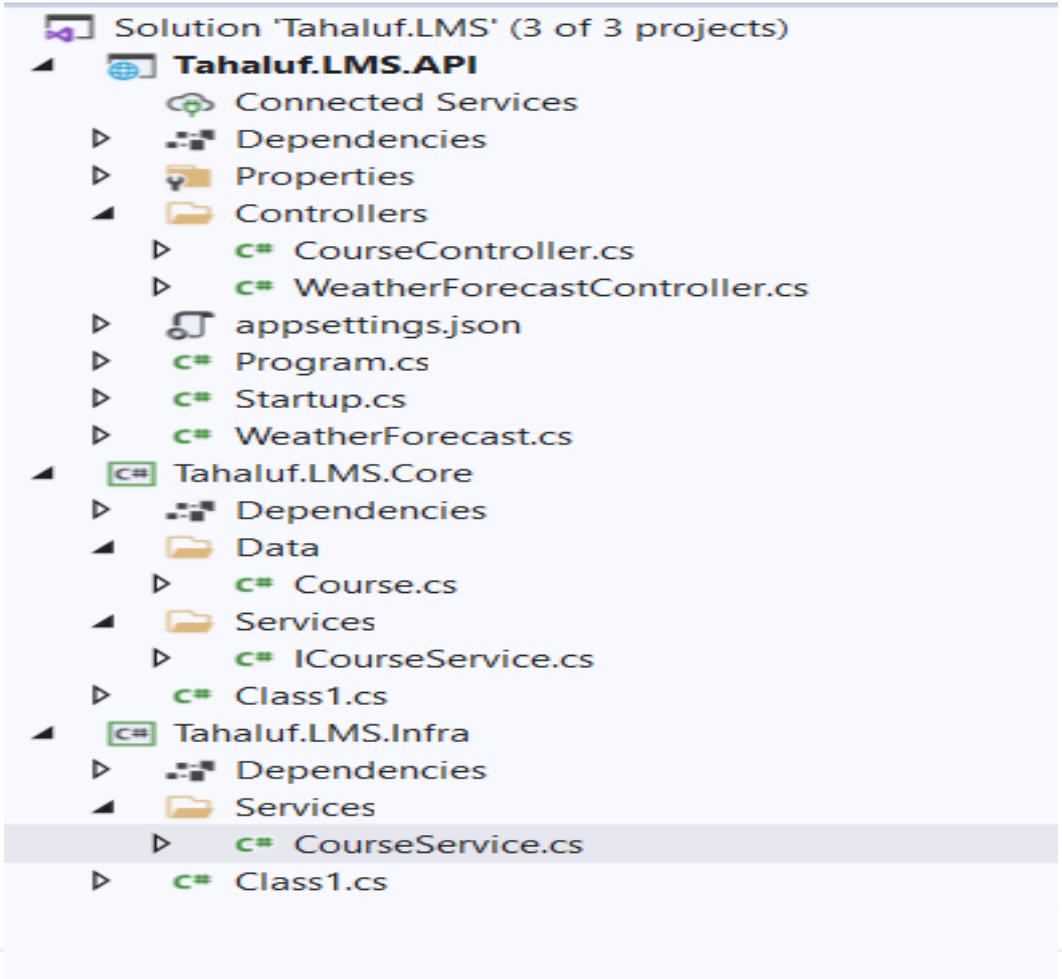


Create a Class Library

Right Click on Dependencies => Add Project Reference => Solution => Select check.



Create a Class Library



شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Chapter 03

1. Overview of Onion Architecture
2. Overview of Class Library
3. Create a Class Library
4. Create a Services in ASP.Net Web API

Create a Course Class:

```
public class Course
{
    public int CourseId { get; set; }
    public string CourseName { get; set; }
    public DateTime CreateDate { get; set; }
    public string Category { get; set; }
}
```

Create a Course Service:

```
public class CourseService : ICourseService
 {
    private static readonly List<Course> Courses = new
    List<Course>
    {
        new Course()
        {
            CourseId=1,
            CourseName="Math101",
            CreateDate= new DateTime(),
            Category="Math"
        },
```

Create a Services in ASP.Net Web API

```
    new Course()
    {
        CourseId=2,
        CourseName="Stat101",
        CreateDate= new DateTime(),
        Category="Stat"
    },
      new Course()
    {
        CourseId=3,
        CourseName="English101",
        CreateDate= new DateTime(),
        Category="English"
    },
};
```

Create a Services in ASP.Net Web API

```
public List<Course> GetAll()
{
   return Courses;
}
public Course Create(Course Course)
{
    return Course;
}

public Course Update(Course Course)
{
    return Course;
}
```

Create ICourseService interface:

```
public interface ICourseService
{
    List<Course> GetAll();
    Course Create(Course course);
    Course Update(Course course);
}
```

Create Course Controller:

```
public class CourseController : Controller
{
 private readonly ICourseService CourseService;
 public CourseController(ICourseService courseService)
 {
  CourseService = courseService;
 }
```

Create a Services in ASP.Net Web API

```
[HttpGet]
[ProducesResponseType(typeof(List<Course>),
StatusCodes.Status200OK)]
public List<Course> GetAll()
{
    return CourseService.GetAll();
}

[HttpPost]
[ProducesResponseType(typeof(List<Course>),
StatusCodes.Status200OK)]
[ProducesResponseType(typeof(List<Course>),
StatusCodes.Status400BadRequest)]
public Course Create([FromBody] Course Course)
{
    return CourseService.Create(Course);
}
```

```
[HttpPut]
[ProducesResponseType(typeof(List<Course>),
StatusCodes.Status200OK)]
[ProducesResponseType(typeof(List<Course>),
StatusCodes.Status400BadRequest)]
public Course Update([FromBody] Course Course)
{
    return CourseService.Update(Course);
}
```

In Startup:

```
services.AddScoped<ICourseService, CourseService>();
```