PROYECTO ASP.Net parte 6

Despliegue

1. Inicializador - Configuración de Despliegue

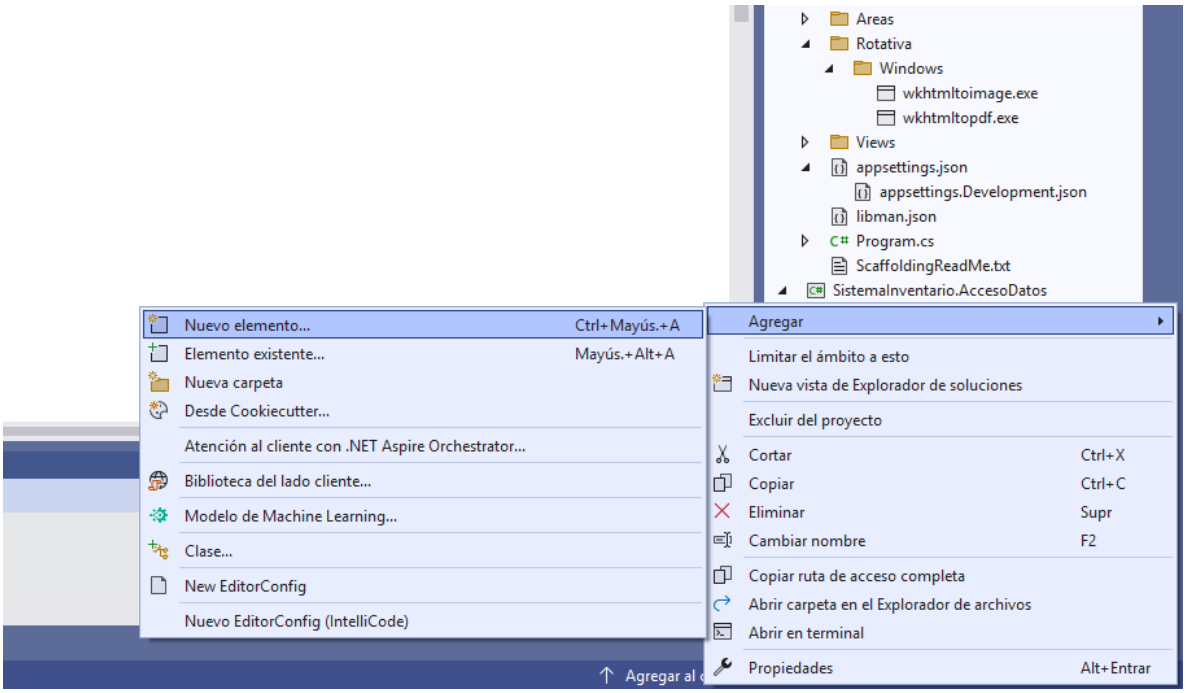
- Copiamos el contenido de SistemaInventario/appsettings.json en SistemaInventario/appsettings.json/appsettings.Development.json

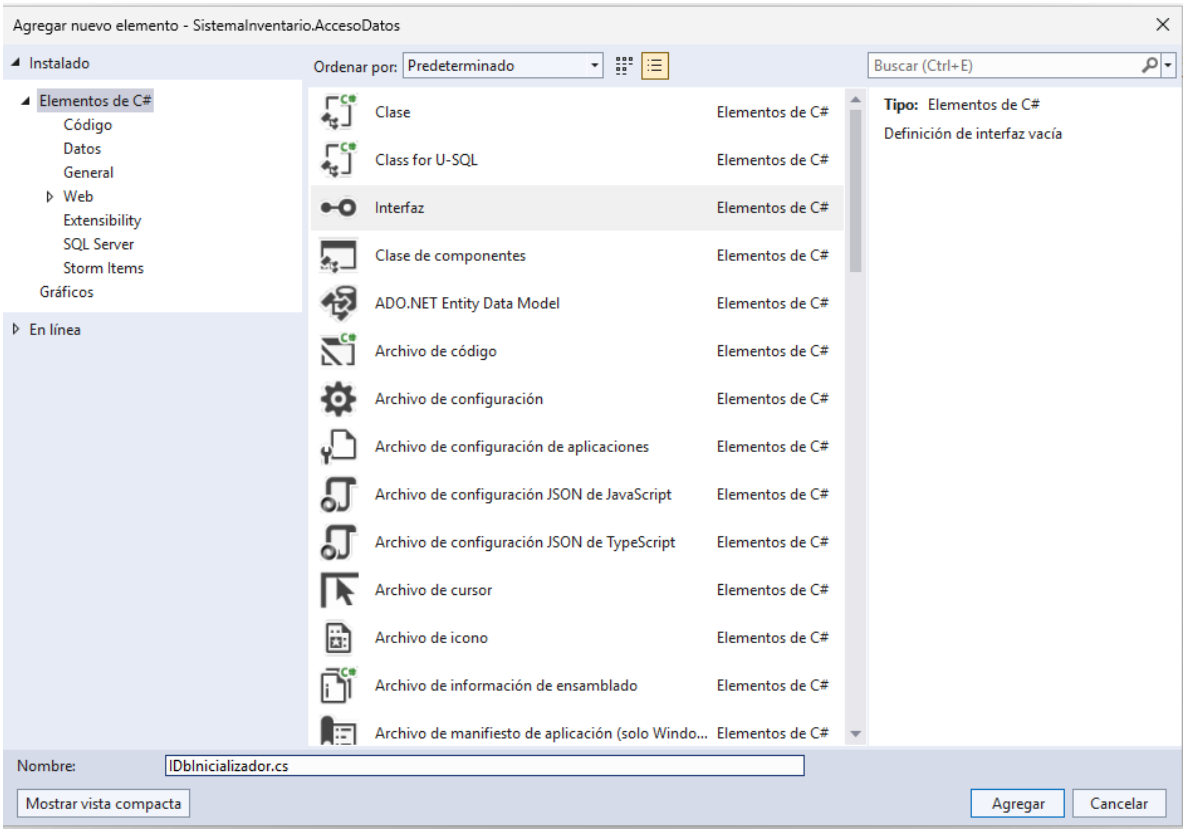
- En SistemaInventario/appsettings.json cambiamos el nombre de la base de datos.

"ApplicationDbContextConnection": "Server=(localdb)\\mssqllocaldb;Database=SistemaInventarioProduccion;

- En SistemaInventario.AccesoDatos creamos una carpeta llamada Inicializador

- En SistemaInventario.AccesoDatos/Inicializador creamos una interfaz llamada IDbInicializador.cs






```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Inicializador
{
   public interface IDbInicializador
   {
      void Inicializar();
   }
}
```

