**Step 1. Setup PostgreSQL database**

Install from the [PostgreSQL Official Website](https://www.postgresql.org/download/)

**Step 2. Create ASP.NET Core Web API project**

Creation of the ASP.NET Core Web API project and configure it to use our database.

NuGet Packages to be installed to use PostgreSQL:

Npgsql.EntityFrameworkCore.PostgreSQL

Do basic code setup in Startup.cs file and Hit Ctrl+F5 to run application:

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

namespace AspNetHangfireUsingPostgres

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

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();

var connString = Configuration.GetConnectionString("defaultConnection");

services.AddEntityFrameworkNpgsql().AddDbContext<AppDbContext>(options => {

options.UseNpgsql(connString);

});

}

// 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.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

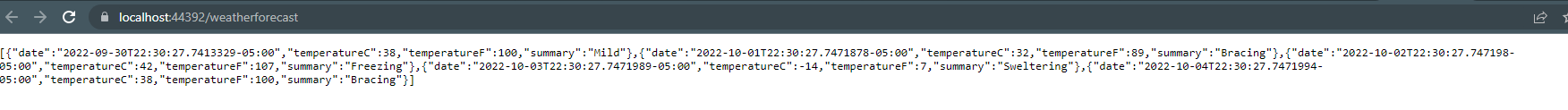
});

}

}

}

Application launched in browser at <https://localhost:44392/weatherforecast>



**Step 3. Install Hangfire packages to use PostgreSQL**

Hangfire.AspNetCore

Hangfire.Postgresql

Add the following using statements to the Startup.cs.

using Hangfire;

using Hangfire.PostgreSql;

In ConfigureServices method, let Hangfire server to use our default connection string:

services.AddHangfire(x =>

x.UsePostgreSqlStorage(Configuration.GetConnectionString("defaultConnection")));

In Configure method enter:

app.UseHangfireDashboard();

**Create few Hangfire jobs:**

In the Configure method, below the app.UseHangfireDashboard(); add the following jobs:

BackgroundJob.Enqueue(() => Console.WriteLine("Fire-and-forget"));

RecurringJob.AddOrUpdate(() => Console.WriteLine("Minutely Job"), Cron.Minutely);

Actual code:

using Hangfire;

using Hangfire.PostgreSql;

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.EntityFrameworkCore;

using Microsoft.Extensions.Configuration;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using System;

namespace AspNetHangfireUsingPostgres

{

public class Startup

{

public Startup(IConfiguration configuration)

{

Configuration = configuration;

}

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();

var connString = Configuration.GetConnectionString("defaultConnection");

services.AddEntityFrameworkNpgsql().AddDbContext<AppDbContext>(options => {

options.UseNpgsql(connString);

});

services.AddHangfire(x => x.UsePostgreSqlStorage(connString));

services.AddHangfireServer(options => options.WorkerCount = 20);

}

// 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.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllers();

});

app.UseHangfireDashboard();

BackgroundJob.Enqueue(() => Console.WriteLine("Fire-and-forget"));

RecurringJob.AddOrUpdate(() => Console.WriteLine("Minutely Job"), Cron.Minutely);

