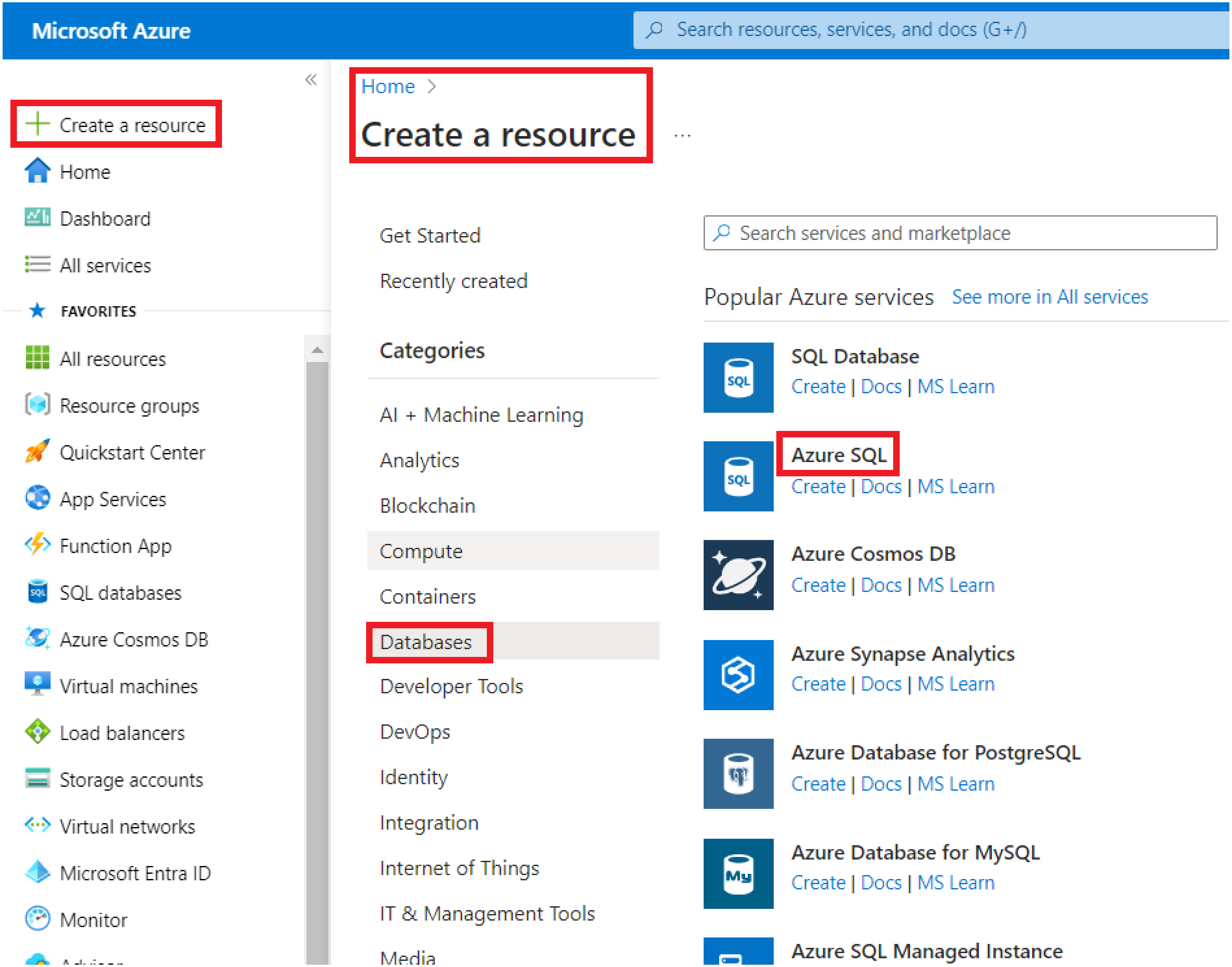
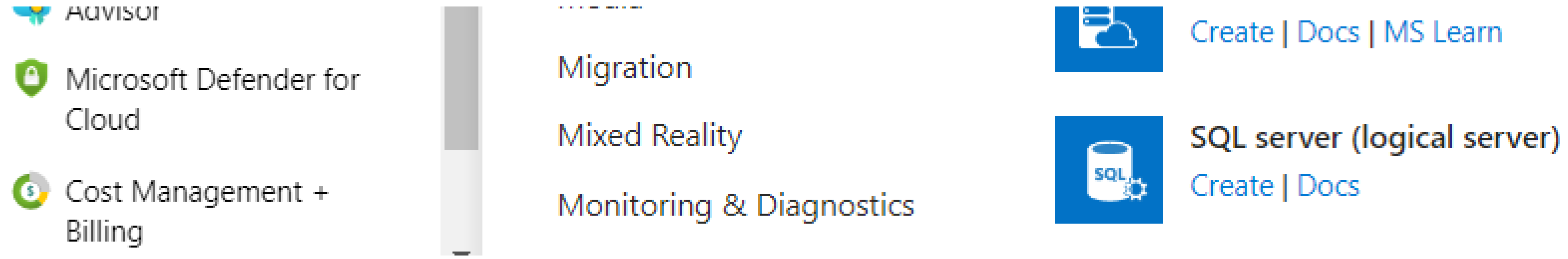
# How to create .NET8 WebAPI Azure SQL MicroService

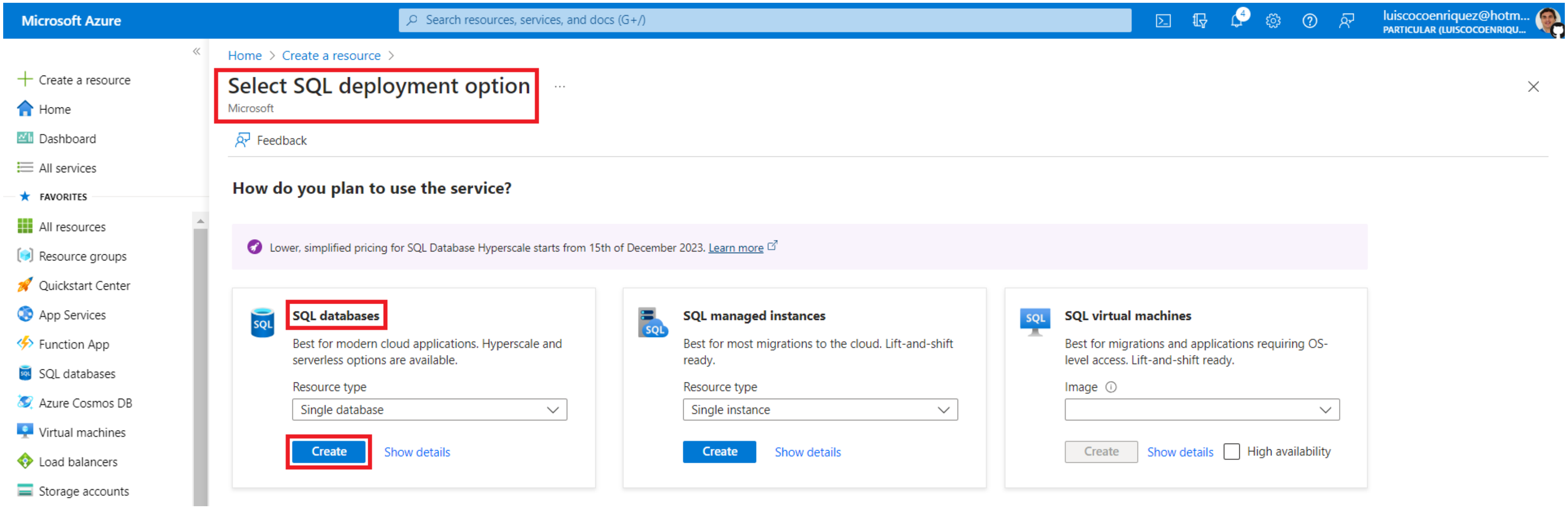
## 1. Create Azure SQL

Create Azure SQL service:


Microsoft Azure
Search resources, services, and docs (G+/)
Create a resource
Home
Dashboard
All services
FAVORITES
All resources
Resource groups
Quickstart Center
App Services
Function App
SQL databases
Azure Cosmos DB
Virtual machines
Load balancers
Storage accounts
Virtual networks
Microsoft Entra ID
Monitor
Home >
Create a resource
...
Get Started
Recently created
Categories
AI + Machine Learning
Analytics
Blockchain
Compute
Containers
Databases
Developer Tools
DevOps
Identity
Integration
Internet of Things
IT & Management Tools
Media
Search services and marketplace
Popular Azure services
See more in All services
SQL Database
Create | Docs | MS Learn
Azure SQL
Create | Docs | MS Learn
Azure Cosmos DB
Create | Docs | MS Learn
Azure Synapse Analytics
Create | Docs | MS Learn
Azure Database for PostgreSQL
Create | Docs | MS Learn
Azure Database for MySQL
Create | Docs | MS Learn
Azure SQL Managed Instance



Select SQL databases and press **Create** button



Select SQL deployment option
Microsoft

How do you plan to use the service?

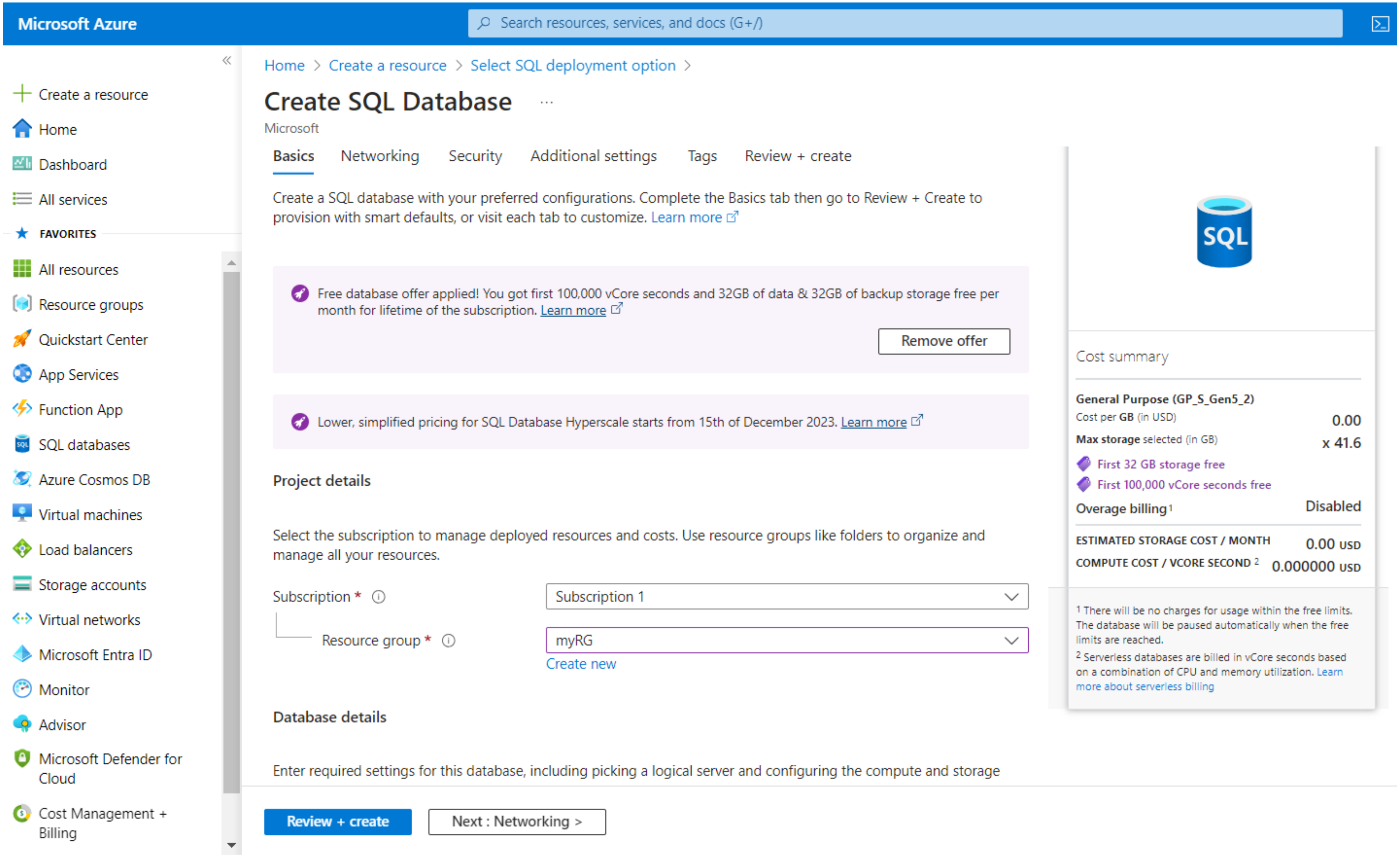
Lower, simplified pricing for SQL Database Hyperscale starts from 15th of December 2023. Learn more