- En SistemaInventario.AccesoDatos/Inicializador creamos una clase llamada DbInicializador.cs

```
using Microsoft.AspNetCore.Identity;
using Microsoft.EntityFrameworkCore;
using SistemaInventario.AccesoDatos.Data;
using SistemaInventario.Modelos;
using SistemaInventario.Utilidades;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Threading.Tasks;

namespace SistemaInventario.AccesoDatos.Inicializador
{
   public class DbInicializador : IDbInicializador
   {
      private readonly ApplicationDbContext _db;
      private readonly UserManager<IdentityUser> _userManager;
      private readonly RoleManager<IdentityRole> _roleManager;

      public DbInicializador(ApplicationDbContext db, UserManager<IdentityUser> userManager,
                             RoleManager<IdentityRole> roleManager)
      {
         _db = db;
         _userManager = userManager;
         _roleManager = roleManager;
      }

      public void Inicializar()
      {
         try
         {
            if (_db.Database.GetPendingMigrations().Count() > 0)
            {
               _db.Database.Migrate();  // Ejecuta las migraciones pendientes
            }
         }
         catch (Exception)
         {

            throw;
         }

         // Datos Iniciales
         if (_db.Roles.Any(r => r.Name == DS.Role_Admin)) return;

         _roleManager.CreateAsync(new IdentityRole(DS.Role_Admin)).GetAwaiter().GetResult();
         _roleManager.CreateAsync(new IdentityRole(DS.Role_Cliente)).GetAwaiter().GetResult();
         _roleManager.CreateAsync(new IdentityRole(DS.Role_Inventario)).GetAwaiter().GetResult();

         // Crea un usuario administrador
         _userManager.CreateAsync(new UsuarioAplicacion
         {
            UserName = "bgj827@gmail.com",
            Email = "bgj827@gmail.com",
            EmailConfirmed = true,
            Nombres = "Jorge",
            Apellidos = "Bolaños"
         }, "Admin123*").GetAwaiter().GetResult();

         // Asigna el Rol al usuario
         UsuarioAplicacion usuario = _db.UsuarioAplicacion.Where(u => u.UserName ==
"bgj827@gmail.com").FirstOrDefault();
```

```
            _userManager.AddToRoleAsync(usuario, DS.Role_Admin).GetAwaiter().GetResult();
        }
    }
}
```

- En SistemaInventario/Program.cs agregamos el servicio de la nueva interfaz

```
.
.
.
builder.Services.Configure<StripeSettings>(builder.Configuration.GetSection("Stripe"));

builder.Services.AddScoped<IDbInicializador, DbInicializador>();

var app = builder.Build();
.
.
.
app.UseAuthentication();
app.UseAuthorization();

// Aplicar Migraciones y Datos Iniciales
using (var scope = app.Services.CreateScope())
{
    var services = scope.ServiceProvider;
    var loggerFactory = services.GetRequiredService<ILoggerFactory>();

    try
    {
        var inicializador = services.GetRequiredService<IDbInicializador>();
        inicializador.Inicializar();
    }
    catch (Exception ex)
    {

        var logger = loggerFactory.CreateLogger<Program>();
        logger.LogError(ex, "Un Error ocurrio al ejecutar la migracion");
    }
}
.
.
.
```

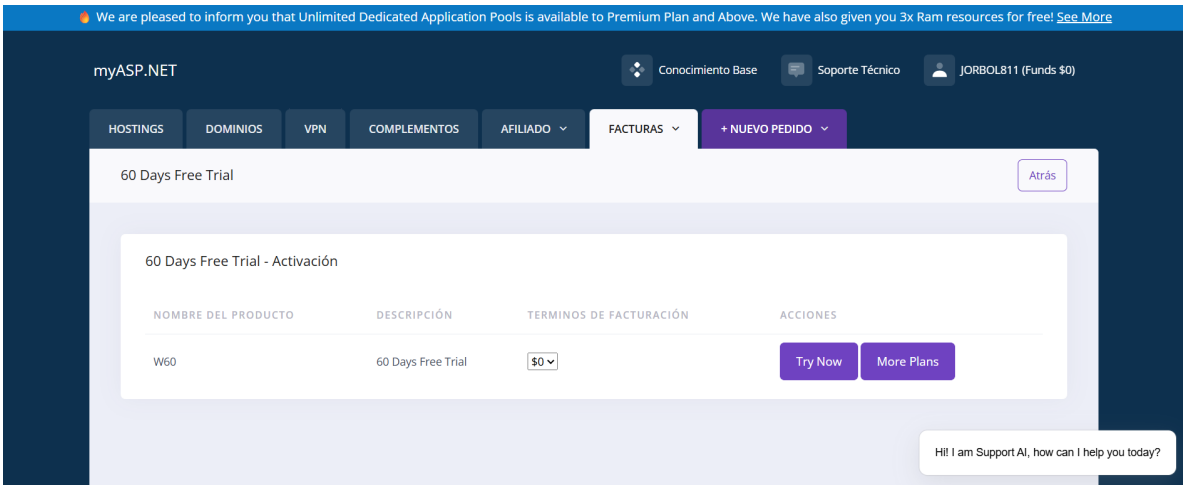
2. Subir la Aplicacion a Hosting MyASP Parte 1

- Hosting recomendados para subir aplicaciones ASP.Net

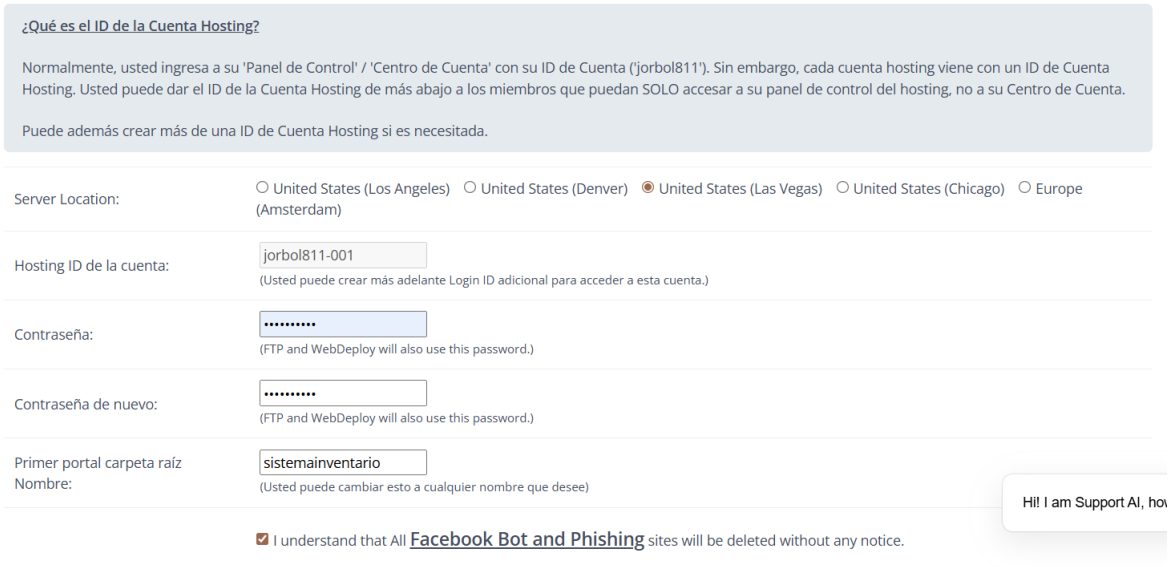
https://dotnet.microsoft.com/en-us/apps/aspnet/hosting