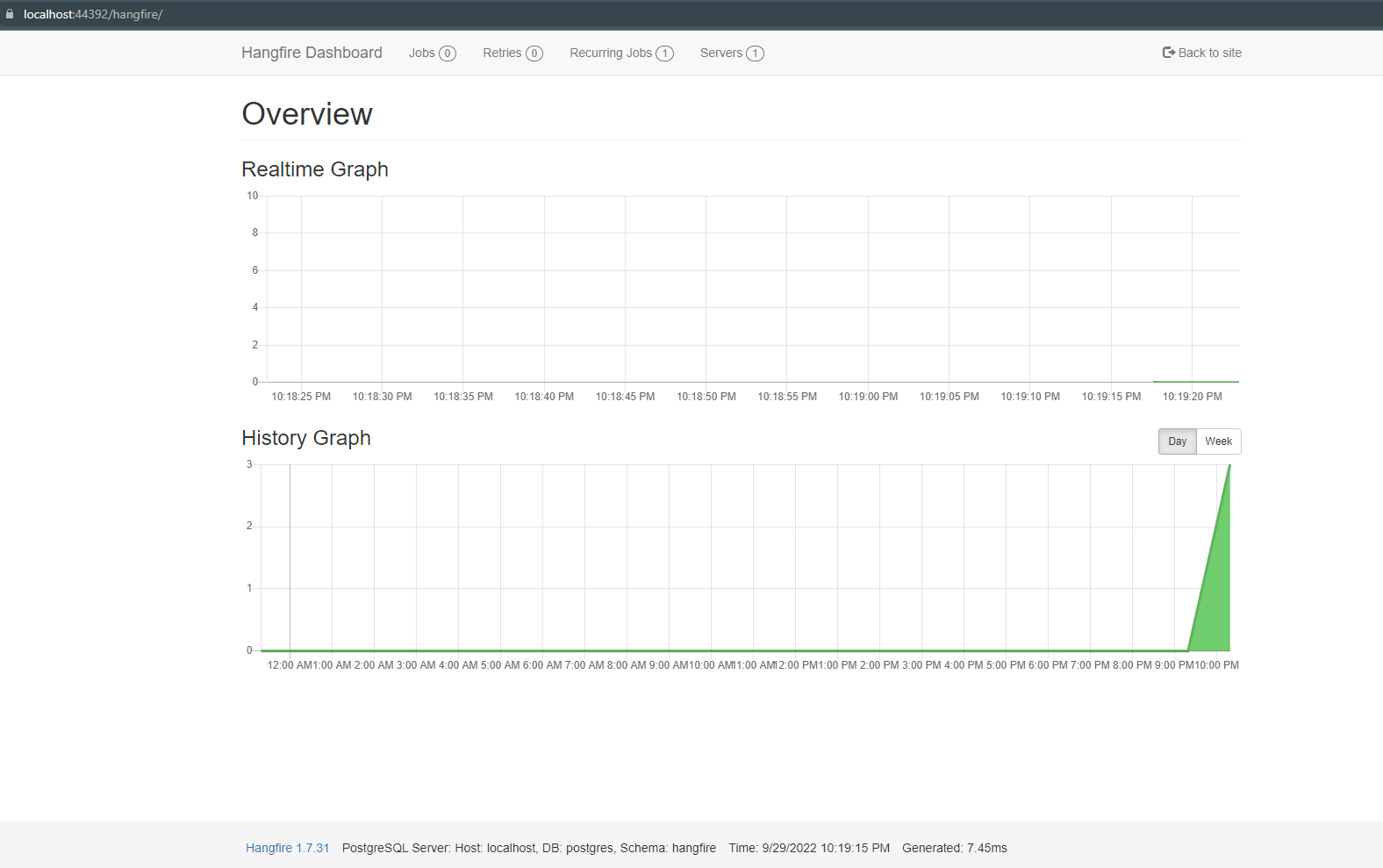
}

}

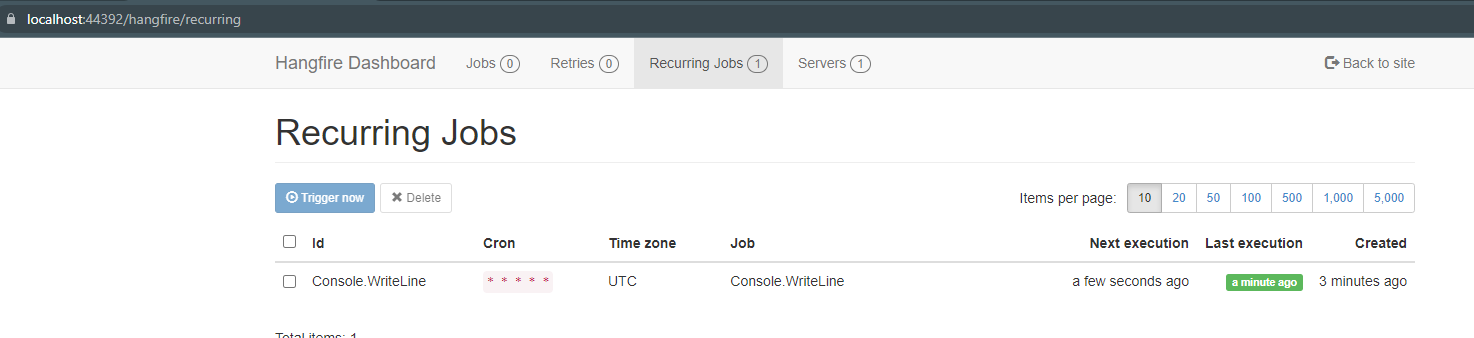
}

And finally run the app using CTRL+F5.

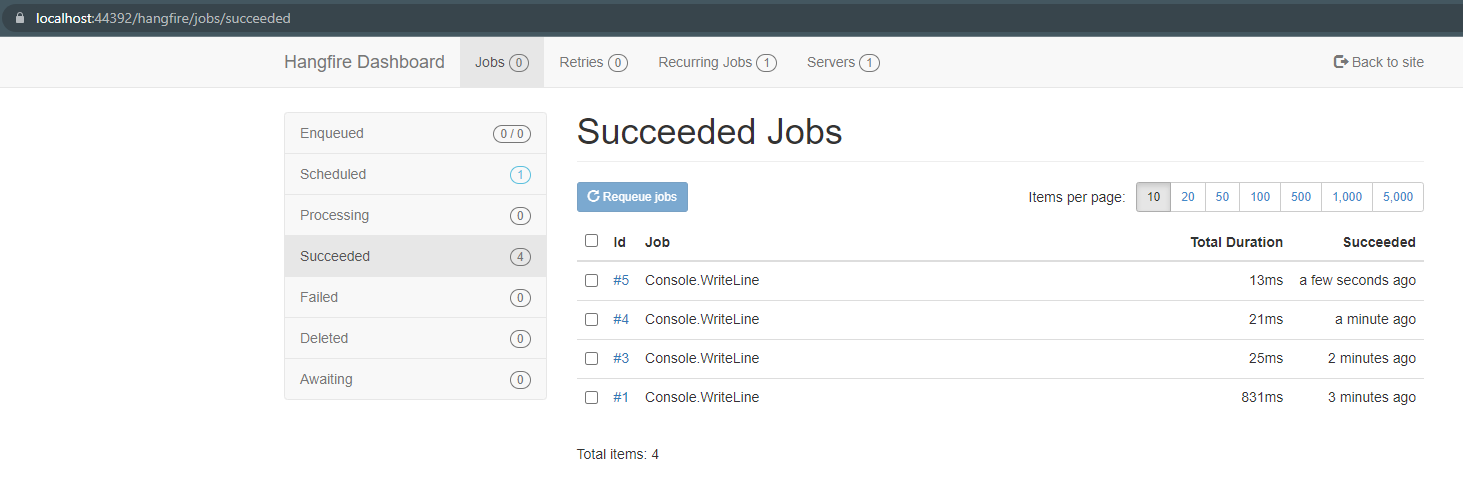
Open <https://localhost:44392/hangfire> in the browser and it would show Hangfire dashboard with scheduled recurring job.