**SQL databases**
Best for modern cloud applications. Hyperscale and serverless options are available.
Resource type: Single database

**SQL managed instances**
Best for most migrations to the cloud. Lift-and-shift ready.
Resource type: Single instance

**SQL virtual machines**
Best for migrations and applications requiring OS-level access. Lift-and-shift ready.
Image:

We set the database name and we create a SQL Server



## Create SQL Database

Microsoft

Create a SQL database with your preferred configurations. Complete the Basics tab then go to Review + Create to provision with smart defaults, or visit each tab to customize. Learn more

Free database offer applied! You got first 100,000 vCore seconds and 32GB of data & 32GB of backup storage free per month for lifetime of the subscription. Learn more

Lower, simplified pricing for SQL Database Hyperscale starts from 15th of December 2023. Learn more

### Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * : Subscription 1

Resource group * : myRG

### Database details

Enter required settings for this database, including picking a logical server and configuring the compute and storage

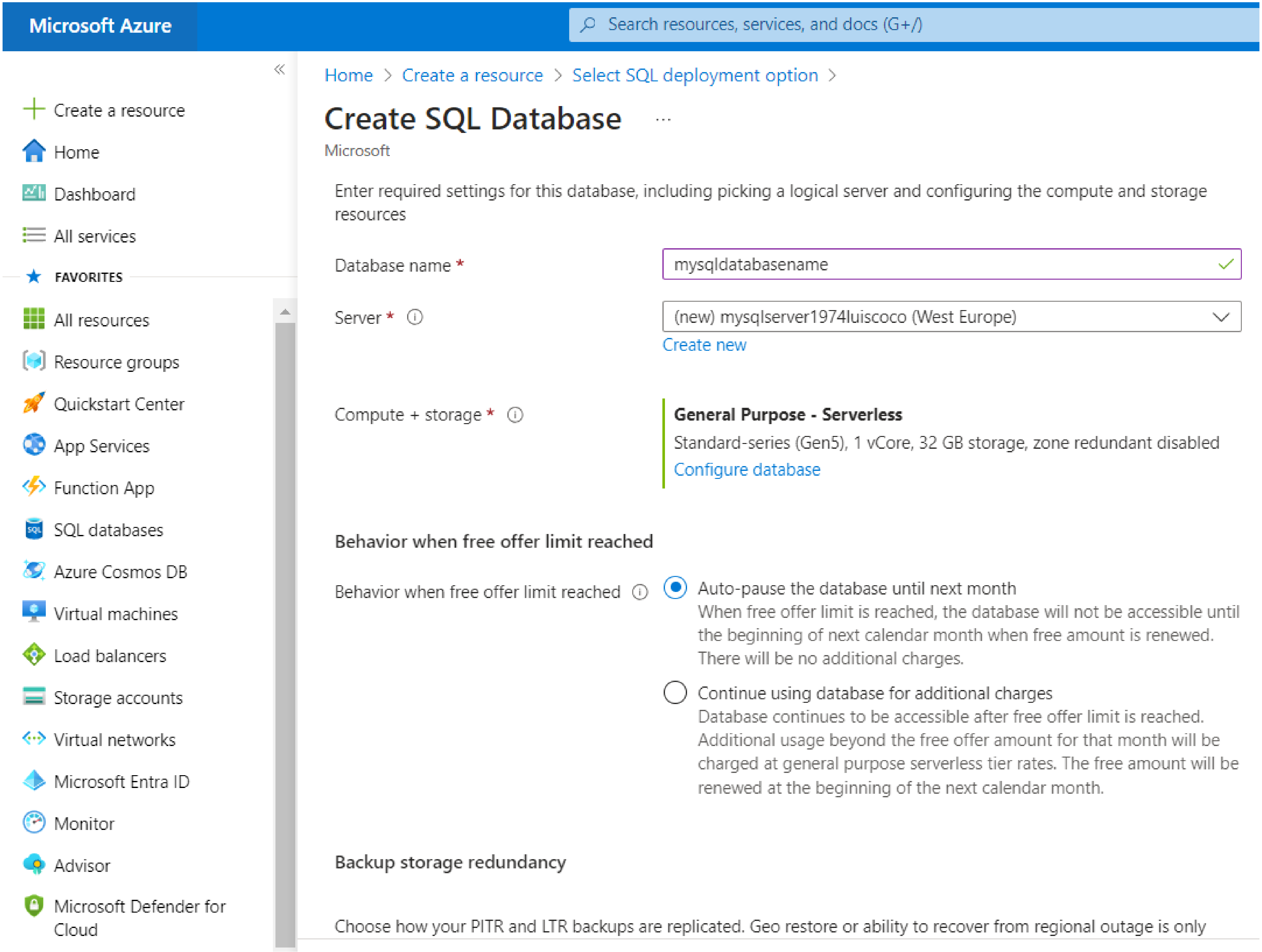
Cost summary

**General Purpose (GP_S_Gen5_2)**

| | |
|---|---|
| Cost per **GB** (in USD) | 0.00 |
| **Max storage** selected (in GB) | x 41.6 |
| First 32 GB storage free | |
| First 100,000 vCore seconds free | |
| Overage billing[1] | Disabled |
| ESTIMATED STORAGE COST / MONTH | 0.00 USD |
| COMPUTE COST / VCORE SECOND [2] | 0.000000 USD |

[1] There will be no charges for usage within the free limits. The database will be paused automatically when the free limits are reached.

[2] Serverless databases are billed in vCore seconds based on a combination of CPU and memory utilization. Learn more about serverless billing



Create SQL Database ...

Microsoft

Enter required settings for this database, including picking a logical server and configuring the compute and storage resources

Database name * mysqldatabasename ✓

Server * ⓘ (new) mysqlserver1974luiscoco (West Europe)

Create new

Compute + storage * ⓘ **General Purpose - Serverless**
Standard-series (Gen5), 1 vCore, 32 GB storage, zone redundant disabled
Configure database

Behavior when free offer limit reached

Behavior when free offer limit reached ⓘ

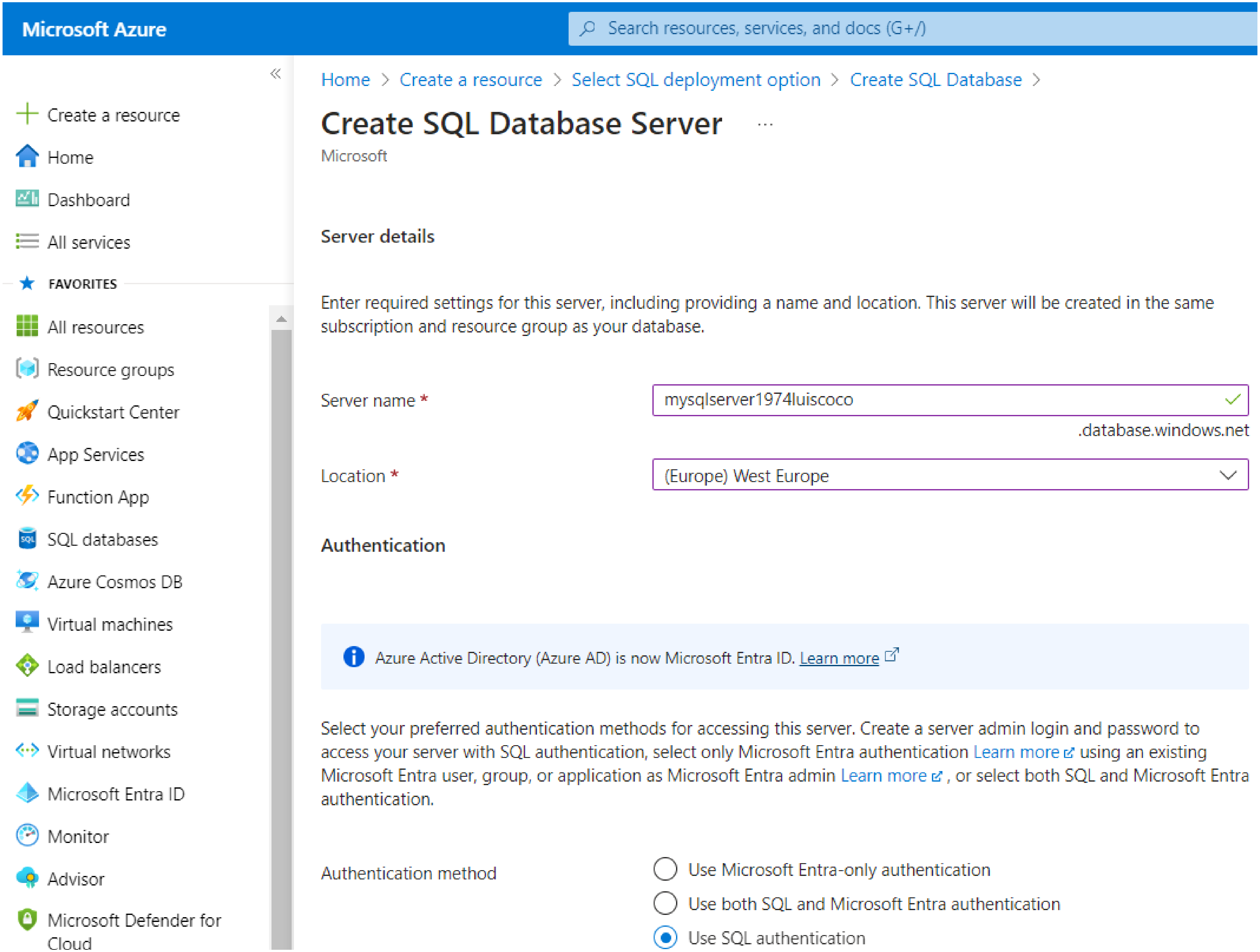
● Auto-pause the database until next month
When free offer limit is reached, the database will not be accessible until the beginning of next calendar month when free amount is renewed. There will be no additional charges.

