'AddDbContext' was called with configuration, but the context type 'LocalDbContext' only declares a parameterless constructor. This means that the configuration passed to 'AddDbContext' will never be used. If configuration is passed to 'AddDbContext', then 'LocalDbContext' should declare a constructor that accepts a DbContextOptions<LocalDbContext> and must pass it to the base constructor for DbContext.'

Soln: Need to add constructor in LocalDbContext Class.

public LocalDbContext(DbContextOptions<LocalDbContext> options) : base(options) { }

**:** 'Services for database providers 'Microsoft.EntityFrameworkCore.SqlServer', 'Microsoft.EntityFrameworkCore.Sqlite' have been registered in the service provider. Only a single database provider can be registered in a service provider. If possible, ensure that Entity Framework is managing its service provider by removing the call to 'UseInternalServiceProvider'. Otherwise, consider conditionally registering the database provider, or maintaining one service provider per database provider.'

Soln: LocalDbContext and ApplicationDbContext both used same single service provider.

Option1 : We can define different service provider for each one.

Option2 : Use the same service provider based on the network connection whether application is connected online or offline.

builder.Services.AddDbContext<LocalDbContext>(options =>

{

var isOfflineMode = Environment.GetEnvironmentVariable("USE\_SQLITE") == "true";

if (isOfflineMode)

options.UseSqlite(builder.Configuration.GetConnectionString("LocalDbConnection"));

else

options.UseSqlServer(builder.Configuration.GetConnectionString("RemoteDbConnection"));

});