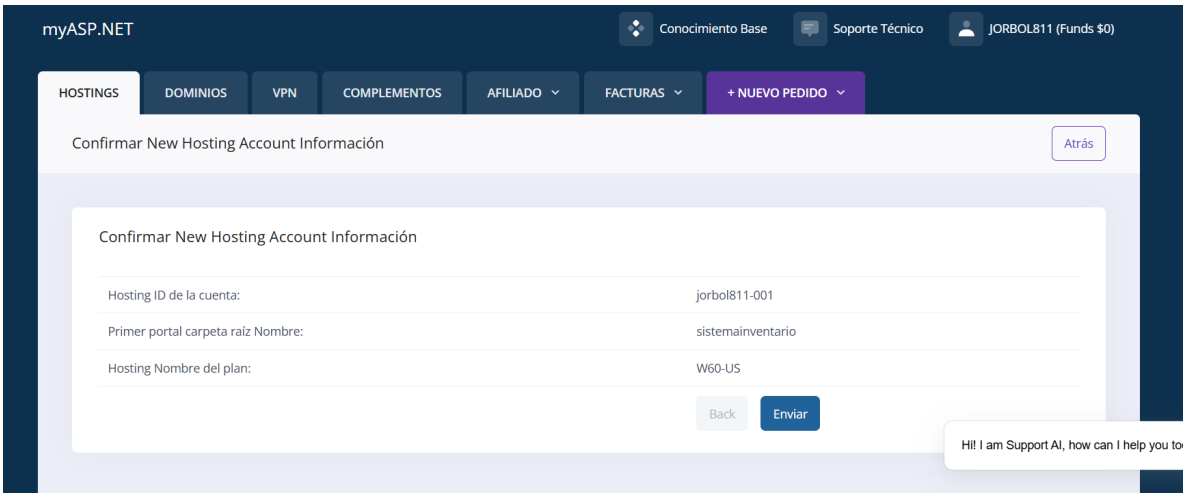
- Registrarse en myASP.NET

https://www.myasp.net/

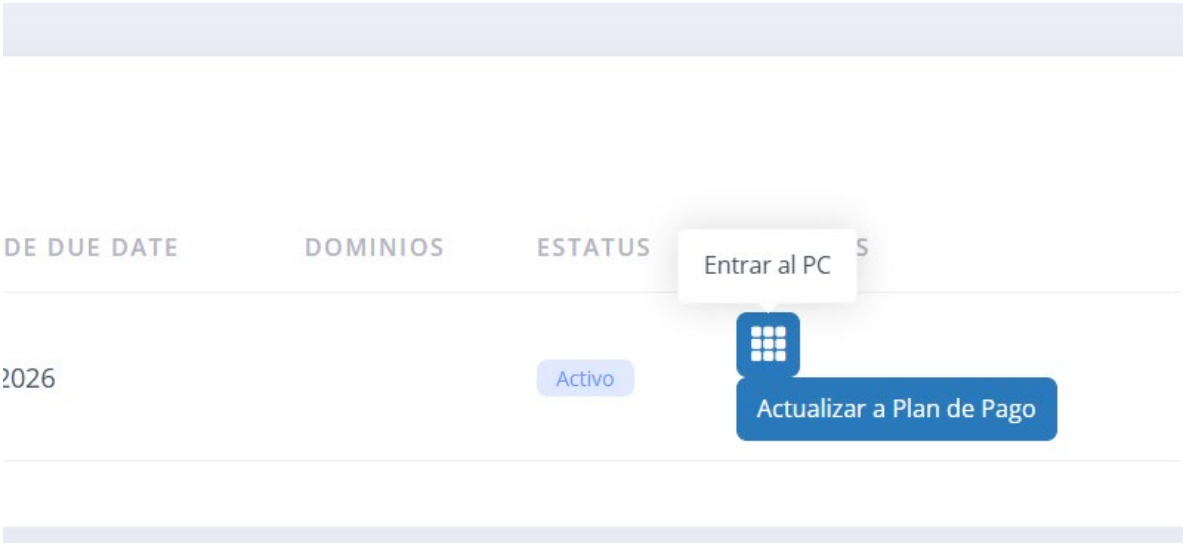


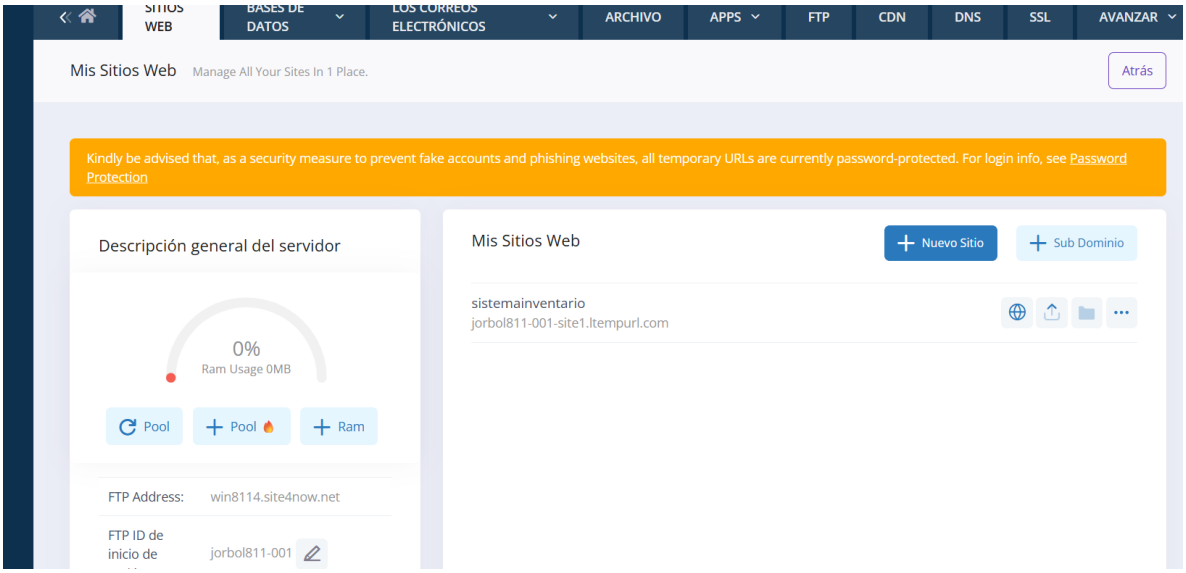
- Click en el botón Try Now
- Llenar la información requerida.





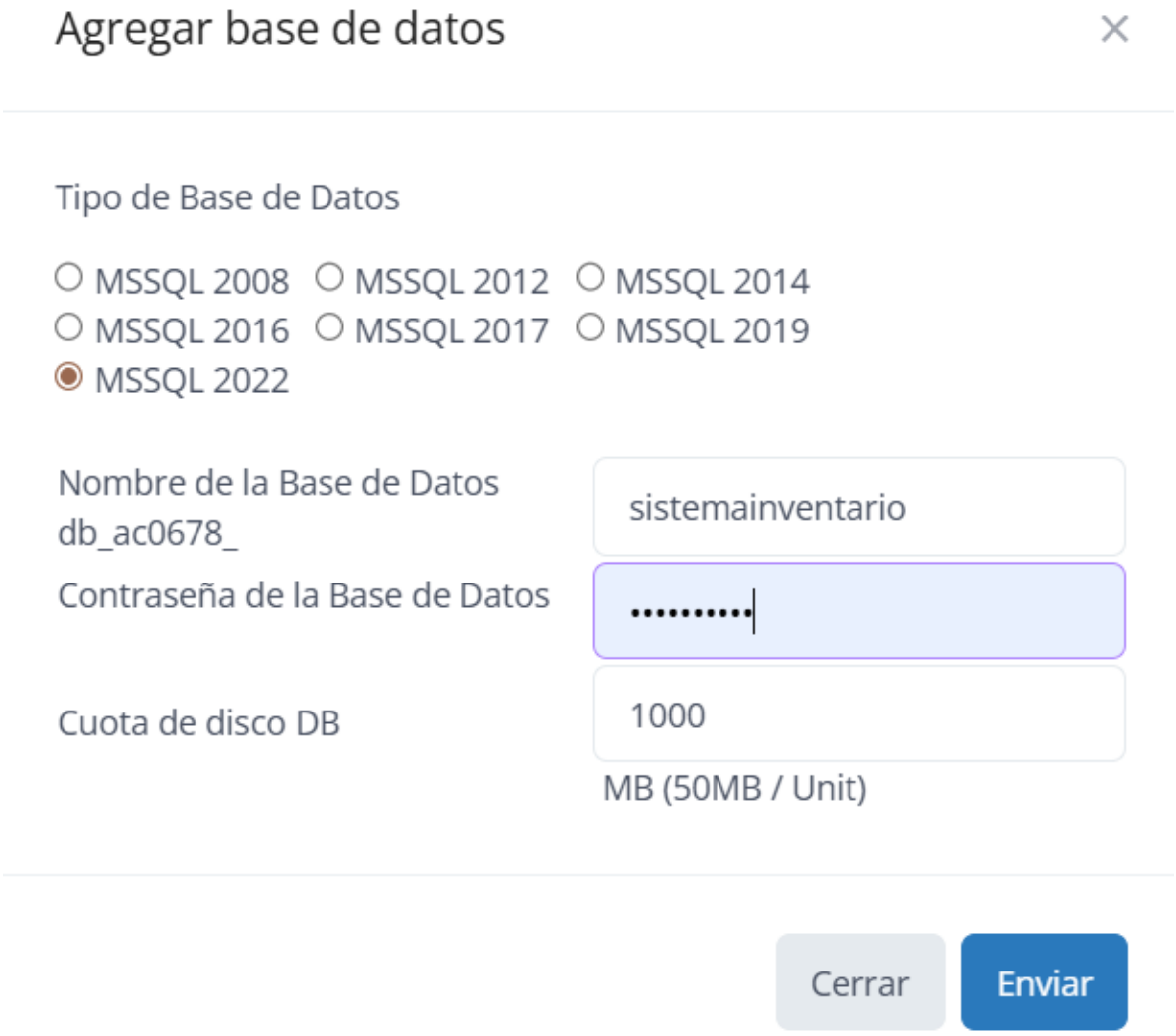
- Verificar correo con información de conexiones ftp y otras conexiones.
- Ingresar al Control Panel



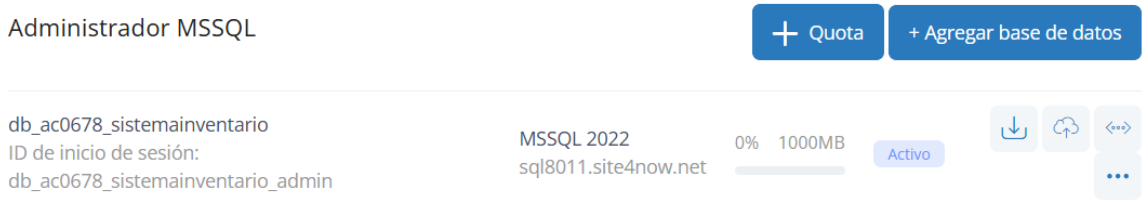


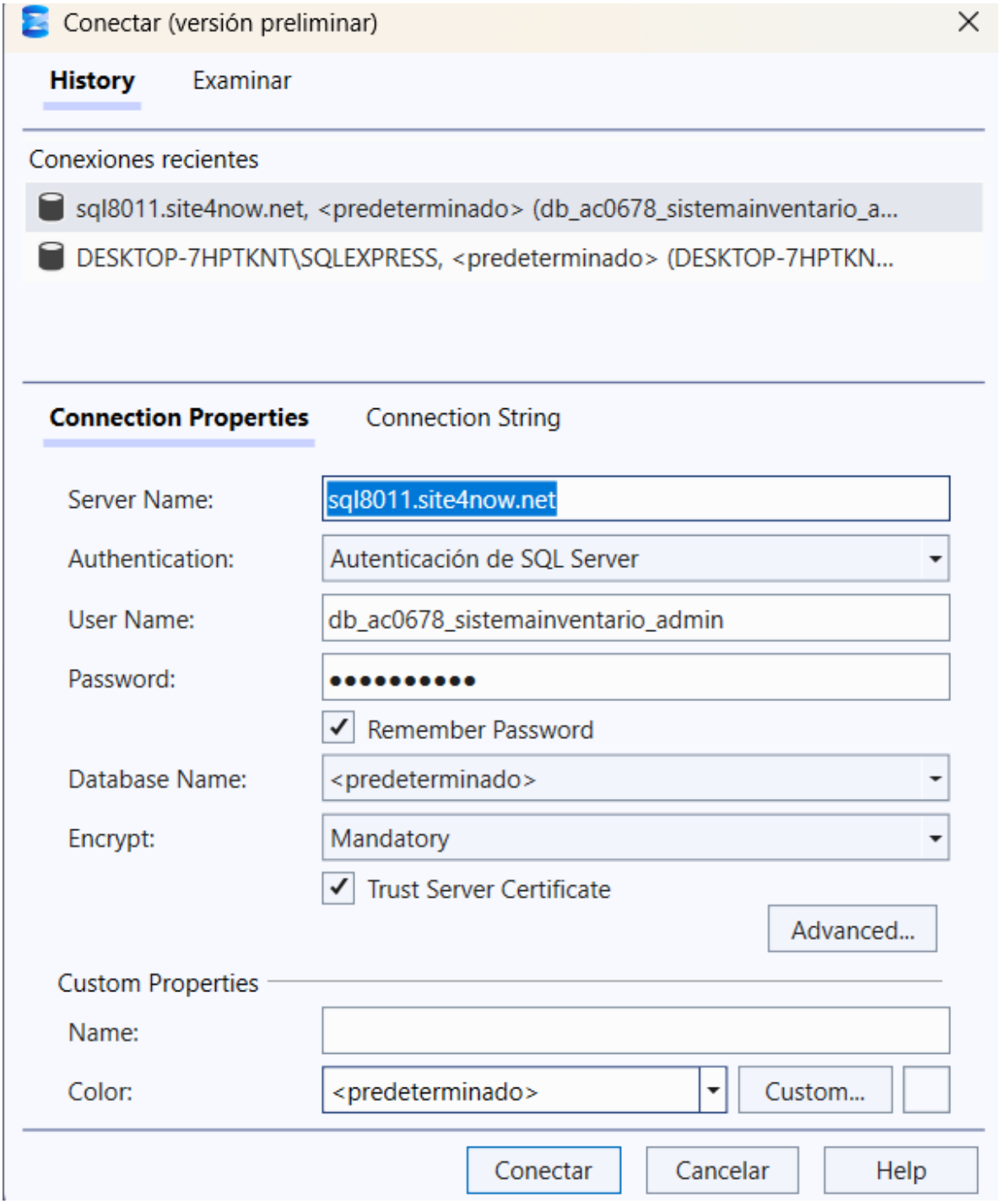
- Crear base de datos

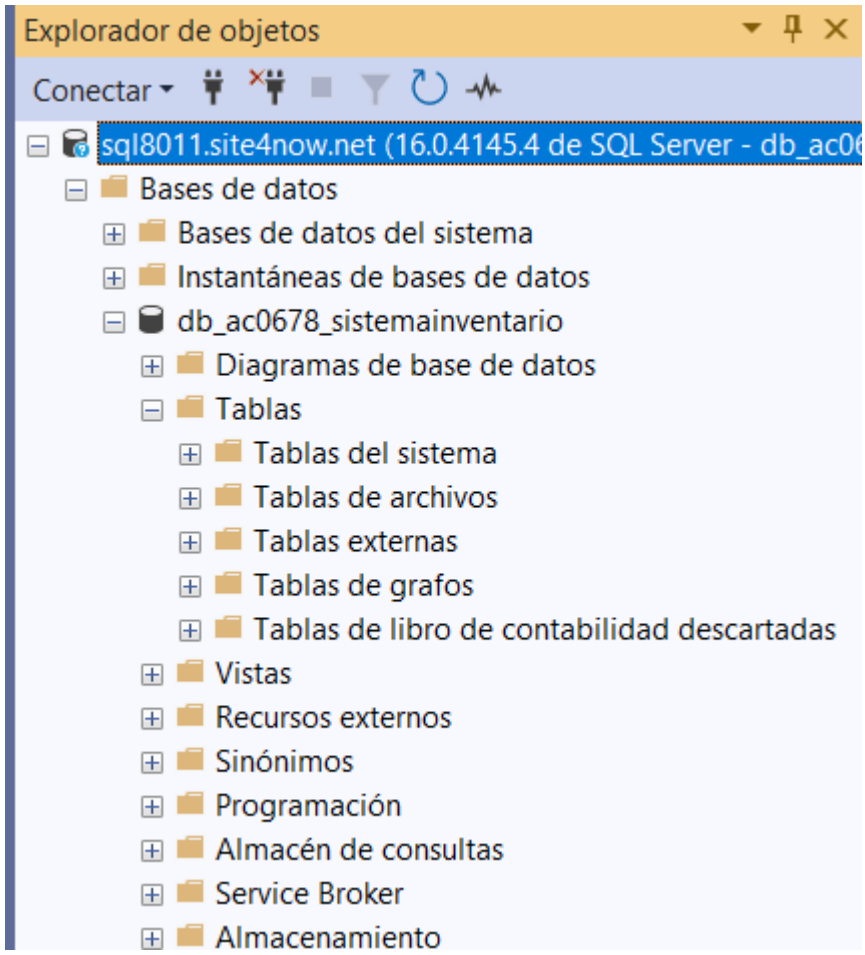

SITIOS WEB
BASES DE DATOS
LOS CORREOS ELECTRÓNICOS
Mis Sitios Web
MSSQL
MySQL
SQL Reporting
Copia de seguridad personalizada avanzada
Kindly be advised that,
Protection
unts and phishing w
Descripción gene
Mis Sitios Web
sistemainventario
jorbol811-001-site1.lt
0%



- No conectamos a la base de datos desde el SQL Server Management con los datos proporcionados.

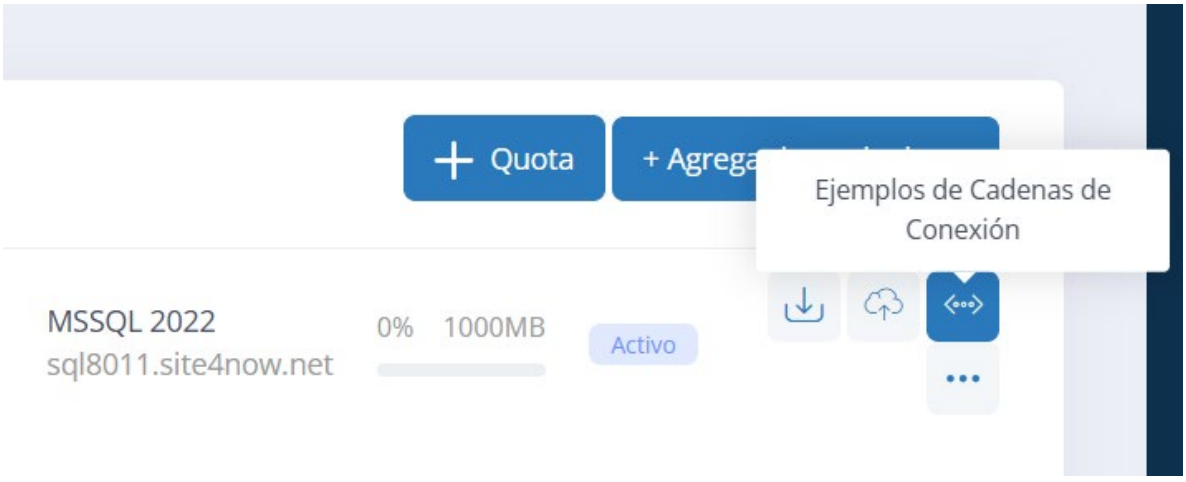



Conectar (versión preliminar)
History
Examinar
Conexiones recientes
sql8011.site4now.net, <predeterminado> (db_ac0678_sistemainventario_a...
DESKTOP-7HPTKNT\SQLEXPRESS, <predeterminado> (DESKTOP-7HPTKN...
Connection Properties
Connection String
Server Name:
sql8011.site4now.net
Authentication:
Autenticación de SQL Server
User Name:
db_ac0678_sistemainventario_admin
Password:
Remember Password
Database Name:
<predeterminado>
Encrypt:
Mandatory
Trust Server Certificate
Advanced...
Custom Properties
Name:
Color:
<predeterminado>
Custom...
Conectar
Cancelar
Help



3. Subir la Aplicacion a Hosting MyASP Parte 2

- Modificamos del archivo SistemaInventario/appsettings.json cambiando los datos de conexión al servidor remoto.



Ejemplos de Cadenas de Conexión

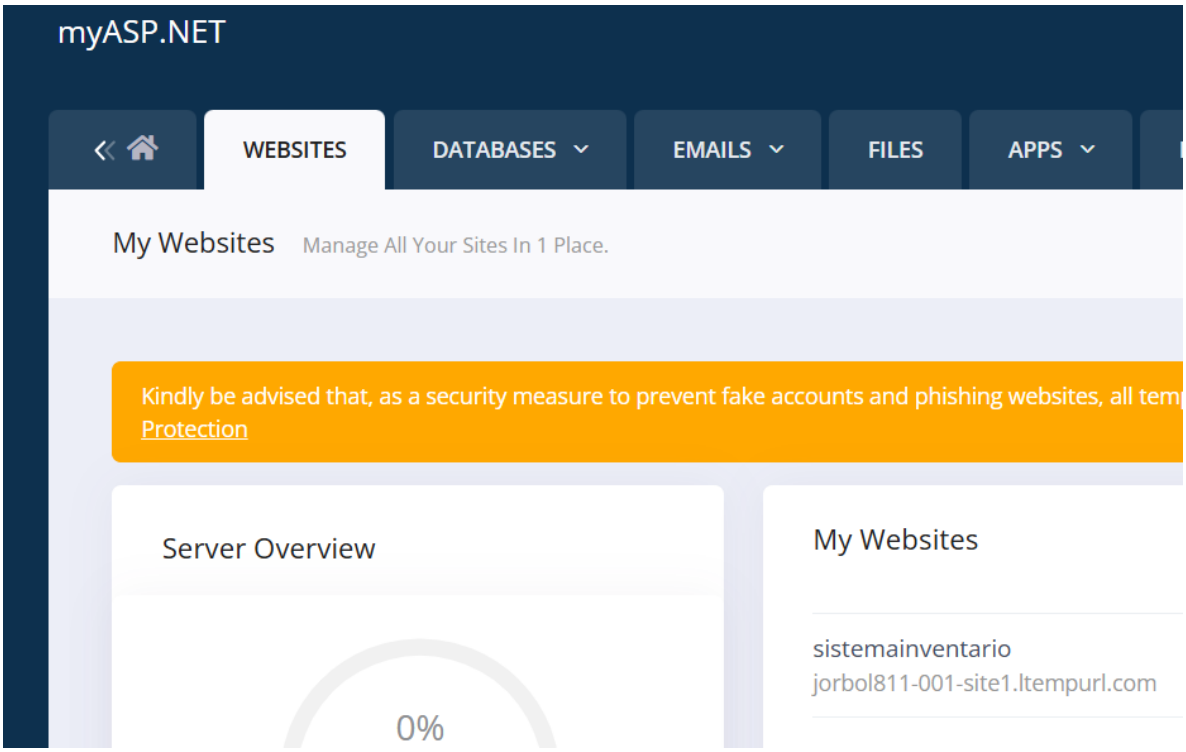
ASP.NET

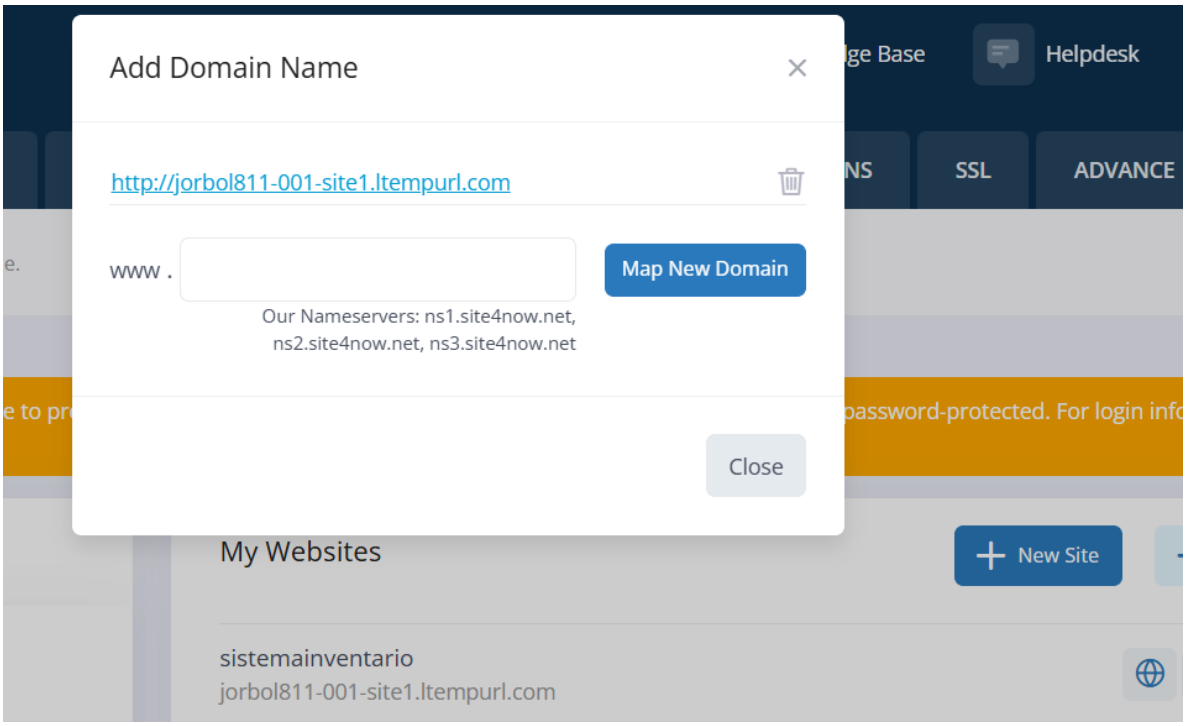
```
"Data Source=SQL8011.site4now.net;Initial Catalog=db_ac0678_sistemainventario;User Id=db_ac0678_sistemainventari
```

Classic ASP

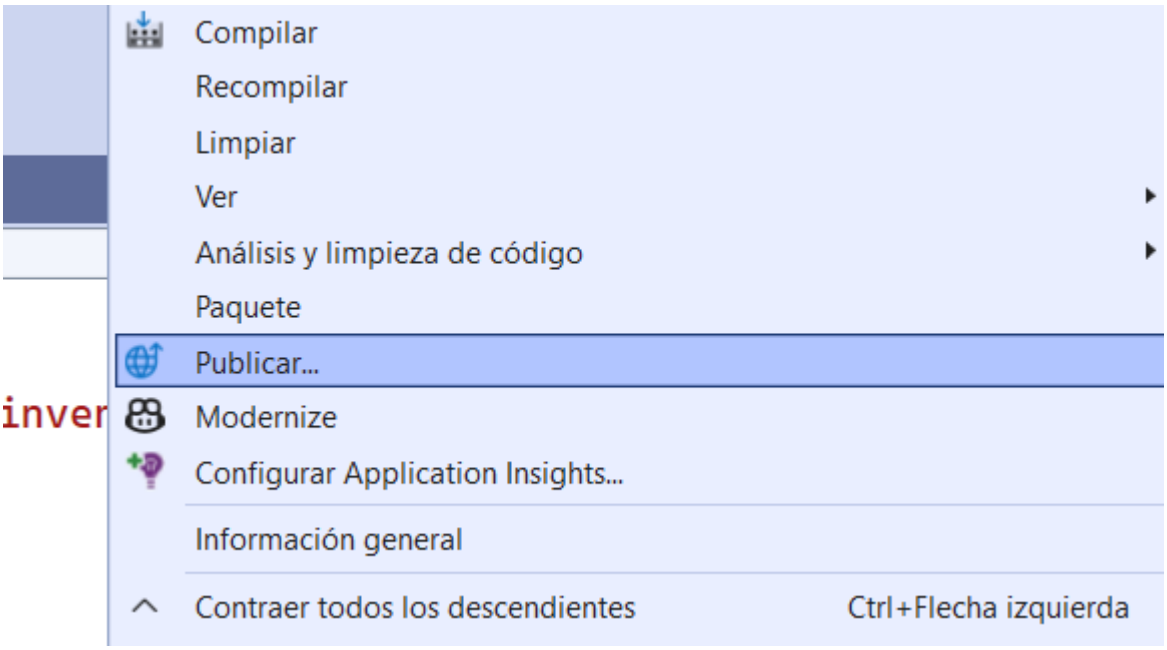
```
"Provider=SQLOLEDB;Data Source=SQL8011.site4now.net;Initial Catalog=;User Id=db_ac0678_sistemainventario_admin;P
```

- Modificamos del archivo SistemaInventario/appsettings.json cambiando los datos de la url remota.