○ Continue using database for additional charges
Database continues to be accessible after free offer limit is reached. Additional usage beyond the free offer amount for that month will be charged at general purpose serverless tier rates. The free amount will be renewed at the beginning of the next calendar month.

Backup storage redundancy

Choose how your PITR and LTR backups are replicated. Geo restore or ability to recover from regional outage is only

We create SQL Server



Home > Create a resource > Select SQL deployment option > Create SQL Database >

# Create SQL Database Server ...

Microsoft

## Server details

Enter required settings for this server, including providing a name and location. This server will be created in the same subscription and resource group as your database.

Server name * mysqlserver1974luiscoco ✓
.database.windows.net

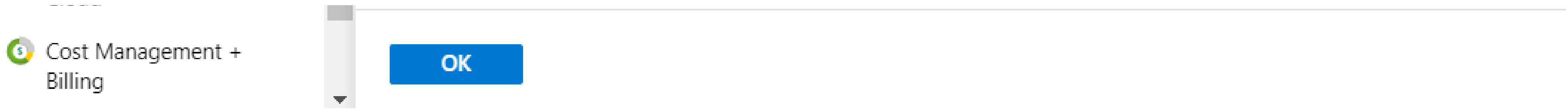
Location * (Europe) West Europe

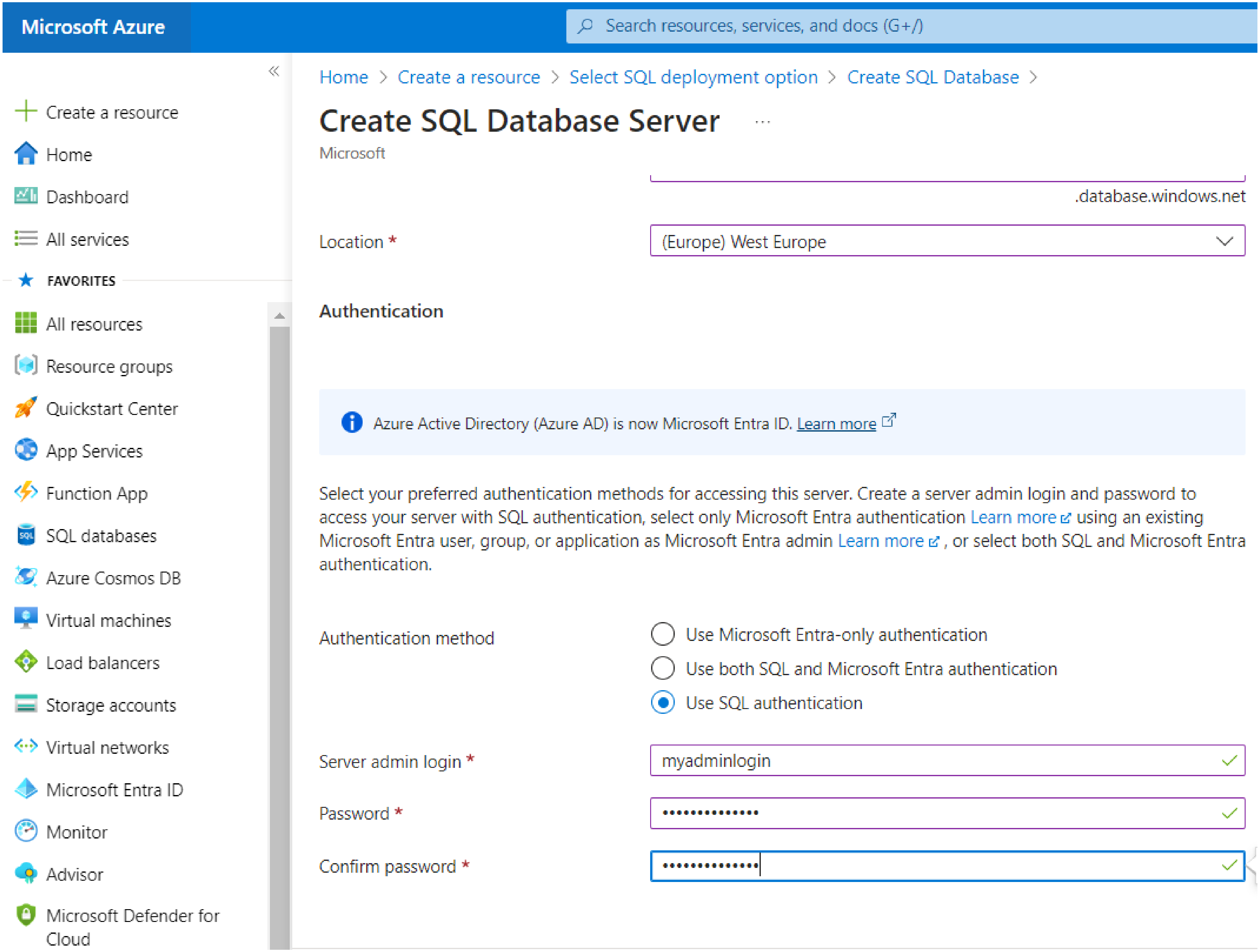
## Authentication

Azure Active Directory (Azure AD) is now Microsoft Entra ID. Learn more

Select your preferred authentication methods for accessing this server. Create a server admin login and password to access your server with SQL authentication, select only Microsoft Entra authentication Learn more using an existing Microsoft Entra user, group, or application as Microsoft Entra admin Learn more , or select both SQL and Microsoft Entra authentication.

Authentication method
○ Use Microsoft Entra-only authentication
○ Use both SQL and Microsoft Entra authentication
● Use SQL authentication


Cost Management + Billing
OK



Create SQL Database Server ...

Microsoft

.database.windows.net

Location * (Europe) West Europe

Authentication

Azure Active Directory (Azure AD) is now Microsoft Entra ID. Learn more

Select your preferred authentication methods for accessing this server. Create a server admin login and password to access your server with SQL authentication, select only Microsoft Entra authentication Learn more using an existing Microsoft Entra user, group, or application as Microsoft Entra admin Learn more , or select both SQL and Microsoft Entra authentication.

Authentication method

○ Use Microsoft Entra-only authentication

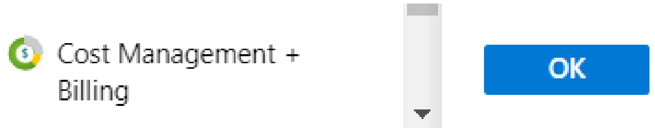
○ Use both SQL and Microsoft Entra authentication

● Use SQL authentication

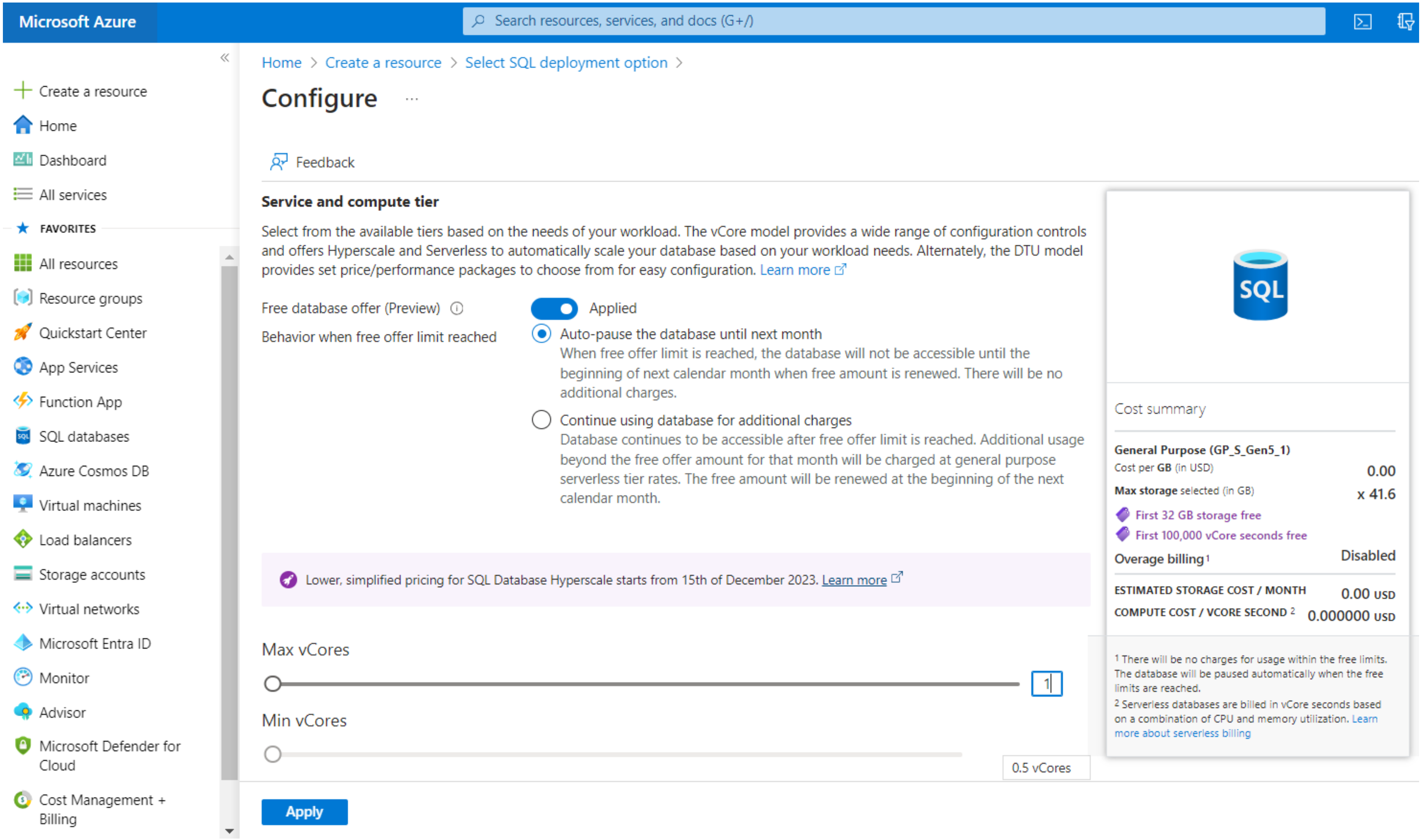
Server admin login * myadminlogin ✓

Password * ••••••••••••••• ✓

Confirm password * ••••••••••••••• ✓



We configure the database engine CPUs and Memory



Configure ...

