Now that we have all the packages for Serilog installed, let's inject it inside our application and

let's provide the configuration for Serilog inside our application as well.

For that I will open program.cs file.

var builder = WebApplication.CreateBuilder(args);

// Configure Serilog for logging

var logger = new LoggerConfiguration()

.WriteTo.Console()

.MinimumLevel.Information()

.CreateLogger();

// clear default logging providers and add Serilog

builder.Logging.ClearProviders();

// Add Serilog as the logging provider

builder.Logging.AddSerilog(logger);

Program.cs file

Inject into controller for logging to console

[Route("api/[controller]")]

[ApiController]

public class TestLoggingController : ControllerBase

{

private readonly IRegionRepository \_regionRepository;

private readonly ILogger<TestLoggingController> \_Logger;

public TestLoggingController(IRegionRepository regionRepository, ILogger<TestLoggingController> logger1)

{

\_regionRepository = regionRepository;

\_Logger = logger1;

}

[HttpGet]

[Route("Serilog")]

public async Task<IActionResult> SerilogLogging()

{

\_Logger.LogInformation($"serilog logging is working fine");

//calling repository to get all regions

var regionsDomain = await \_regionRepository.GetAllAsync();

\_Logger.LogInformation($"Finished getting all regions request total records : {regionsDomain.Count()} and records: {System.Text.Json.JsonSerializer.Serialize(regionsDomain)}");

return Ok(regionsDomain);

}

}

**📊 Log Levels (low to high):**

| **Level** | **Use Case** |
| --- | --- |
| Verbose | Sab kuch log karo (full detail) |
| Debug | Developer debugging info |
| Information | Normal operations (e.g., API hit) |
| Warning | Something odd but not an error |
| Error | Exceptions or failed operations |
| Fatal | Critical failure (app crash) |

**🔍 Serilog Logging Levels (low to high)**

| **Level** | **Use Case Example** | **Captured when...** |
| --- | --- | --- |
| Verbose | Har chhoti detail (for developers only) | ❌ Ignore hota hai agar minimum level Warning hai |
| Debug | Development time messages | ❌ Ignore hota hai |
| Information | Normal flow info (e.g. user logged in) | ❌ Ignore hota hai |
| Warning | Kuch galat ho sakta hai (e.g. low disk) | ✅ Captured |
| Error | Error hua, lekin app chal rahi hai | ✅ Captured |
| Fatal | Bahut bada error, app crash ho sakti hai | ✅ Captured |

**🔧 Example:**



**🧪 Output Only:**

****