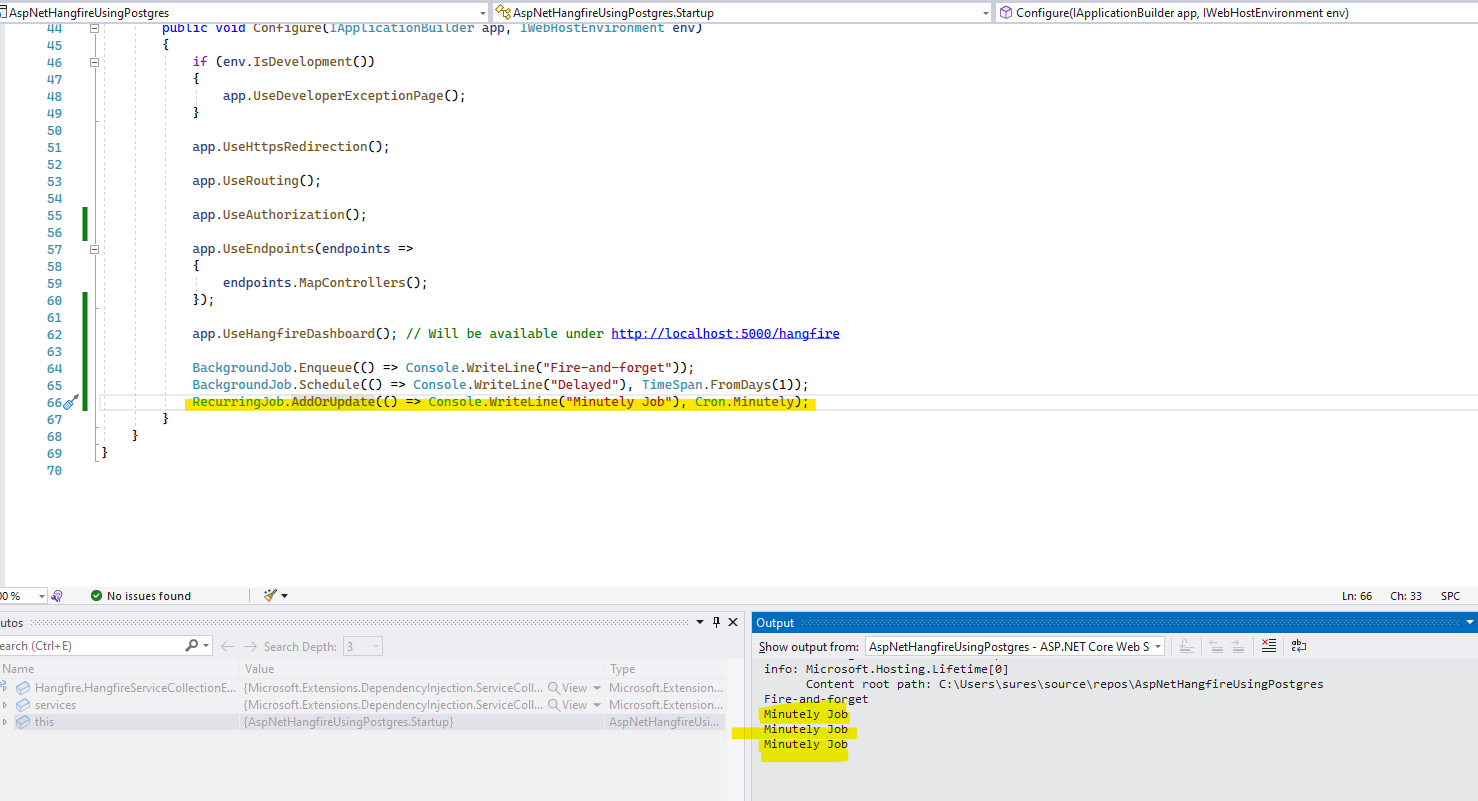
Recurring job created



Recurring job successful runs



Recurring job successful run logs



Above code automatically creates database tables under Hangfire schema:

