

Consul is used for Service Discovery and a load balancing

[Downloads | Consul by HashiCorp](https://www.consul.io/downloads)

Copy the path  
Consul.exe agent –dev

Its running on port 8500

<http://localhost:8500/ui/dc1/services>

Steps

Create your service

Install Consul package

Add a class

namespace Microsoft.Extensions.DependencyInjection

{

public static class ServicesConfiguration

{

public static IServiceCollection AddConsulConfig(this IServiceCollection services, IConfiguration configuration)

{

services.AddSingleton<IConsulClient, ConsulClient>

(p => new ConsulClient(consulConfig =>

{

consulConfig.Address = new Uri("http://localhost:8500");

}));

return services;

}

public static IApplicationBuilder UseConsul(this IApplicationBuilder app,

IConfiguration configurationSetting)

{

var consulClient = app.ApplicationServices.GetRequiredService<IConsulClient>();

var logger = app.ApplicationServices.GetRequiredService<ILoggerFactory>().CreateLogger("AppExtensions");

var lifetime = app.ApplicationServices.GetRequiredService<IApplicationLifetime>();

var registration = new AgentServiceRegistration()

{

//ID = configurationSetting.GetValue<string>("ServiceName:Configuration"),

//Name = configurationSetting.GetValue<string>("ServiceName:Configuration"),

//Address = configurationSetting.GetValue<string>("ServiceHost:Configuration"),

//Port = configurationSetting.GetValue<int>("ServicePort:Configuraton")

ID= "students",

Name= "students",

Address="localhost",

Port = 58491

};

logger.LogInformation("Registering with Consul");

consulClient.Agent.ServiceDeregister(registration.ID).ConfigureAwait(true);

consulClient.Agent.ServiceRegister(registration).ConfigureAwait(true);

lifetime.ApplicationStopping.Register(() =>

{

logger.LogInformation("UnRegistreing from Consul");

});

return app;

}

In Startup.cs file

public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.

public void ConfigureServices(IServiceCollection services){

services.AddControllers();

services.AddDbContext<StudentDbContext>(op => op.UseSqlServer(Configuration["ConnectionStrings:StudentDbContext"]));

**services.AddConsulConfig(Configuration);**

services.AddSwaggerGen();

}

// This method gets called by the runtime. Use this method to configure the HTTP request pipeline.

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

**app.UseConsul(Configuration);**

app.UseSwagger();

// Enable middleware to serve swagger-ui (HTML, JS, CSS, etc.),

// specifying the Swagger JSON endpoint.

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "My API V1");

});

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

//loggerFactory.AddConsole();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

});

}