**Service and compute tier**

Select from the available tiers based on the needs of your workload. The vCore model provides a wide range of configuration controls and offers Hyperscale and Serverless to automatically scale your database based on your workload needs. Alternately, the DTU model provides set price/performance packages to choose from for easy configuration. Learn more

Free database offer (Preview) ⓘ — Applied

Behavior when free offer limit reached

● Auto-pause the database until next month
When free offer limit is reached, the database will not be accessible until the beginning of next calendar month when free amount is renewed. There will be no additional charges.

○ Continue using database for additional charges
Database continues to be accessible after free offer limit is reached. Additional usage beyond the free offer amount for that month will be charged at general purpose serverless tier rates. The free amount will be renewed at the beginning of the next calendar month.

Lower, simplified pricing for SQL Database Hyperscale starts from 15th of December 2023. Learn more

Max vCores: 1

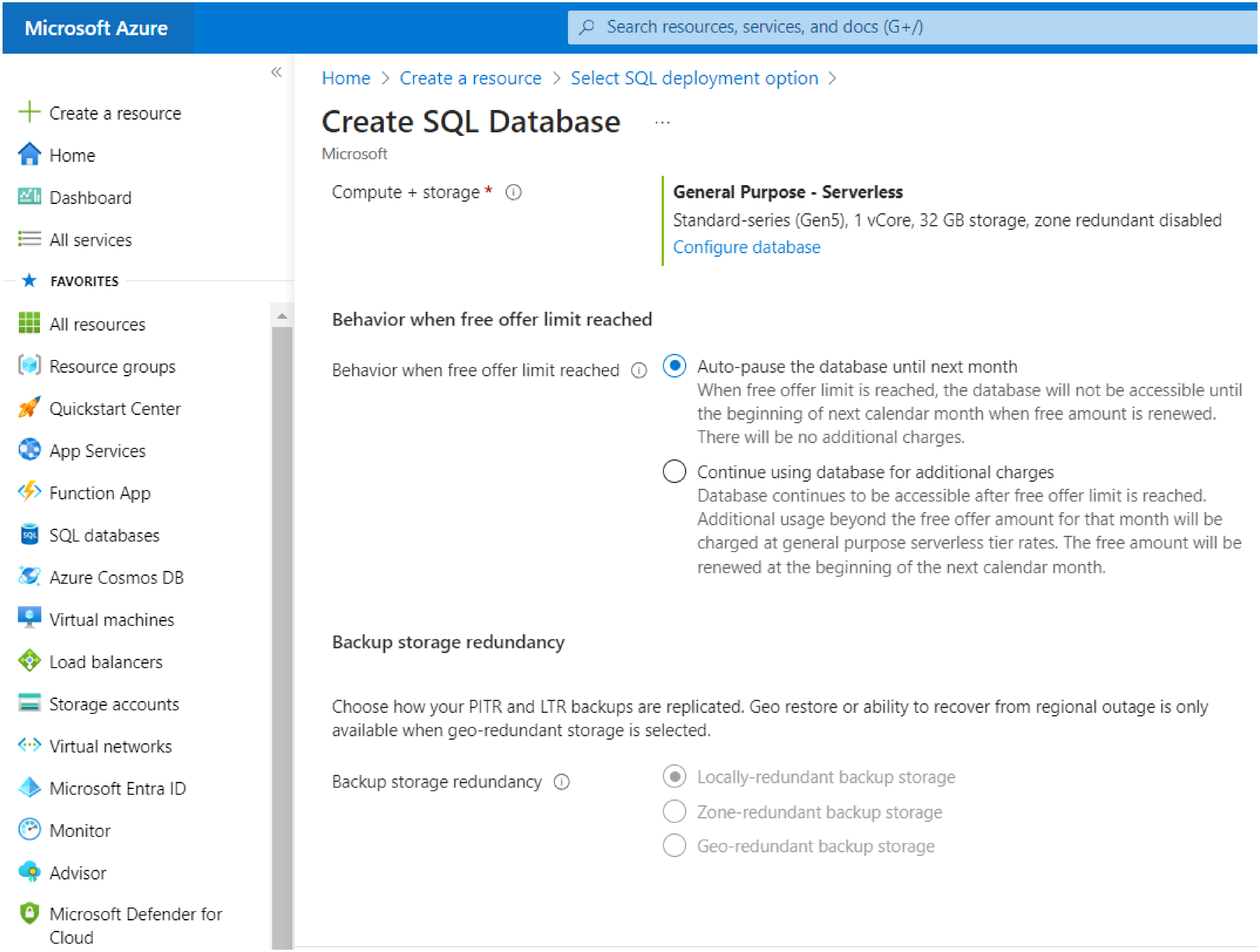
Min vCores: 0.5 vCores

Cost summary

| | |
|---|---|
| **General Purpose (GP_S_Gen5_1)** | |
| Cost per **GB** (in USD) | 0.00 |
| **Max storage** selected (in GB) | x 41.6 |
| First 32 GB storage free | |
| First 100,000 vCore seconds free | |
| Overage billing [1] | Disabled |
| ESTIMATED STORAGE COST / MONTH | 0.00 USD |
| COMPUTE COST / VCORE SECOND [2] | 0.000000 USD |

[1] There will be no charges for usage within the free limits. The database will be paused automatically when the free limits are reached.

[2] Serverless databases are billed in vCore seconds based on a combination of CPU and memory utilization. Learn more about serverless billing



Create SQL Database ...

Microsoft

Compute + storage * ⓘ

**General Purpose - Serverless**
Standard-series (Gen5), 1 vCore, 32 GB storage, zone redundant disabled
Configure database

**Behavior when free offer limit reached**

Behavior when free offer limit reached ⓘ

● Auto-pause the database until next month
When free offer limit is reached, the database will not be accessible until the beginning of next calendar month when free amount is renewed. There will be no additional charges.

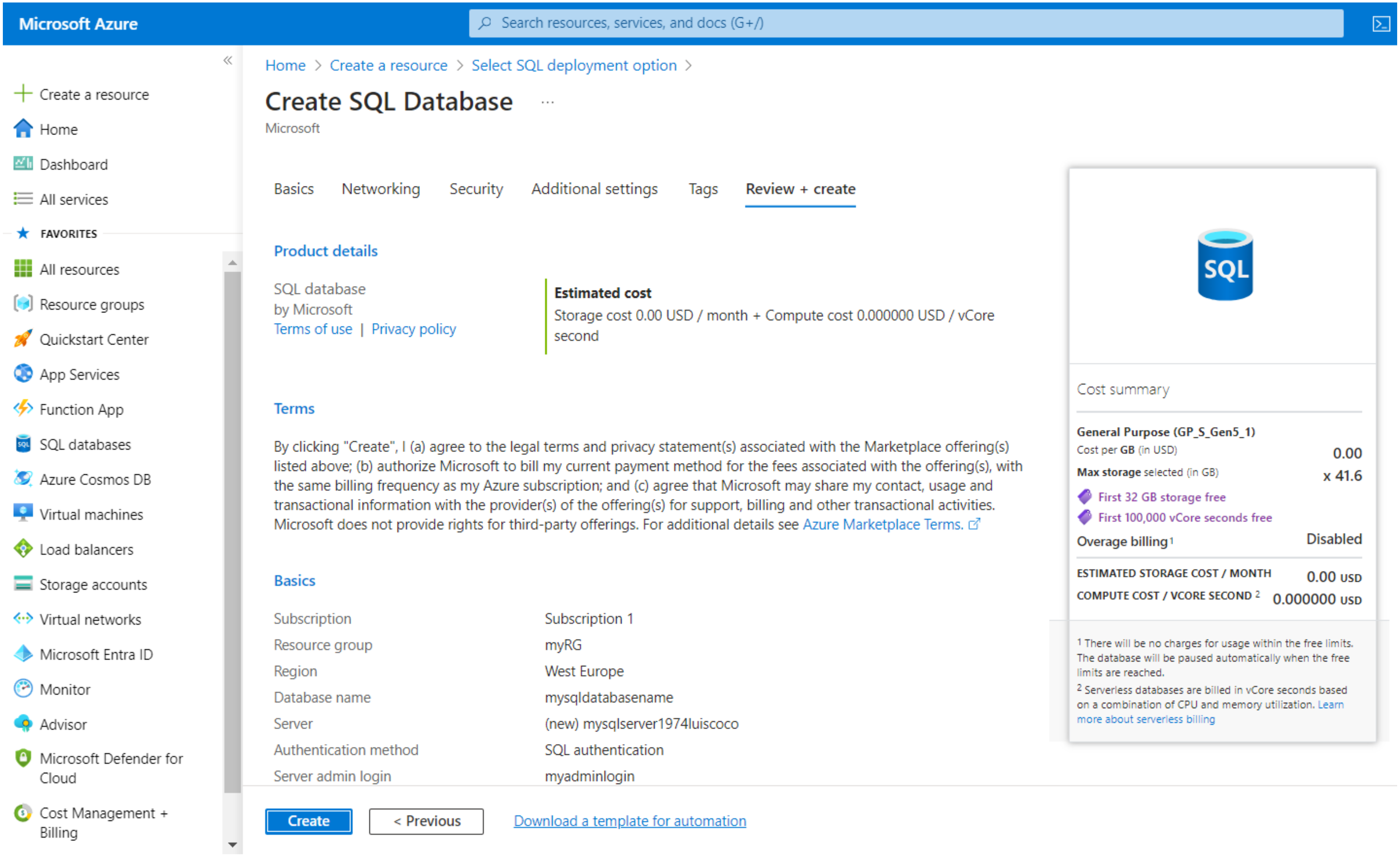
○ Continue using database for additional charges
Database continues to be accessible after free offer limit is reached. Additional usage beyond the free offer amount for that month will be charged at general purpose serverless tier rates. The free amount will be renewed at the beginning of the next calendar month.

**Backup storage redundancy**

Choose how your PITR and LTR backups are replicated. Geo restore or ability to recover from regional outage is only available when geo-redundant storage is selected.

Backup storage redundancy ⓘ

● Locally-redundant backup storage
○ Zone-redundant backup storage
○ Geo-redundant backup storage



Create SQL Database

Microsoft

Product details

SQL database
by Microsoft
Terms of use | Privacy policy

**Estimated cost**
Storage cost 0.00 USD / month + Compute cost 0.000000 USD / vCore second

Terms

By clicking "Create", I (a) agree to the legal terms and privacy statement(s) associated with the Marketplace offering(s) listed above; (b) authorize Microsoft to bill my current payment method for the fees associated with the offering(s), with the same billing frequency as my Azure subscription; and (c) agree that Microsoft may share my contact, usage and transactional information with the provider(s) of the offering(s) for support, billing and other transactional activities. Microsoft does not provide rights for third-party offerings. For additional details see Azure Marketplace Terms.

