

PROYECTO ASP.Net parte 6

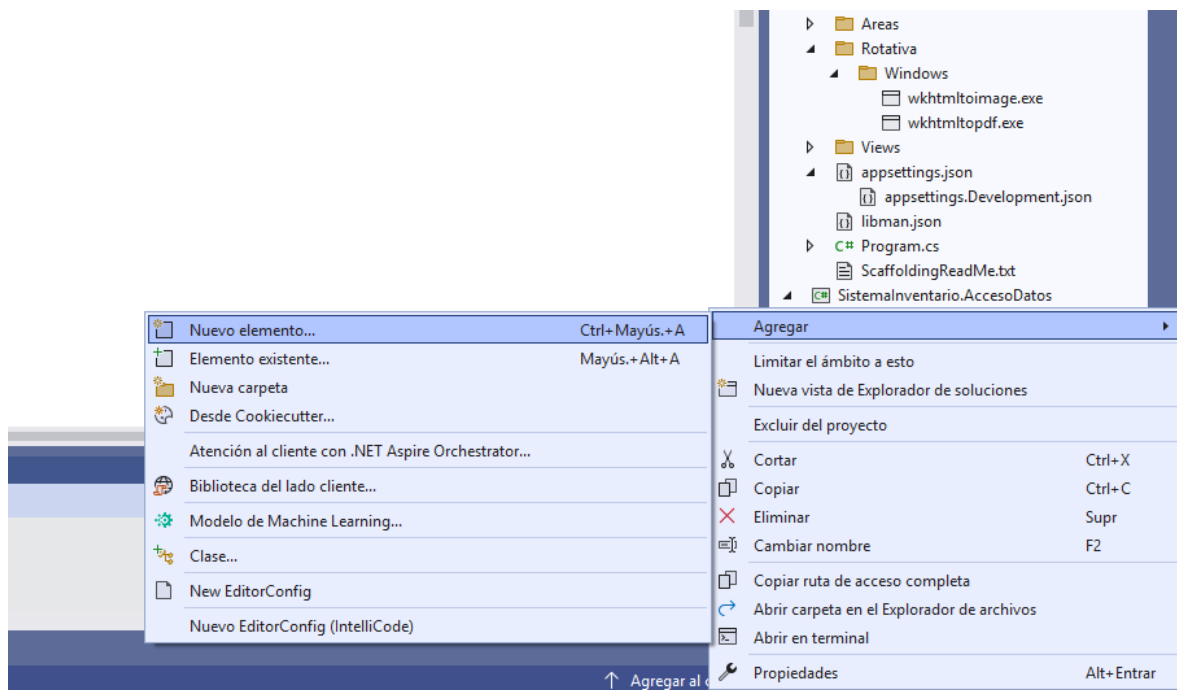
Despliegue

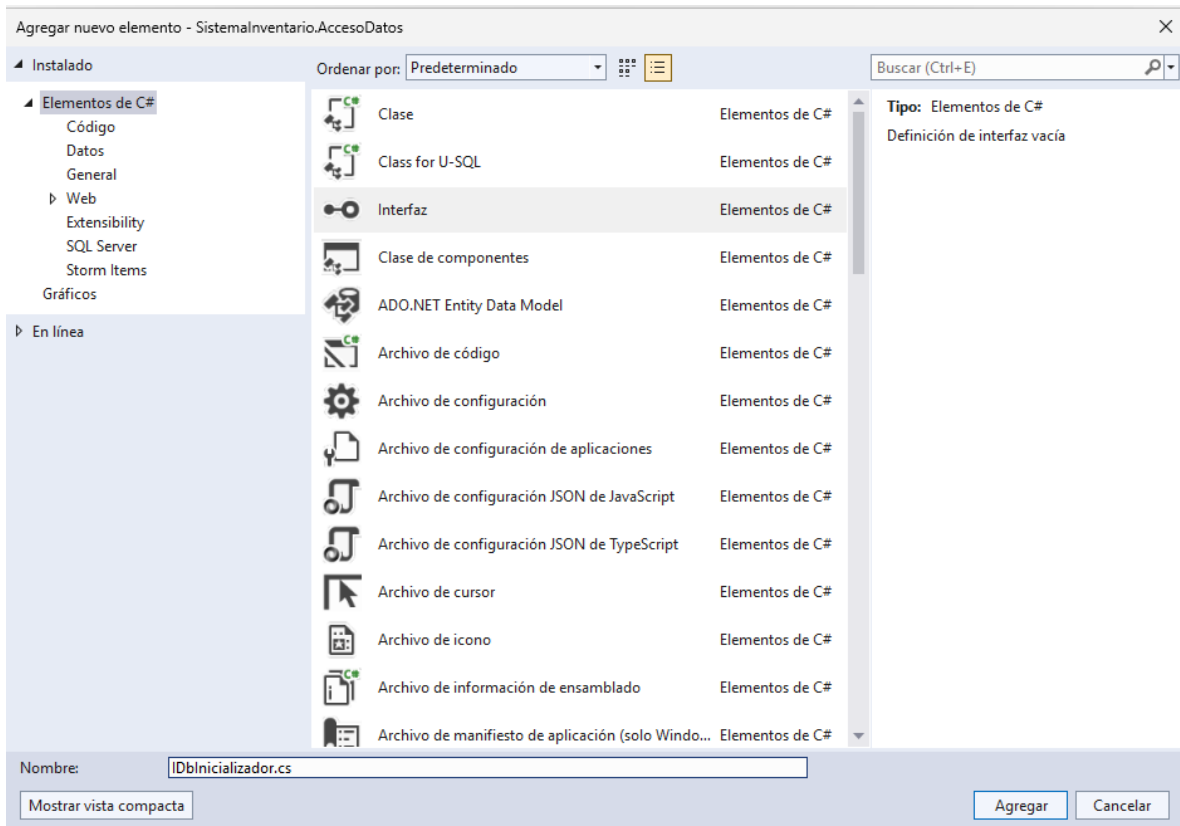
1. Inicializador - Configuración de Despliegue