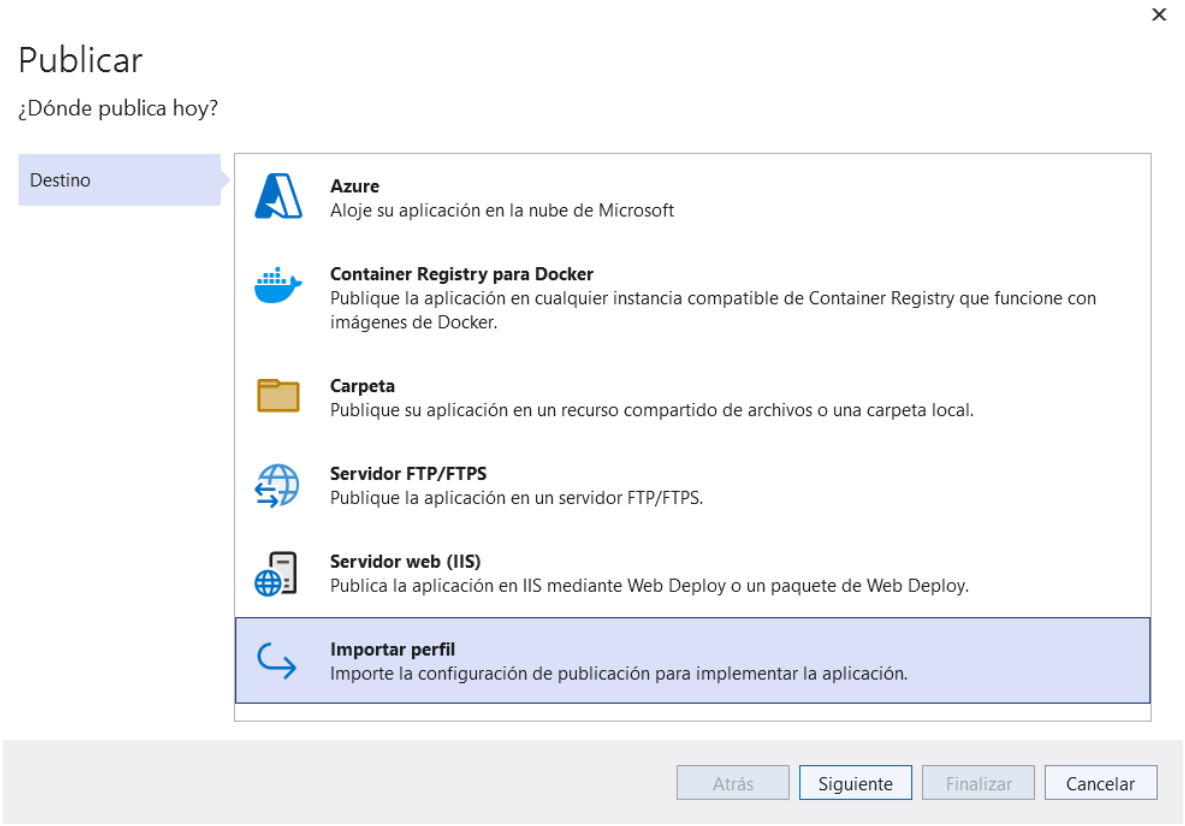


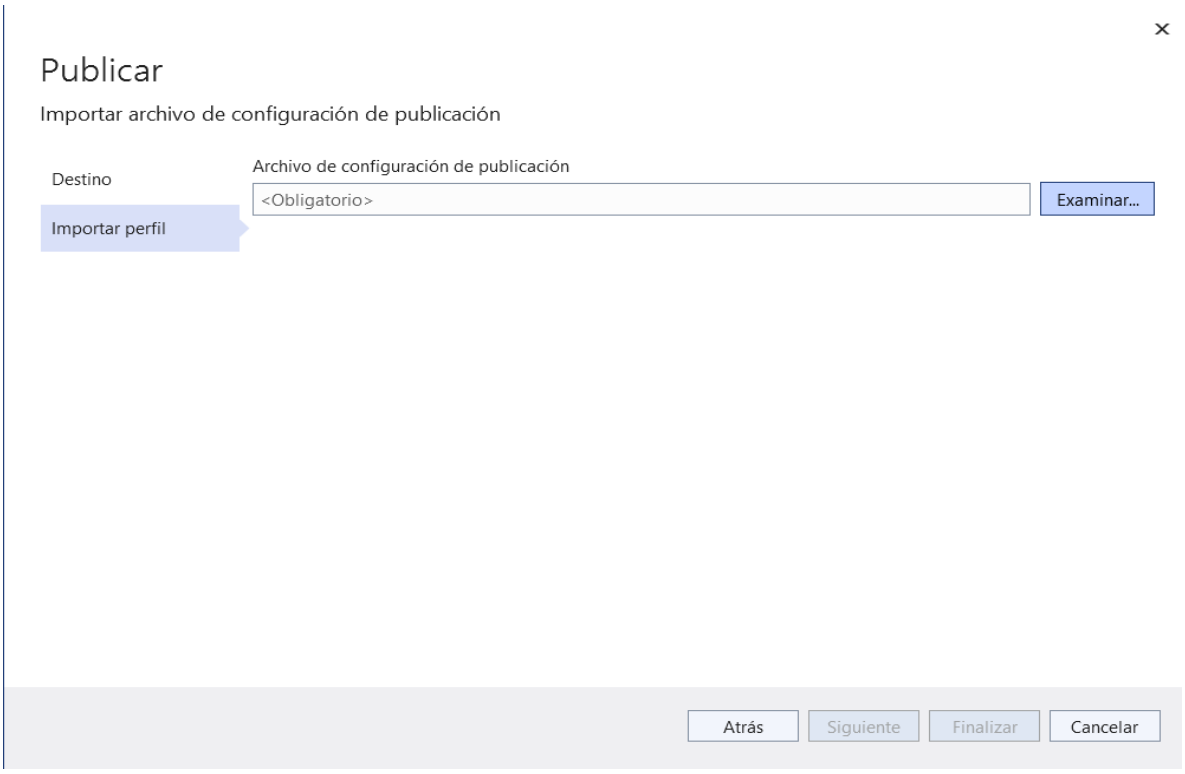


- Publicamos el proyecto principal haciendo click derecho sobre el proyecto en Visual Studio.

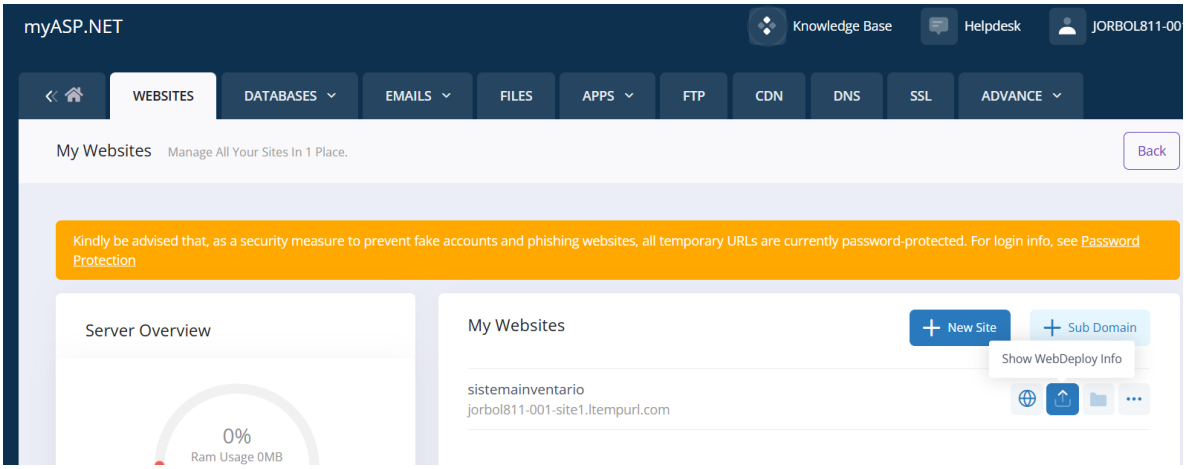


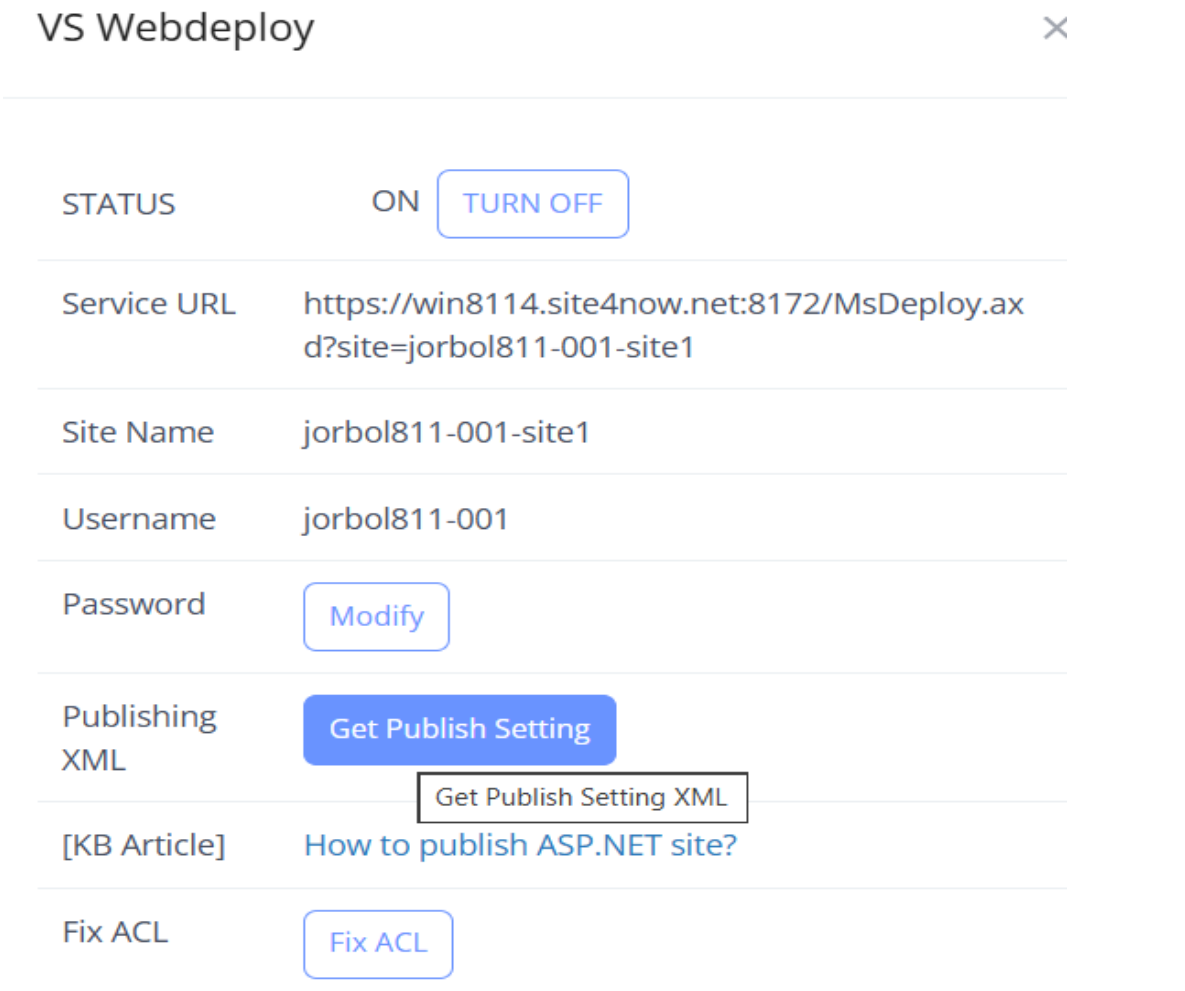
- Seleccionamos la opción importar perfil



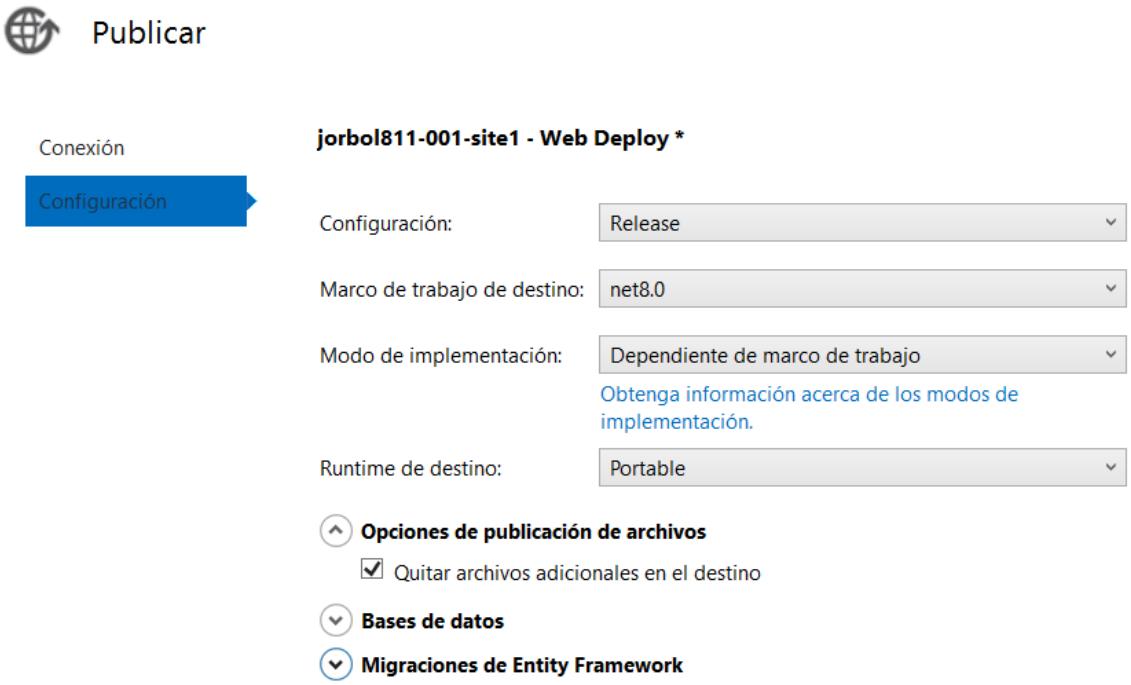


- El archivo que nos pide, lo descargamos del cpanel/Show WebDeploy Info

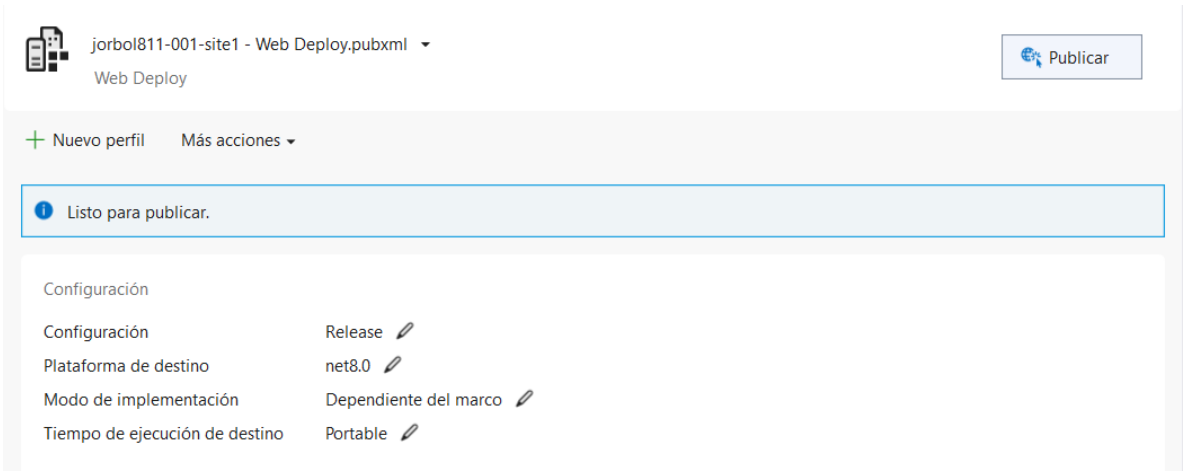




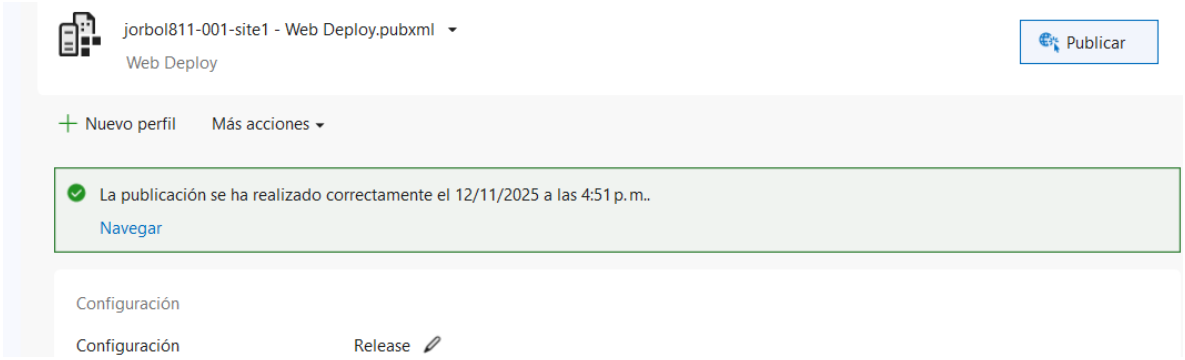
- Damos Click en Mostrar todas las configuraciones



- Presionamos el botón publicar.



- Ingresamos la contraseña del sitio y comienza el despliegue.



- Verificamos la base de datos en el SQL Server Management