Basics

| | |
|---|---|
| Subscription | Subscription 1 |
| Resource group | myRG |
| Region | West Europe |
| Database name | mysqldatabasename |
| Server | (new) mysqlserver1974luiscoco |
| Authentication method | SQL authentication |
| Server admin login | myadminlogin |

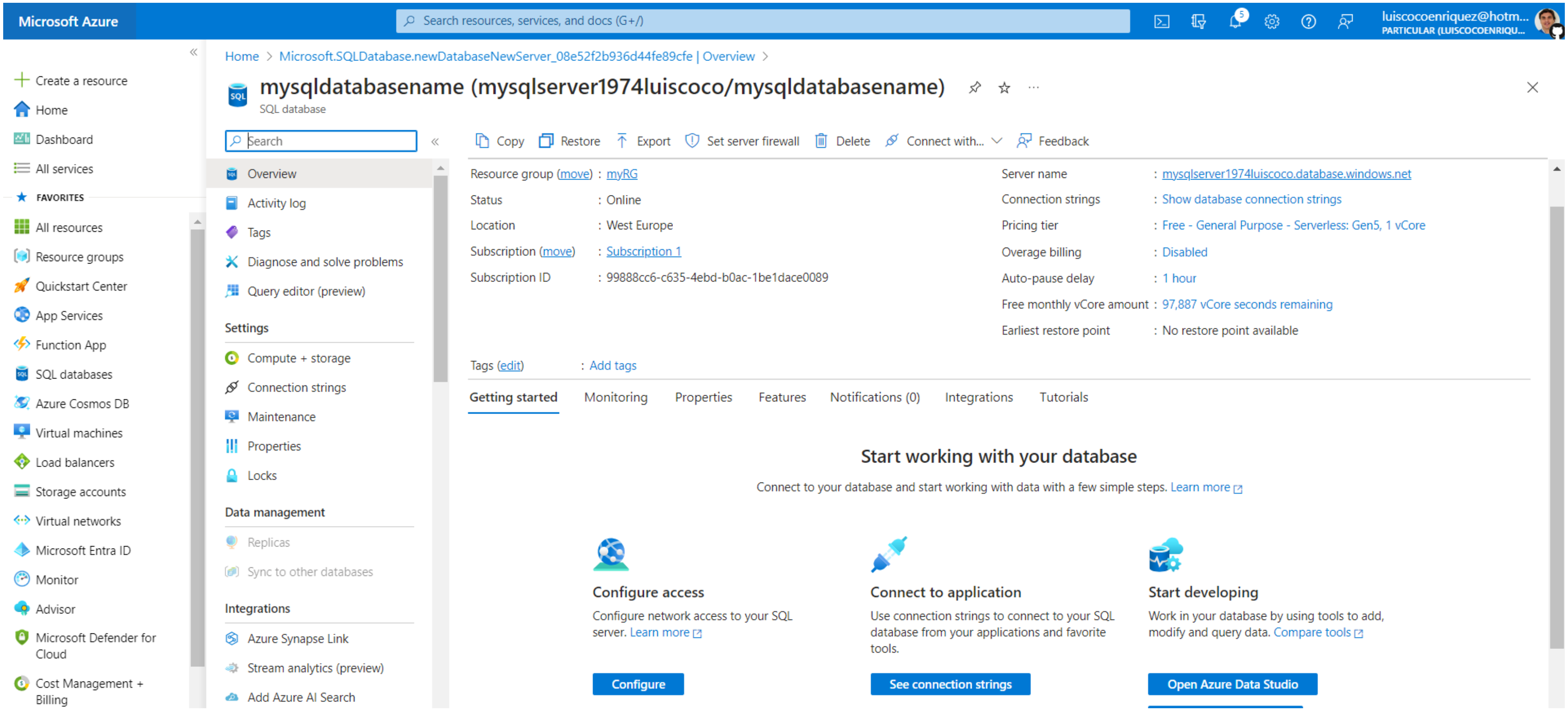
Cost summary

**General Purpose (GP_S_Gen5_1)**

| | |
|---|---|
| Cost per **GB** (in USD) | 0.00 |
| **Max storage** selected (in GB) | x 41.6 |
| First 32 GB storage free | |
| First 100,000 vCore seconds free | |
| Overage billing[1] | Disabled |
| ESTIMATED STORAGE COST / MONTH | 0.00 USD |
| COMPUTE COST / VCORE SECOND [2] | 0.000000 USD |

[1] There will be no charges for usage within the free limits. The database will be paused automatically when the free limits are reached.

[2] Serverless databases are billed in vCore seconds based on a combination of CPU and memory utilization. Learn more about serverless billing



mysqldatabasename (mysqlserver1974luiscoco/mysqldatabasename)

SQL database

Resource group (move) : myRG
Status : Online
Location : West Europe
Subscription (move) : Subscription 1
Subscription ID : 99888cc6-c635-4ebd-b0ac-1be1dace0089

Server name : mysqlserver1974luiscoco.database.windows.net
Connection strings : Show database connection strings
Pricing tier : Free - General Purpose - Serverless: Gen5, 1 vCore
Overage billing : Disabled
Auto-pause delay : 1 hour
Free monthly vCore amount : 97,887 vCore seconds remaining
Earliest restore point : No restore point available

Tags (edit) : Add tags

Start working with your database

Connect to your database and start working with data with a few simple steps. Learn more

Configure access

Configure network access to your SQL server. Learn more

Connect to application

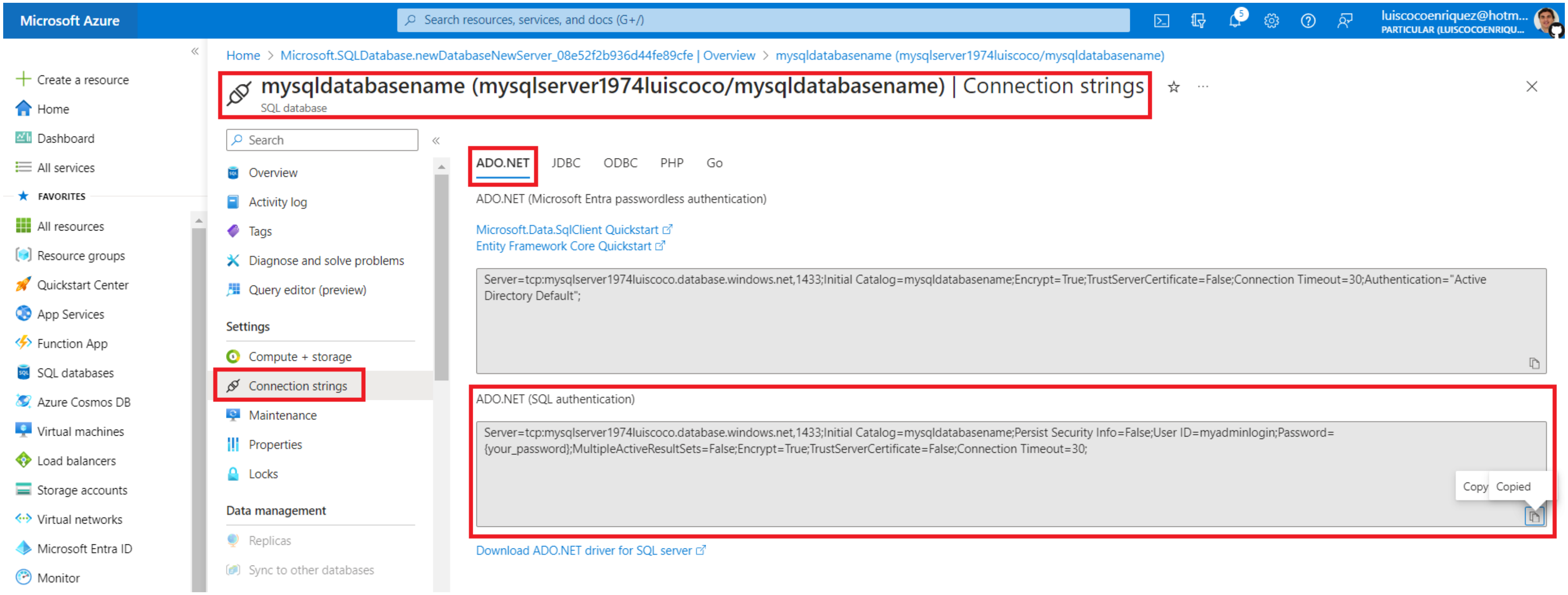
Use connection strings to connect to your SQL database from your applications and favorite tools.

Start developing

Work in your database by using tools to add, modify and query data. Compare tools

## IMPORTANT:

We copy the connection string to the appsettings.json file



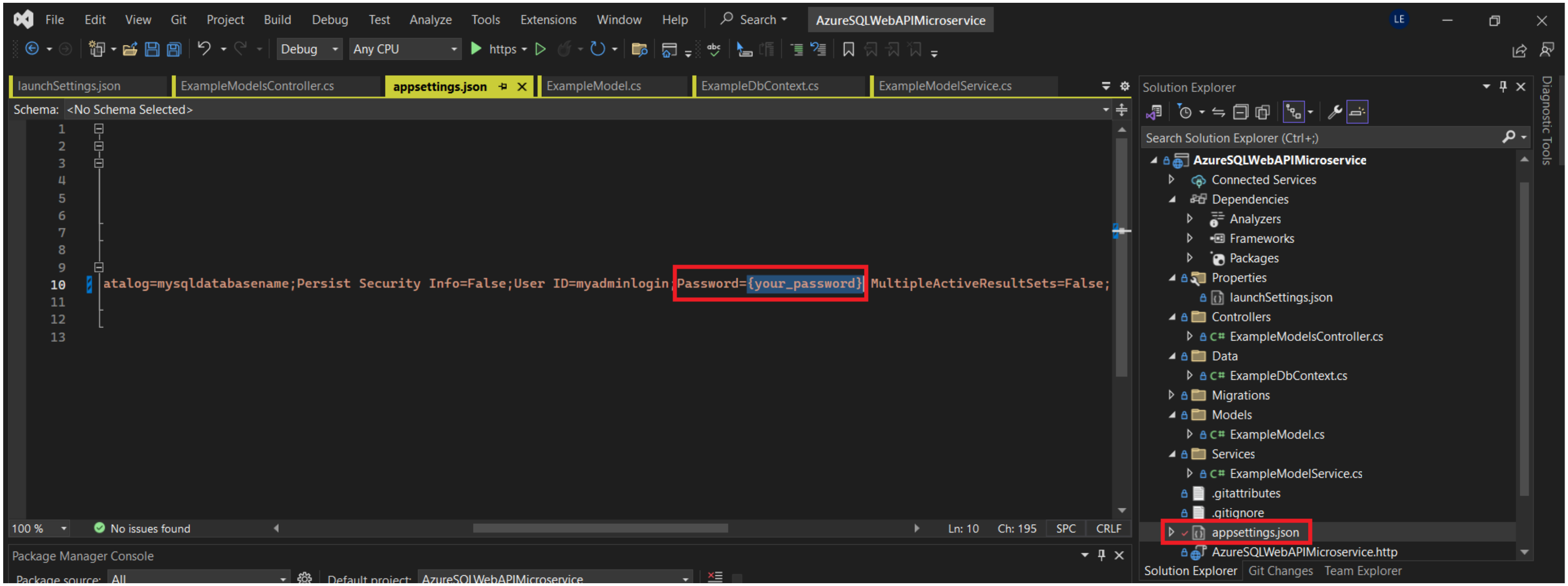
ADO.NET (Microsoft Entra passwordless authentication)

```
Server=tcp:mysqlserver1974luiscoco.database.windows.net,1433;Initial Catalog=mysqldatabasename;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;Authentication="Active Directory Default";
```

