1

app.Map("/Currenttime", Index);

app.Run(context =>

{

return context.Response.WriteAsync("Hello ASP.NET Core MVC Class!!!.");

});

static void Index(IApplicationBuilder app)

{

app.Run(async context =>

{

context.Response.WriteAsync(DateTime.Now.ToString());

});

}

app.Map("/Currenttime", (IApplicationBuilder app) =>

{

app.Use(async (context, next) =>

{

await next();

await context.Response.WriteAsync(DateTime.Now.ToString());

});

app.Use(async (context, next) =>

{

await next();

});

app.Run((context) =>

{

return context.Response.WriteAsync(DateTime.Now.ToString()+"\n");

});

2

var x = 2;

app.Map("/X", (IApplicationBuilder app) =>

{

app.Use(async (context, next) =>

{

x = x + 2;

await next();

await context.Response.WriteAsync($"x={x}");

});

app.Use(async (context, next) =>

{

x = x + 2;

await next();

});

app.Use(async (context, next) =>

{

x = x + 2;

await next();

});

});

3

DatabaseContext.cs:

public DbSet<Department> Departments { get; set; }

public DatabaseContext(DbContextOptions<DatabaseContext> options): base(options)

{

Database.EnsureCreated();

}

Department.cs:

public class Department

{

public int Id { get; set; }

public string Name { get; set; }

public string Address { get; set;}

}

Program.cs:

string connection =

builder.Configuration.GetConnectionString("DefaultConnection");

builder.Services.AddDbContext<DatabaseContext>(options => options.UseSqlServer(connection));

app.MapGet("/get", (DatabaseContext context) => context.Departments);

app.MapPost("/post", (DatabaseContext context, Department department) =>

{

context.Departments.Add(department);

context.SaveChanges();

});

app.MapPut("/put", (DatabaseContext context, Department department) =>

{

var dept = context.Departments.FirstOrDefault(x => x.Id == department.Id);

if (dept != null)

{

dept.Name = department.Name;

dept.Address = department.Address;

context.Departments.Update(dept);

context.SaveChanges();

}

});

app.MapDelete("/{id}", (DatabaseContext context, int id) =>

{

var dept = context.Departments.FirstOrDefault(x => x.Id == id);

if (dept != null)

{

context.Departments.Remove(dept);

context.SaveChanges();

}

});

OR

DatabaseContext.cs:

public DbSet<Employee> Employees { get; set; }

public DatabaseContext(DbContextOptions<DatabaseContext> options): base(options)

{

Database.EnsureCreated();

}

Employee.cs:

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

public int Salary { get; set; }

}

Program.cs:

//incase of list or repository remove the database context and incase he mentioned database context remove List.

List<Employee> employees = new List<Employee>

{

   new() { Id = 1, Name = "Ohoud", Salary=1250},

   new() { Id = 2, Name  = "Rana", Salary=3200 },

   new() { Id = 3, Name  = "Marah", Salary=4000}

};

string connection =

builder.Configuration.GetConnectionString("DefaultConnection");

builder.Services.AddDbContext<DatabaseContext>(options => options.UseSqlServer(connection));

app.MapGet("/get", (DatabaseContext context) => context.Employees);

app.MapPost(("/post"), (DatabaseContext context, Employee employee) =>

{

context.Employees.Add(employee);

context.SaveChanges();

});

app.MapPut(("/put"), (DatabaseContext context, Employee employee) =>

{

var emp = context.Employees.FirstOrDefault(x => x.Id == employee.Id);

if (emp != null)

{

emp.Salary = employee.Salary;

emp.Name = employee.Name;

context.Employees.Update(emp);

context.SaveChanges();

}

});

app.MapDelete(("/{id}"), (DatabaseContext context, int id) =>

{

var emp = context.Employees.FirstOrDefault(x => x.Id == id);

if (emp != null)

{

context.Employees.Remove(emp);

context.SaveChanges();

}

});