Working with TeamService

(Install .Net core sdk first)

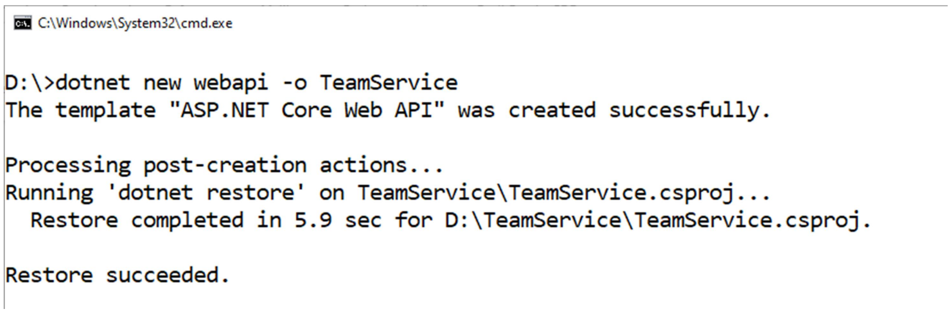
Link: https://dotnet.microsoft.com/learn/dotnet/hello-world-tutorial/install

1) **Create new project:**

   Command :

   ```
   dotnet new webapi -o TeamService
   ```

   output:



C:\Windows\System32\cmd.exe

```
D:\>dotnet new webapi -o TeamService
The template "ASP.NET Core Web API" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on TeamService\TeamService.csproj...
  Restore completed in 5.9 sec for D:\TeamService\TeamService.csproj.

Restore succeeded.
```

2) **Remove existing weatherforecast files both model and controller files.**
3) **Add new files as follows:**
4) **Add Member.cs to “D:\TeamService\Models” folder**

```
using System;
namespace TeamService.Models
{    public class Member {
        public Guid ID { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public Member() {       }
        public Member(Guid id) : this()
         {
           this.ID = id;
        }
        public Member(string firstName, string lastName, Guid id) : this(id)
        {
           this.FirstName = firstName;
           this.LastName = lastName;
        }
        public override string ToString() {
           return this.LastName;
        }
     }
}
```

5) **Add Team.cs to “D:\TeamService\Models” folder**

```
using System;
using System.Collections.Generic;
namespace TeamService.Models
{    public class Team
   {
        public string Name { get; set; }
        public Guid ID { get; set; }
        public ICollection<Member> Members { get; set; }
```

```
        public Team()
        {
           this.Members = new List<Member>();
        }
        public Team(string name) : this()
        {
           this.Name = name;
        }
        public Team(string name, Guid id)  : this(name)
        {
           this.ID = id;
        }
        public override string ToString() {
           return this.Name;
        }
      }
    }
```

3)add TeamsController.cs file to "D:\TeamService\Controllers" folder

```
using System;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;
using System.Linq;
using TeamService.Models;
using System.Threading.Tasks;
using TeamService.Persistence;

namespace TeamService
{           [Route("[controller]")]
            public class TeamsController : Controller
            {           ITeamRepository repository;
                        public TeamsController(ITeamRepository repo)
                        {
                         repository = repo;
                        }
                        [HttpGet]
                    public virtual IActionResult GetAllTeams()
                        {
                                    return this.Ok(repository.List());
                        }

                        [HttpGet("{id}")]
                  public IActionResult GetTeam(Guid id)
                  {    Team team = repository.Get(id);
                        if (team != null) // I HATE NULLS, MUST FIXERATE THIS.
                        {  return this.Ok(team);
                        }
```

```
                else {
                        return this.NotFound();
                }
        }
        [HttpPost]
        public virtual IActionResult CreateTeam([FromBody]Team newTeam)
        {
                repository.Add(newTeam);
                return this.Created($"/teams/{newTeam.ID}", newTeam);
        }

        [HttpPut("{id}")]
        public virtual IActionResult UpdateTeam([FromBody]Team team, Guid id)
        {       team.ID = id;
                if(repository.Update(team) == null)
                {
                        return this.NotFound();
                }
                else
                {
                        return this.Ok(team);
                }
        }

        [HttpDelete("{id}")]
             public virtual IActionResult DeleteTeam(Guid id)
         {      Team team = repository.Delete(id);
                if (team == null)
                 {
                return this.NotFound();
                }
                else {
                        return this.Ok(team.ID);
                        }
                }
        }
}
```