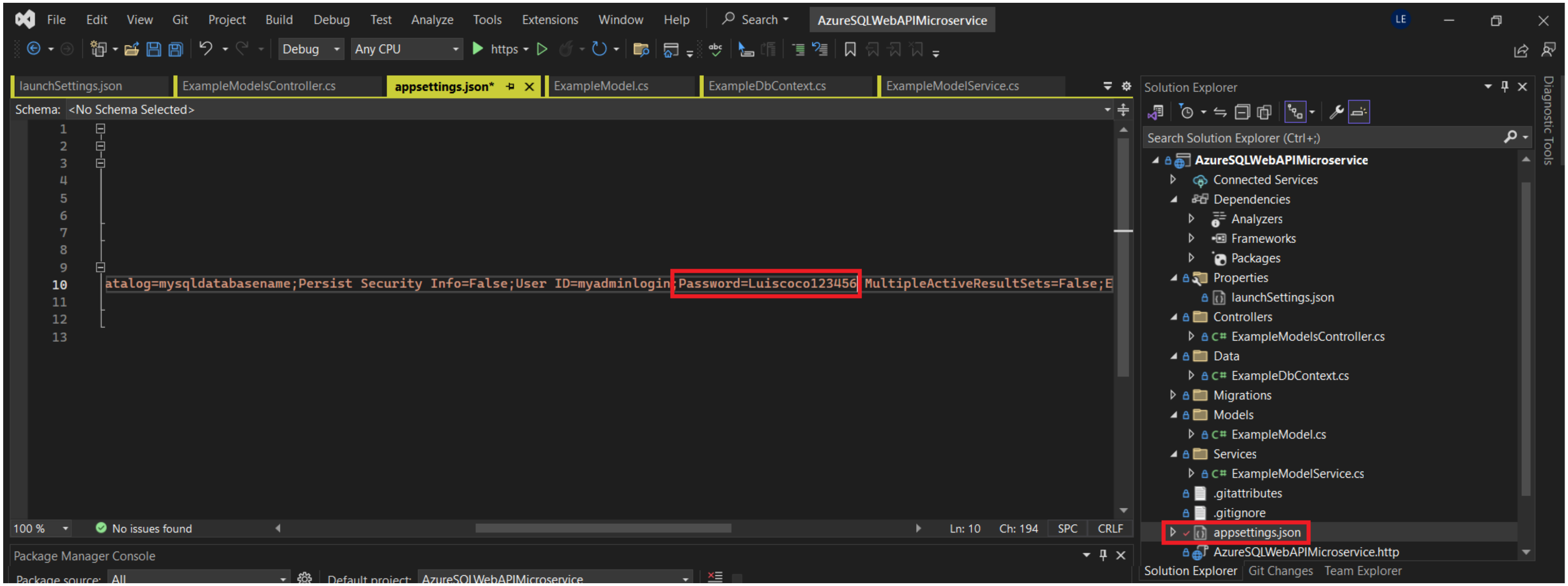
ADO.NET (SQL authentication)

```
Server=tcp:mysqlserver1974luiscoco.database.windows.net,1433;Initial Catalog=mysqldatabasename;Persist Security Info=False;User ID=myadminlogin;Password={your_password};MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;
```

Do not forget to set the Server Password in the connection string



```
atalog=mysqldatabasename;Persist Security Info=False;User ID=myadminlogin;Password={your_password};MultipleActiveResultSets=False;
```

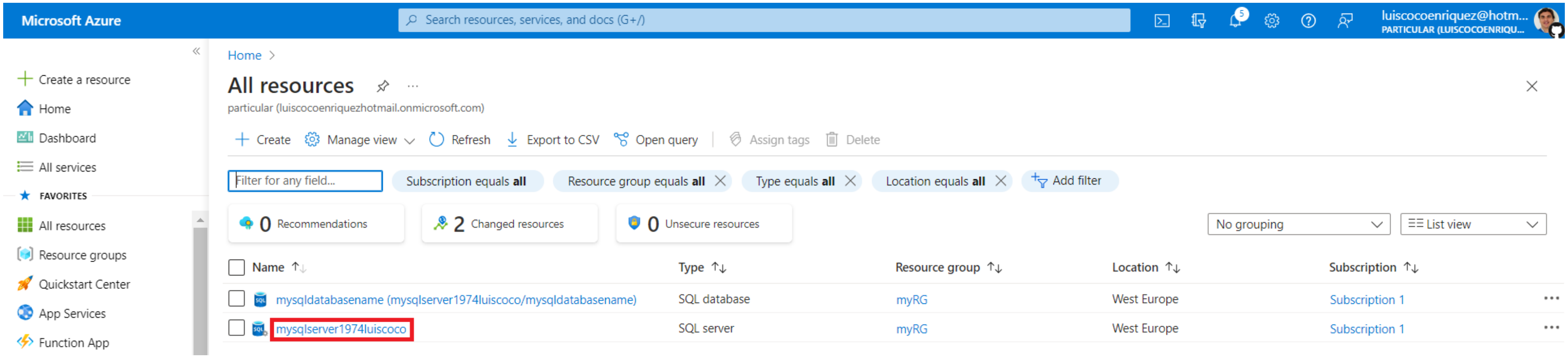


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help Search AzureSQLWebAPIMicroservice
Debug Any CPU https
launchSettings.json ExampleModelsController.cs appsettings.json* ExampleModel.cs ExampleDbContext.cs ExampleModelService.cs
Schema: <No Schema Selected>
100 % No issues found Ln: 10 Ch: 194 SPC CRLF
Package Manager Console
Package source: All Default project: AzureSQLWebAPIMicroservice
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution Explorer Git Changes Team Explorer
Diagnostic Tools

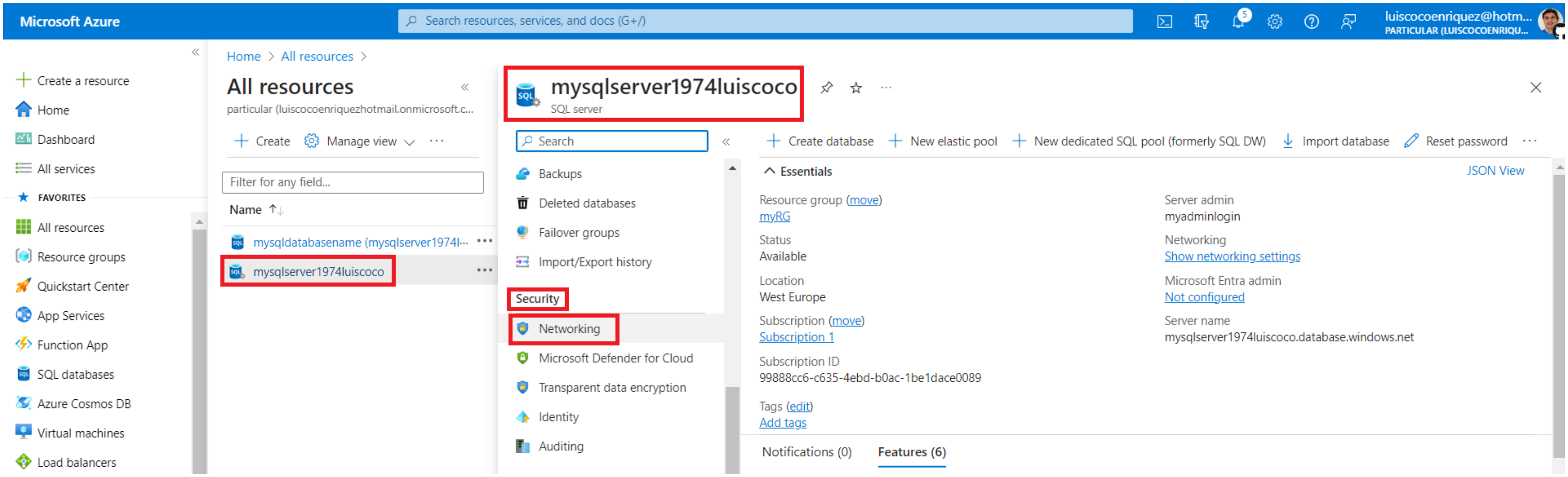
```
atalog=mysqldatabasename;Persist Security Info=False;User ID=myadminlogin;Password=Luiscoco123456;MultipleActiveResultSets=False;E
```

```
AzureSQLWebAPIMicroservice
  Connected Services
  Dependencies
    Analyzers
    Frameworks
    Packages
  Properties
    launchSettings.json
  Controllers
    ExampleModelsController.cs
  Data
    ExampleDbContext.cs
  Migrations
  Models
    ExampleModel.cs
  Services
    ExampleModelService.cs
  .gitattributes
  .gitignore
  appsettings.json
  AzureSQLWebAPIMicroservice.http
```

We also enter in the SQL Server and add the FireWall rules



| Name | Type | Resource group | Location | Subscription |
|---|---|---|---|---|
| mysqldatabasename (mysqlserver1974luiscoco/mysqldatabasename) | SQL database | myRG | West Europe | Subscription 1 |
| mysqlserver1974luiscoco | SQL server | myRG | West Europe | Subscription 1 |



## All resources

particular (luiscocoenriquezhotmail.onmicrosoft.c...