4) add MembersController.cs file to "D:\TeamService\Controllers" folder

```
using System;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;
using System.Linq;
using TeamService.Models;
using System.Threading.Tasks;
using TeamService.Persistence;
```

```
namespace TeamService
{           [Route("/teams/{teamId}/[controller]")]
            public class MembersController : Controller
            {           ITeamRepository repository;
                        public MembersController(ITeamRepository repo)
                        {
                                    repository = repo;
                        }
                        [HttpGet]
                        public virtual IActionResult GetMembers(Guid teamID)
                        {
                                    Team team = repository.Get(teamID);
                                    if(team == null)
                                    {
                                                return this.NotFound();
                                    }
                                    else {
                                                return this.Ok(team.Members);
                                    }
                        }
                        [HttpGet]
                        [Route("/teams/{teamId}/[controller]/{memberId}")]
                        public virtual IActionResult GetMember(Guid teamID, Guid memberId)
                        {           Team team = repository.Get(teamID);
                                    if(team == null)
                                    {
                                                return this.NotFound();
                                    }
                                     else
                                    {
                                                var q = team.Members.Where(m => m.ID == memberId);
                                                if(q.Count() < 1)
                                                {
                                                            return this.NotFound();
                                                }
                                                else
                                                {
                                                            return this.Ok(q.First());
                                                }
                                    }
                        }

                        [HttpPut]
                        [Route("/teams/{teamId}/[controller]/{memberId}")]
public virtual IActionResult UpdateMember([FromBody]Member updatedMember, Guid teamID, Guid memberId)
                        {    Team team = repository.Get(teamID);
                        if(team == null)
                          { return this.NotFound();
                        }
```

```
		else {
			var q = team.Members.Where(m => m.ID == memberId);
			if(q.Count() < 1)
			{
				return this.NotFound();
			}
			else {
				team.Members.Remove(q.First());
				team.Members.Add(updatedMember);
				return this.Ok();
			}
		}
	}

	[HttpPost]
	public virtual IActionResult CreateMember([FromBody]Member newMember, Guid teamID)
	{
		Team team = repository.Get(teamID);
		if(team == null)
		{
			return this.NotFound();
		}
		else {
			team.Members.Add(newMember);
			var teamMember = new {TeamID = team.ID, MemberID = newMember.ID};
return this.Created($"/teams/{teamMember.TeamID}/[controller]/{teamMember.MemberID}", teamMember);
		}
	}

	[HttpGet]
	[Route("/members/{memberId}/team")]
	public IActionResult GetTeamForMember(Guid memberId)
	{
		var teamId = GetTeamIdForMember(memberId);
		if (teamId != Guid.Empty)
		{
			return this.Ok(new {TeamID = teamId });
		}
		else {
			return this.NotFound();
		}
	}

	private Guid GetTeamIdForMember(Guid memberId)
	{	foreach (var team in repository.List())
		{	var member = team.Members.FirstOrDefault( m => m.ID == memberId);
			if (member != null)
			{   return team.ID;
			}
```

```
                    }
                    return Guid.Empty;
            }
        }
}
```

5) create folder “D:\TeamService\Persistence”:

6)add file ITeamReposiroty.cs in “D:\TeamService\Persistence” folder

```
using System;
using System.Collections.Generic;
using TeamService.Models;
namespace TeamService.Persistence
{
        public interface ITeamRepository
        {
          IEnumerable<Team> List();
                Team Get(Guid id);
                Team Add(Team team);
                Team Update(Team team);
                Team Delete(Guid id);
        }
}
```

7)Add MemoryTeamRepository.cs in “D:\TeamService\Persistence” folder

```
using System;
using System.Collections.Generic;
using System.Linq;
using TeamService;
using TeamService.Models;

namespace TeamService.Persistence
{
        public class MemoryTeamRepository :  ITeamRepository
        {
                protected static ICollection<Team> teams;
                public MemoryTeamRepository() {
                        if(teams == null) {
                                teams = new List<Team>();
                        }
                }

        public MemoryTeamRepository(ICollection<Team> teams)
        {
               MemoryTeamRepository.teams = teams;
        }
        public IEnumerable<Team> List()
                {
                        return teams;
                }
        public Team Get(Guid id)
        {
            return teams.FirstOrDefault(t => t.ID == id);
        }

        public Team Update(Team t)
        {
            Team team = this.Delete(t.ID);
            if(team != null)
            {
                team = this.Add(t);
            }
            return team;
        }

        public Team Add(Team team)
        {
            teams.Add(team);
            return team;
        }

        public Team Delete(Guid id)
        {
            var q = teams.Where(t => t.ID == id);
            Team team = null;
            if (q.Count() > 0)
            {
                team = q.First();
                teams.Remove(team);
            }

            return team;
        }
    }
}
```

8) add following line to Startup.cs in public void ConfigureServices(IServiceCollection services) method

```
    services.AddScoped<ITeamRepository, MemoryTeamRepository>();
```

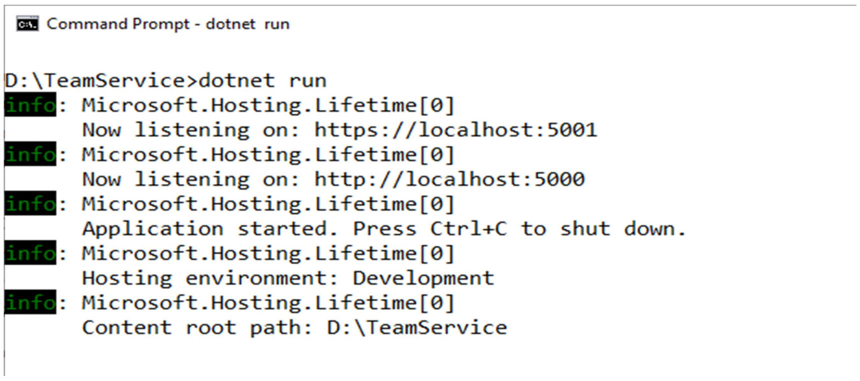
9) Now open two command prompts to run this project

10) On Command prompt 1: (go inside folder teamservice first)

Commands:

```
    dotnet run
```

Output:



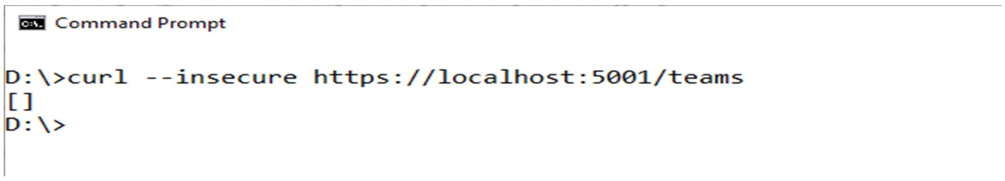
```
Command Prompt - dotnet run

D:\TeamService>dotnet run
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: https://localhost:5001
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: D:\TeamService
```

11)On command prompt 2

Command: To get all teams

```
curl --insecure https://localhost:5001/teams
```

output:

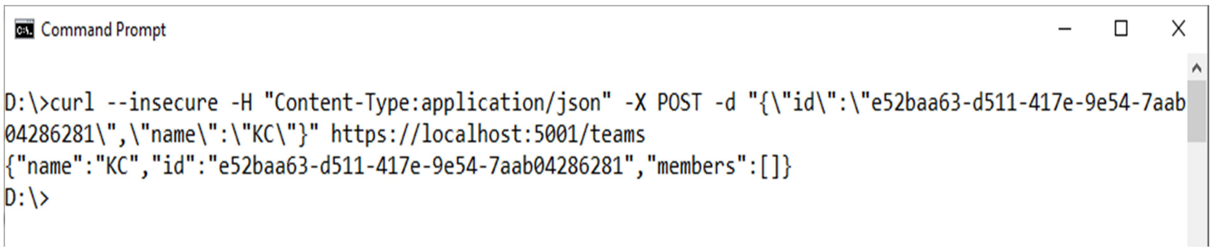


```
Command Prompt

D:\>curl --insecure https://localhost:5001/teams
[]
D:\>
```

Command : To create new team

curl --insecure -H “Content-Type:application/json” –X POST –d “{\“id\”:\”e52baa63-d511-417e-9e54-7aab04286281\”, \”name\”:\”KC\”}” https://localhost:5001/teams

output:

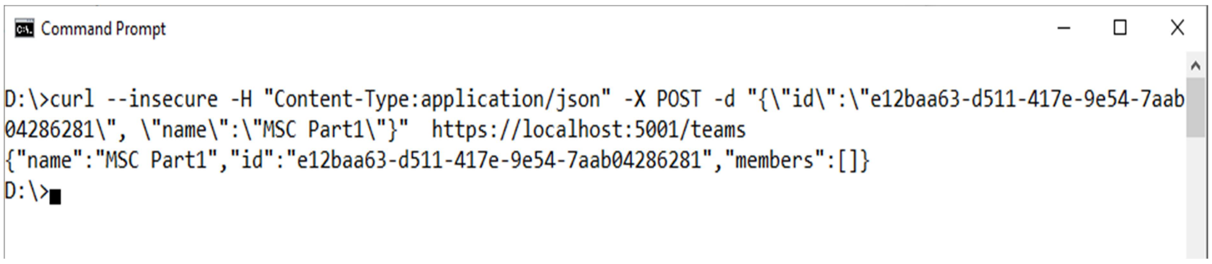


```
Command Prompt

D:\>curl --insecure -H "Content-Type:application/json" -X POST -d "{\"id\":\"e52baa63-d511-417e-9e54-7aab
04286281\",\"name\":\"KC\"}" https://localhost:5001/teams
{"name":"KC","id":"e52baa63-d511-417e-9e54-7aab04286281","members":[]}
D:\>
```

Command : To create one more new team

curl --insecure -H “Content-Type:application/json” –X POST –d “{\“id\”:\”e12baa63-d511-417e-9e54-7aab04286281\”, \”name\”:\”MSC Part1\”}” https://localhost:5001/teams

output:

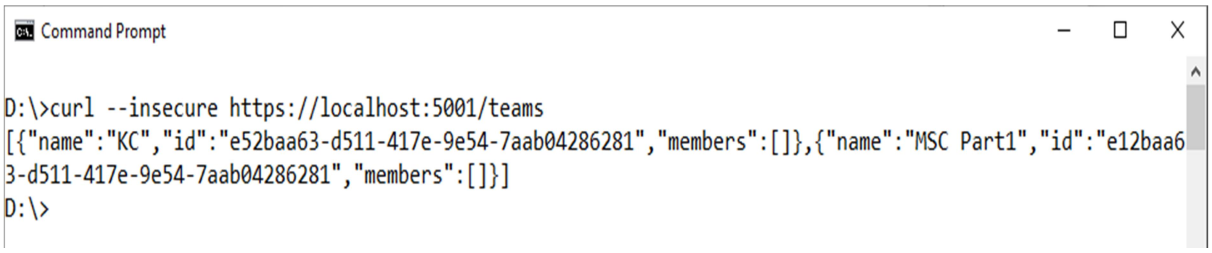


```
Command Prompt

D:\>curl --insecure -H "Content-Type:application/json" -X POST -d "{\"id\":\"e12baa63-d511-417e-9e54-7aab
04286281\", \"name\":\"MSC Part1\"}"  https://localhost:5001/teams
{"name":"MSC Part1","id":"e12baa63-d511-417e-9e54-7aab04286281","members":[]}
D:\>
```

Command : To get all teams

```
curl --insecure https://localhost:5001/teams
```

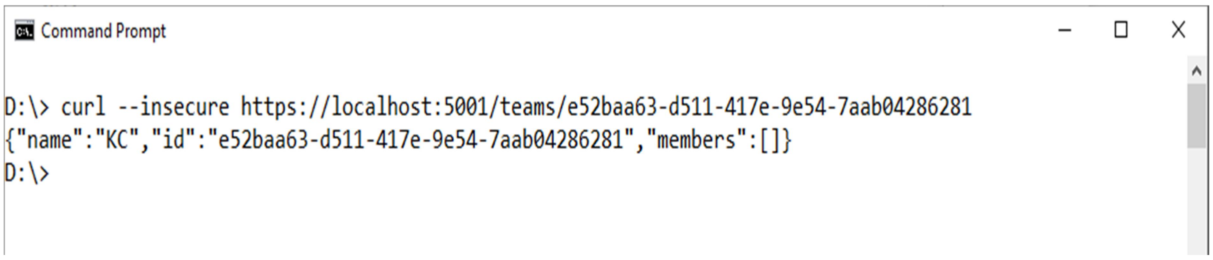
Output:



```
Command Prompt

D:\>curl --insecure https://localhost:5001/teams
[{"name":"KC","id":"e52baa63-d511-417e-9e54-7aab04286281","members":[]},{"name":"MSC Part1","id":"e12baa6
3-d511-417e-9e54-7aab04286281","members":[]}]
D:\>
```

Command : to get single team with team-id as parameter
curl --insecure https://localhost:5001/teams/e52baa63-d511-417e-9e54-7aab04286281

output:



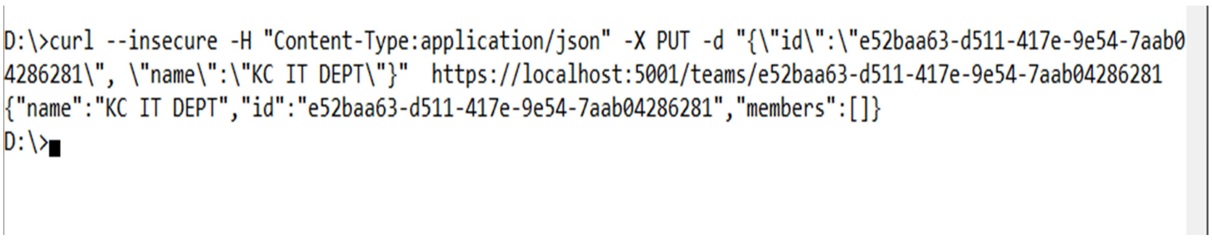
Command Prompt

```
D:\> curl --insecure https://localhost:5001/teams/e52baa63-d511-417e-9e54-7aab04286281
{"name":"KC","id":"e52baa63-d511-417e-9e54-7aab04286281","members":[]}
D:\>
```

Command : to update team details (change name of first team from "KC" to "KC IT DEPT")

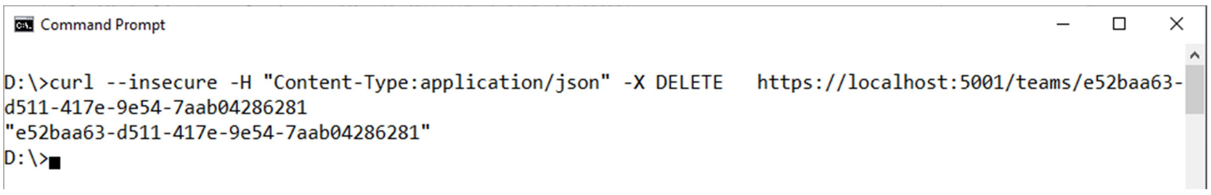
curl --insecure -H "Content-Type:application/json" –X PUT –d "{\"id\":\"e52baa63-d511-417e-9e54-7aab04286281\", \"name\":\"KC IT DEPT\"}" https://localhost:5001/teams/e52baa63-d511-417e-9e54-7aab04286281

output:



```
D:\>curl --insecure -H "Content-Type:application/json" -X PUT -d "{\"id\":\"e52baa63-d511-417e-9e54-7aab0
4286281\", \"name\":\"KC IT DEPT\"}"  https://localhost:5001/teams/e52baa63-d511-417e-9e54-7aab04286281
{"name":"KC IT DEPT","id":"e52baa63-d511-417e-9e54-7aab04286281","members":[]}
D:\>
```

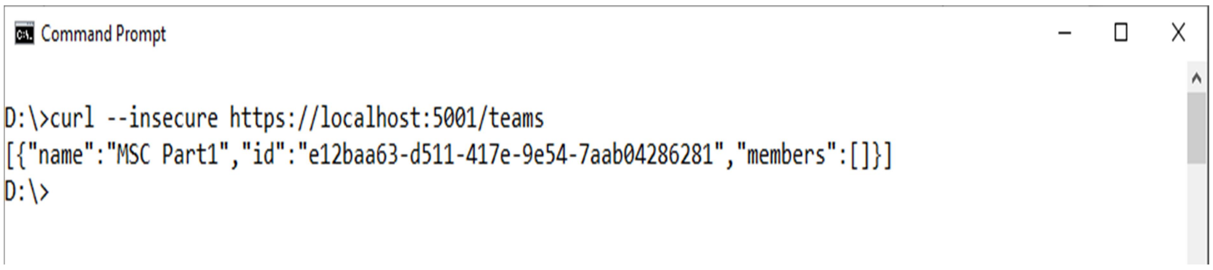
Command: to delete team
curl --insecure -H "Content-Type:application/json" –X DELETE https://localhost:5001/teams/e52baa63-d511-417e-9e54-7aab04286281
output:



Command Prompt

```
D:\>curl --insecure -H "Content-Type:application/json" -X DELETE   https://localhost:5001/teams/e52baa63-
d511-417e-9e54-7aab04286281
"e52baa63-d511-417e-9e54-7aab04286281"
D:\>
```

Confirm: with get all teams now it shows only one team (first one is deleted)
Command:
curl –insecure https://localhost:5001/teams
Output:



Command Prompt

```
D:\>curl --insecure https://localhost:5001/teams
[{"name":"MSC Part1","id":"e12baa63-d511-417e-9e54-7aab04286281","members":[]}]
D:\>
```