## mysqlserver1974luiscoco

Resource group (move)
myRG

Status
Available

Location
West Europe

Subscription (move)
Subscription 1

Subscription ID
99888cc6-c635-4ebd-b0ac-1be1dace0089

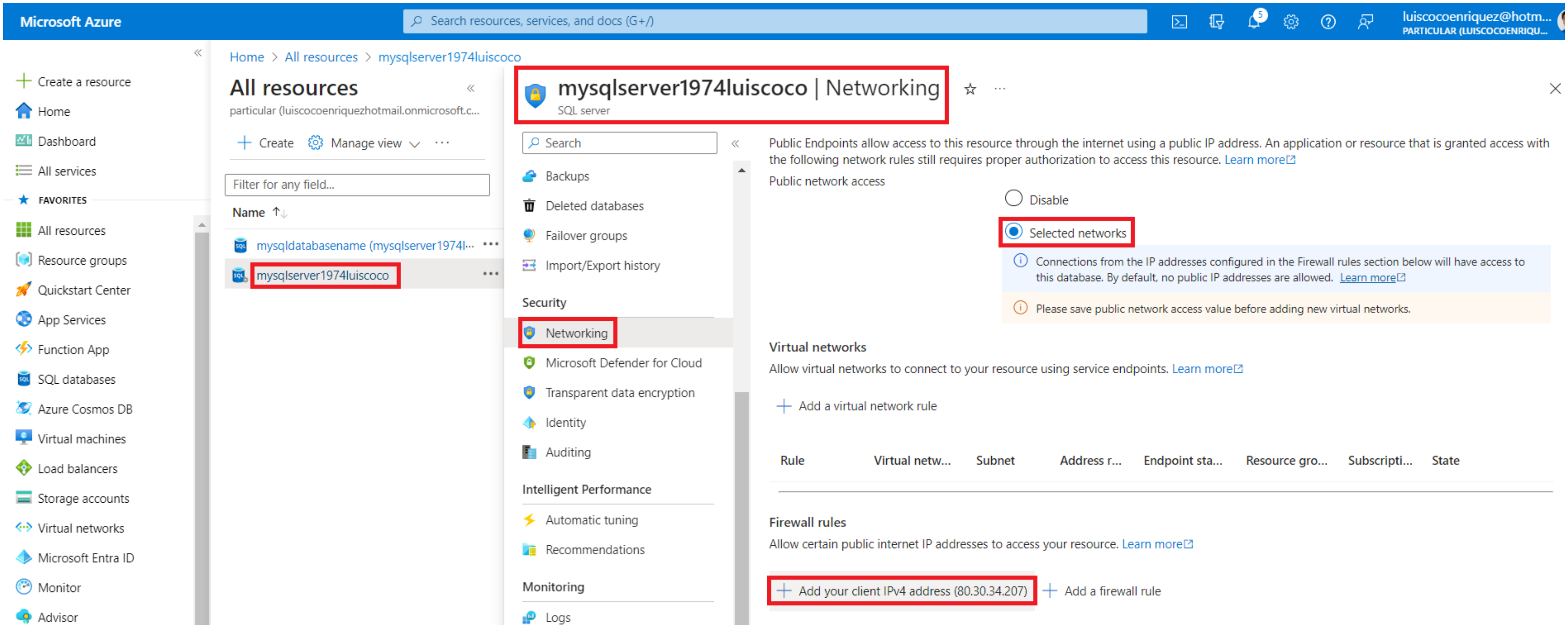
Tags (edit)
Add tags

Server admin
myadminlogin

Networking
Show networking settings

Microsoft Entra admin
Not configured

Server name
mysqlserver1974luiscoco.database.windows.net



## All resources

particular (luiscocoenriquezhotmail.onmicrosoft.c...

## mysqlserver1974luiscoco | Networking

Public Endpoints allow access to this resource through the internet using a public IP address. An application or resource that is granted access with the following network rules still requires proper authorization to access this resource. Learn more

Public network access

○ Disable

● Selected networks

Connections from the IP addresses configured in the Firewall rules section below will have access to this database. By default, no public IP addresses are allowed. Learn more

Please save public network access value before adding new virtual networks.

### Virtual networks

Allow virtual networks to connect to your resource using service endpoints. Learn more

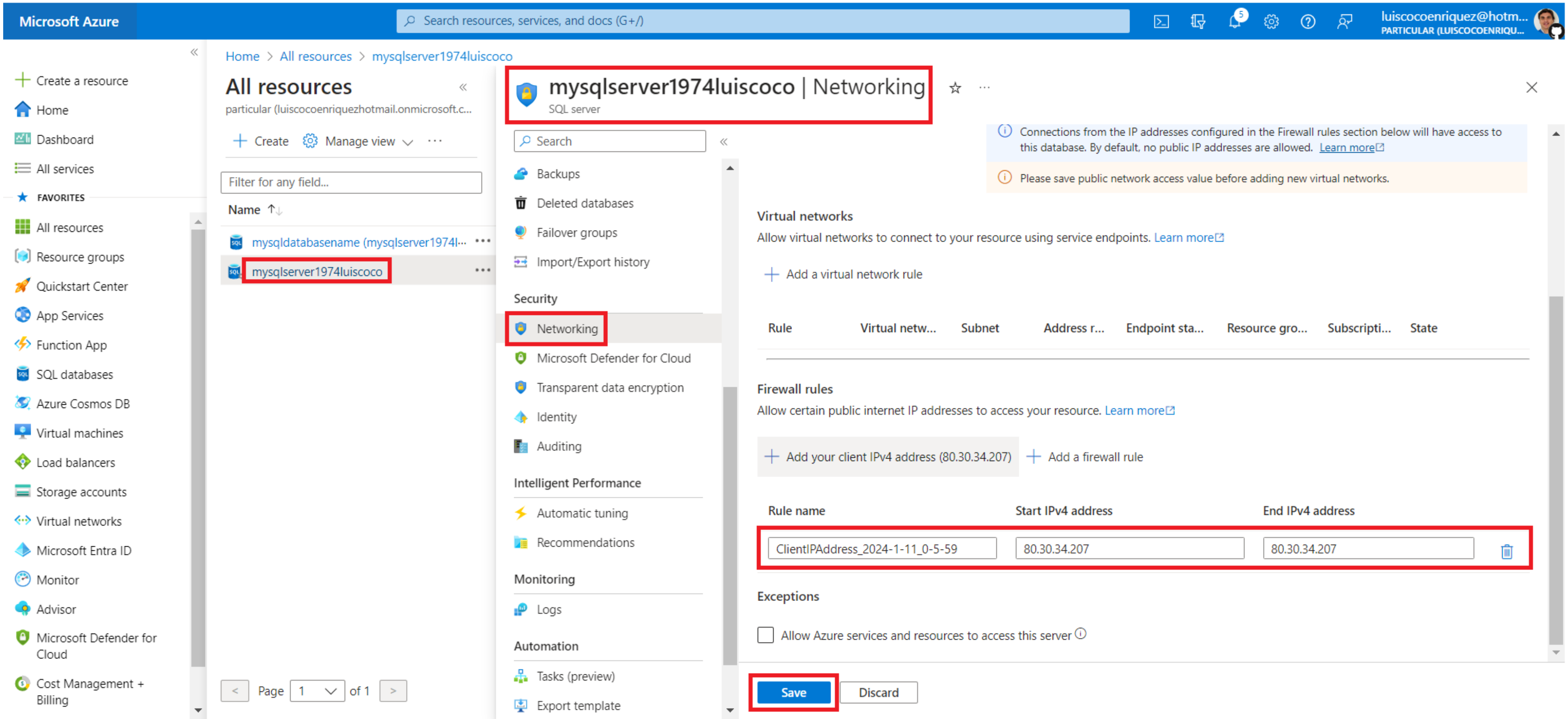
+ Add a virtual network rule

| Rule | Virtual netw... | Subnet | Address r... | Endpoint sta... | Resource gro... | Subscripti... | State |
|---|---|---|---|---|---|---|---|

### Firewall rules

Allow certain public internet IP addresses to access your resource. Learn more

+ Add your client IPv4 address (80.30.34.207) + Add a firewall rule



Connections from the IP addresses configured in the Firewall rules section below will have access to this database. By default, no public IP addresses are allowed. Learn more

Please save public network access value before adding new virtual networks.

Allow virtual networks to connect to your resource using service endpoints. Learn more

Allow certain public internet IP addresses to access your resource. Learn more

| Rule name | Start IPv4 address | End IPv4 address |
|---|---|---|
| ClientIPAddress_2024-1-11_0-5-59 | 80.30.34.207 | 80.30.34.207 |

☐ Allow Azure services and resources to access this server

# 2. Create .NET8 WebAPI CRUD Microservice

## 2.1. appsettings.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection": "Server=tcp:mysqlserver1974luiscoco.database.windows.net,1433;Initial Catalog=mysqldatabasename;Persist
  }
}
```

## 2.2. Program.cs

```
using Microsoft.EntityFrameworkCore;
using AzureSQLWebAPIMicroservice.Data;
using AzureSQLWebAPIMicroservice.Services;
using Microsoft.OpenApi.Models;
using Microsoft.EntityFrameworkCore.SqlServer;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddDbContext<ExampleDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));
builder.Services.AddScoped<ExampleModelService>();

builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen(c =>
{
    c.SwaggerDoc("v1", new OpenApiInfo { Title = "My API", Version = "v1" });
});

var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseAuthorization();

app.MapControllers();

app.Run();
```

## 2.3. Models (ExampleModel.cs)

### ExampleModel.cs

```
namespace AzureSQLWebAPIMicroservice.Models
{
    public class ExampleModel
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Description { get; set; }
        public DateTime CreatedDate { get; set; }
    }
}
```

## 2.4. Service (ExampleModelService.cs)

### ExampleModelService.cs

```
using AzureSQLWebAPIMicroservice.Data;
using AzureSQLWebAPIMicroservice.Models;
using Microsoft.EntityFrameworkCore;
using System.Collections.Generic;
using System.Threading.Tasks;
```

```
namespace AzureSQLWebAPIMicroservice.Services
{
    public class ExampleModelService
    {
        private readonly ExampleDbContext _context;

        public ExampleModelService(ExampleDbContext context)
        {
            _context = context;
        }

        // Create
        public async Task<ExampleModel> AddExampleModel(ExampleModel model)
        {
            _context.ExampleModels.Add(model);
            await _context.SaveChangesAsync();
            return model;
        }

        // Read all
        public async Task<List<ExampleModel>> GetAllExampleModels()
        {
            return await _context.ExampleModels.ToListAsync();
        }

        // Read by ID
        public async Task<ExampleModel> GetExampleModelById(int id)
        {
            return await _context.ExampleModels.FirstOrDefaultAsync(e => e.Id == id);
        }

        // Update
        public async Task<ExampleModel> UpdateExampleModel(int id, ExampleModel model)
        {
            var existingModel = await _context.ExampleModels.FirstOrDefaultAsync(e => e.Id == id);
            if (existingModel == null)
            {
                return null;
            }

            existingModel.Name = model.Name;
            // Update other properties as necessary

            _context.Entry(existingModel).State = EntityState.Modified;
            await _context.SaveChangesAsync();

            return existingModel;
        }

        // Delete
        public async Task<bool> DeleteExampleModel(int id)
        {
            var model = await _context.ExampleModels.FindAsync(id);
            if (model == null)
            {
                return false;
            }

            _context.ExampleModels.Remove(model);
            await _context.SaveChangesAsync();

            return true;
        }
    }
}
```

## 2.5. Data (ExampleDbContext.cs)

### ExampleDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using AzureSQLWebAPIMicroservice.Models;
```

```
namespace AzureSQLWebAPIMicroservice.Data
{
    public class ExampleDbContext:DbContext
    {
        public ExampleDbContext(DbContextOptions<ExampleDbContext> options)
        : base(options)
        {
        }

        public DbSet<ExampleModel> ExampleModels { get; set; }

        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            // Configure the primary key for ExampleModel
            modelBuilder.Entity<ExampleModel>().HasKey(e => e.Id);

            // Configure some properties with more details
            modelBuilder.Entity<ExampleModel>()
                .Property(e => e.Name)
                .IsRequired() // Makes the Name field required
                .HasMaxLength(100); // Sets maximum length of the Name field to 100 characters

            modelBuilder.Entity<ExampleModel>()
                .Property(e => e.Description)
                .HasMaxLength(255); // Sets maximum length of the Description field to 255 characters

            // Set a default value for the CreatedDate field
            modelBuilder.Entity<ExampleModel>()
                .Property(e => e.CreatedDate)
                .HasDefaultValueSql("GETDATE()"); // This will use the SQL Server GETDATE() function to set the default value

            // Seed data
            modelBuilder.Entity<ExampleModel>().HasData(
                new ExampleModel { Id = 1, Name = "Sample Name 1", Description = "Sample Description 1", CreatedDate = DateTime.N
                new ExampleModel { Id = 2, Name = "Sample Name 2", Description = "Sample Description 2", CreatedDate = DateTime.N
                // Add more seed data as needed
            );
        }
    }
}
```

## 2.6. Controllers (ExampleModelsController.cs)

### ExampleModelsController.cs

```
using Microsoft.AspNetCore.Mvc;
using AzureSQLWebAPIMicroservice.Models;
using AzureSQLWebAPIMicroservice.Services;
using System.Threading.Tasks;

namespace AzureSQLWebAPIMicroservice.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ExampleModelsController : ControllerBase
    {
        private readonly ExampleModelService _service;

        public ExampleModelsController(ExampleModelService service)
        {
            _service = service;
        }

        // POST: api/ExampleModels
        [HttpPost]
        public async Task<ActionResult<ExampleModel>> PostExampleModel(ExampleModel model)
        {
            var createdModel = await _service.AddExampleModel(model);
            return CreatedAtAction(nameof(GetExampleModel), new { id = createdModel.Id }, createdModel);
        }
        // GET: api/ExampleModels
        [HttpGet]
        public async Task<ActionResult<IEnumerable<ExampleModel>>> GetExampleModels()
        {
            return await _service.GetAllExampleModels();
        }

        // GET: api/ExampleModels/5
        [HttpGet("{id}")]
        public async Task<ActionResult<ExampleModel>> GetExampleModel(int id)
        {
            var model = await _service.GetExampleModelById(id);

            if (model == null)
            {
                return NotFound();
            }

            return model;
        }

        // PUT: api/ExampleModels/5
        [HttpPut("{id}")]
        public async Task<IActionResult> PutExampleModel(int id, ExampleModel model)
        {
            if (id != model.Id)
            {
                return BadRequest();
            }

            var updatedModel = await _service.UpdateExampleModel(id, model);

            if (updatedModel == null)
            {
                return NotFound();
            }
            return NoContent();
        }

        // DELETE: api/ExampleModels/5
        [HttpDelete("{id}")]
        public async Task<IActionResult> DeleteExampleModel(int id)
        {
            var success = await _service.DeleteExampleModel(id);

            if (!success)
            {
                return NotFound();
            }

            return NoContent();
        }
    }
}
```

## IMPORTANT:

In the AzureSQLWebAPIMicroservice.csproj set **InvariantGlobalization** to **false**

```
<InvariantGlobalization>false</InvariantGlobalization>
```

This is the whole **AzureSQLWebAPIMicroservice.csproj**

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net8.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
    <InvariantGlobalization>false</InvariantGlobalization>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.EntityFrameworkCore" Version="8.0.1" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="8.0.1">
      <PrivateAssets>all</PrivateAssets>
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
    </PackageReference>
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="8.0.1" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="6.4.0" />
  </ItemGroup>

</Project>
```

## 3. Add First Migration

Add the package **Microsoft.EntityFrameworkCore.Design**

Add/create first migration with this command:

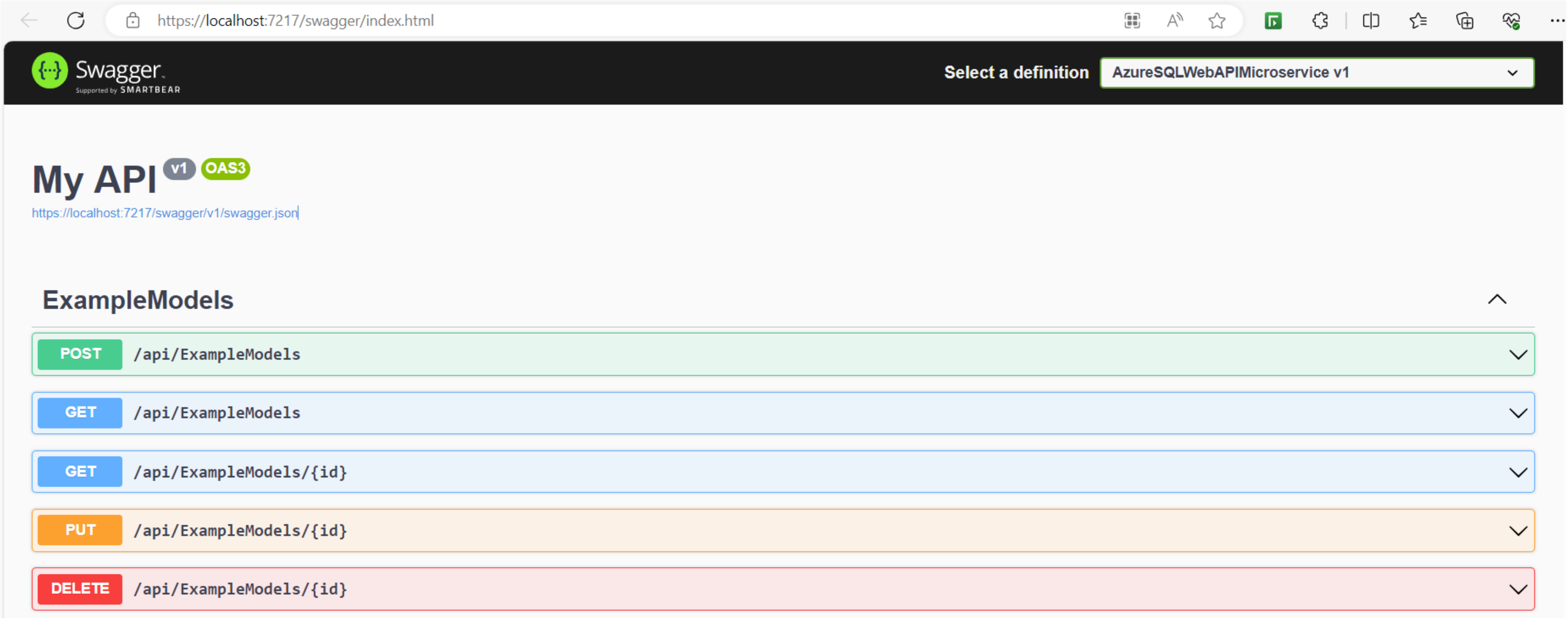
```
dotnet ef migrations add InitialCreate
```

Also update the database with this command

```
dotnet ef database update
```

## 4. Verify application

https://localhost:7217/swagger/index.html



AzureSQLWebAPIMicroservice v1

My API v1 OAS3

https://localhost:7217/swagger/v1/swagger.json

ExampleModels

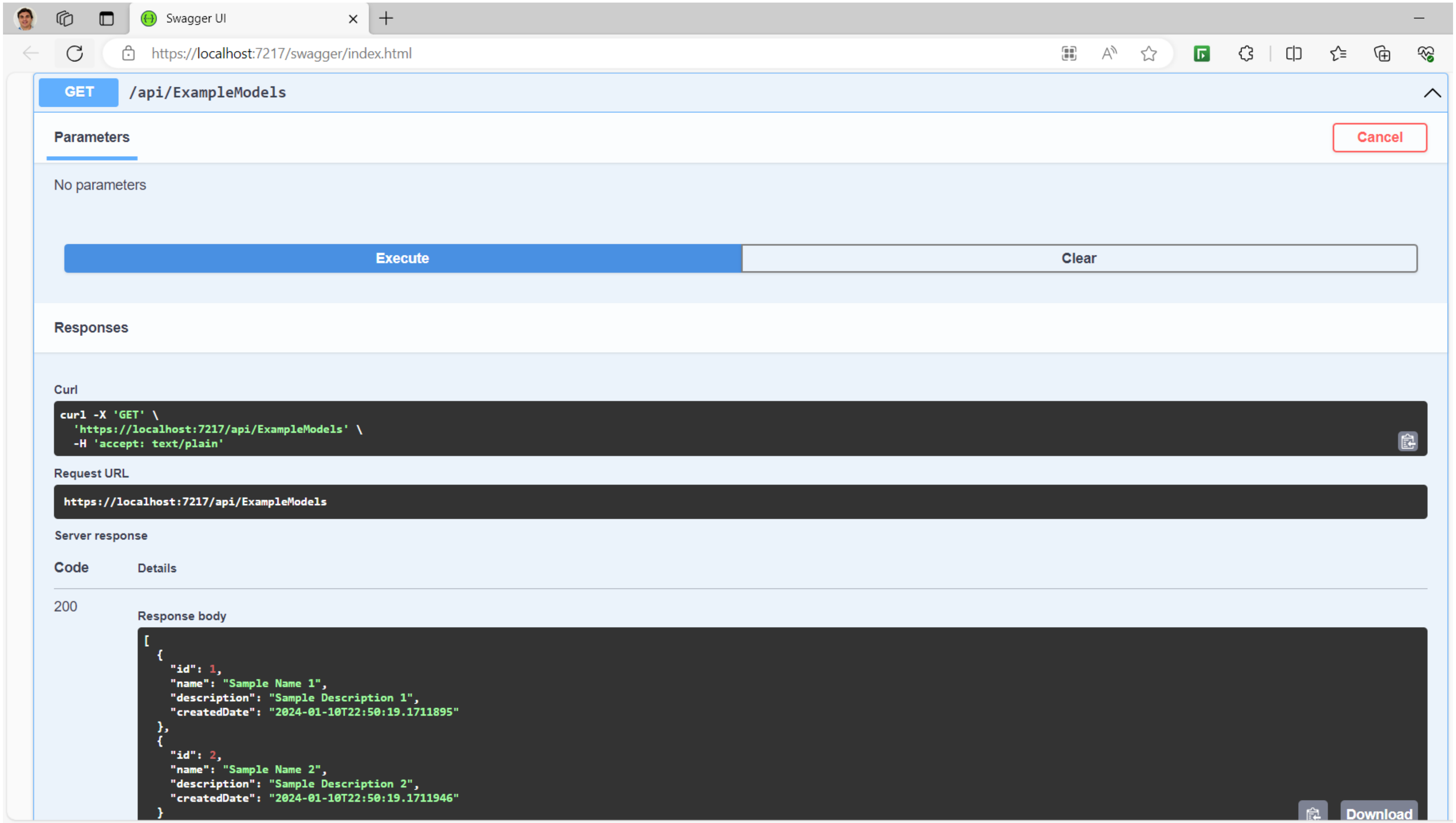
POST /api/ExampleModels

GET /api/ExampleModels

GET /api/ExampleModels/{id}

PUT /api/ExampleModels/{id}

DELETE /api/ExampleModels/{id}



No parameters

Curl

```
curl -X 'GET' \
  'https://localhost:7217/api/ExampleModels' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7217/api/ExampleModels
```

Server response

200

Response body

```
[
  {
    "id": 1,
    "name": "Sample Name 1",
    "description": "Sample Description 1",
    "createdDate": "2024-01-10T22:50:19.1711895"
  },
  {
    "id": 2,
    "name": "Sample Name 2",
    "description": "Sample Description 2",
    "createdDate": "2024-01-10T22:50:19.1711946"
  }
```