- 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)\u005Cmssqllocaldb;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 IDbIniciador
    {
        void Inicializar();
    }
}
```

- En SistemaInventario.AccesoDatos/Iniciador creamos una clase llamada DbIniciador.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<IDbIniciador, DbIniciador>();

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<IDbIniciador>();
        inicializador.Inicializar();
    }
    catch (Exception ex)
    {
        var logger = loggerFactory.CreateLogger<Program>();
        logger.LogError(ex, "Un Error ocurrio al ejecutar la migracion");
    }
}

.
.
.

```

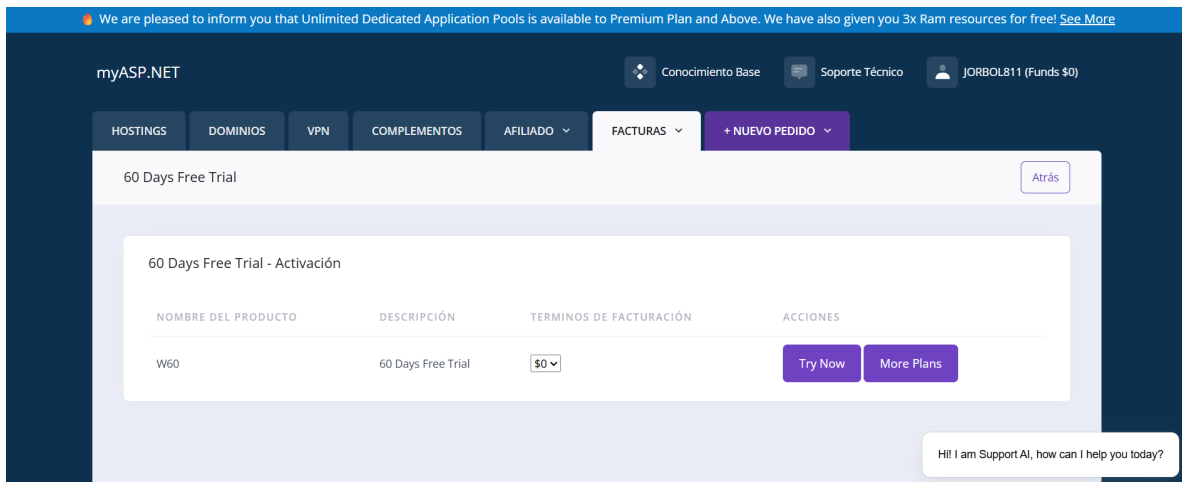
2. Subir la Aplicacion a Hosting MyASP Parte 1

- 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.

¿Qué es el ID de la Cuenta Hosting?

Normalmente, usted ingresa a su 'Panel de Control' / 'Centro de Cuenta' con su ID de Cuenta ('jorbol811'). Sin embargo, cada cuenta hosting viene con un ID de Cuenta Hosting. Usted puede dar el ID de la Cuenta Hosting de más abajo a los miembros que puedan SOLO acceder a su panel de control del hosting, no a su Centro de Cuenta.

Puede además crear más de una ID de Cuenta Hosting si es necesitada.

Server Location: ☐ United States (Los Angeles) ☐ United States (Denver) ☒ United States (Las Vegas) ☐ United States (Chicago) ☐ Europe (Amsterdam)

Hosting ID de la cuenta:
(Usted puede crear más adelante Login ID adicional para acceder a esta cuenta.)

Contraseña:
(FTP and WebDeploy will also use this password.)

Contraseña de nuevo:
(FTP and WebDeploy will also use this password.)

Primer portal carpeta raíz Nombre:
(Usted puede cambiar esto a cualquier nombre que desee)

☒ I understand that All [Facebook Bot and Phishing](#) sites will be deleted without any notice.

Hi! I am Support AI, how can I help you today?

myASP.NET

Conocimiento Base Soporte Técnico JORBOL811 (Funds \$0)

HOSTINGS DOMINIOS VPN COMPLEMENTOS AFILIADO FACTURAS + NUEVO PEDIDO

Confirmar New Hosting Account Información

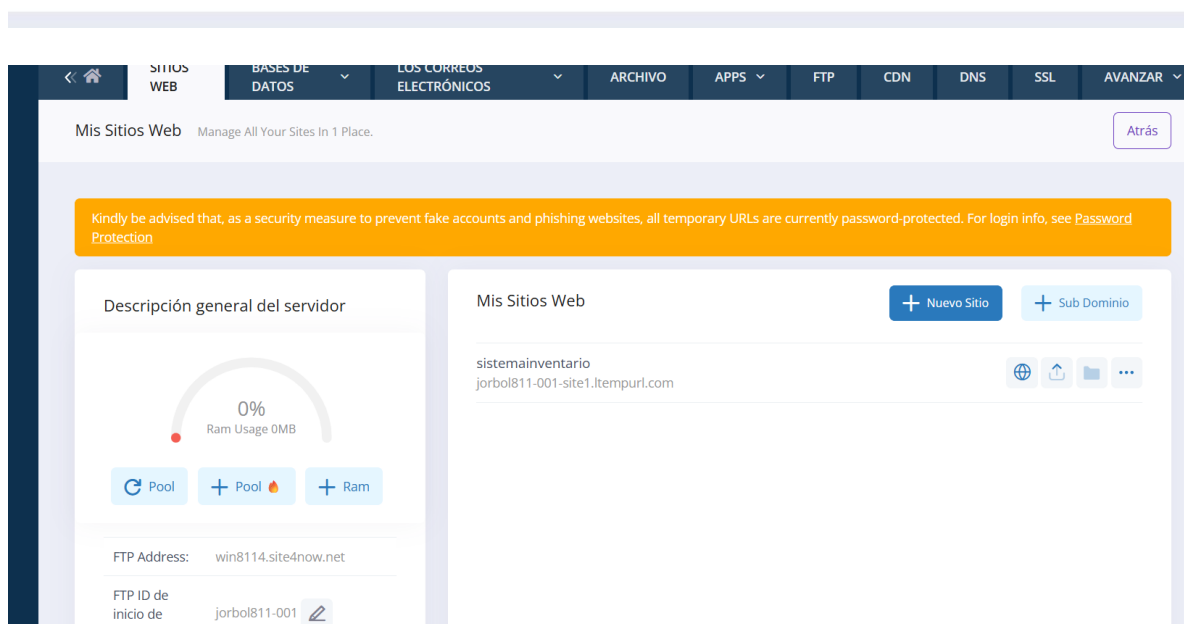
Confirmar New Hosting Account Información

Hosting ID de la cuenta:	jorbol811-001
Primer portal carpeta raíz Nombre:	sistemainventario
Hosting Nombre del plan:	W60-US

Back Enviar

Hi! I am Support AI, how can I help you today?

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



- Crear base de datos

Agregar base de datos



Tipo de Base de Datos

- ☐ MSSQL 2008 ☐ MSSQL 2012 ☐ MSSQL 2014
☐ MSSQL 2016 ☐ MSSQL 2017 ☐ MSSQL 2019
☒ MSSQL 2022

Nombre de la Base de Datos
db_ac0678_

sistemainventario

Contraseña de la Base de Datos

.....

Cuota de disco DB

1000

MB (50MB / Unit)

Cerrar

Enviar

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

Administrador MSSQL

+ Quota

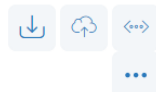
+ Agregar base de datos

db_ac0678_sistemainventario
ID de inicio de sesión:
db_ac0678_sistemainventario_admin

MSSQL 2022
sql8011.site4now.net

0% 1000MB

Activo





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:

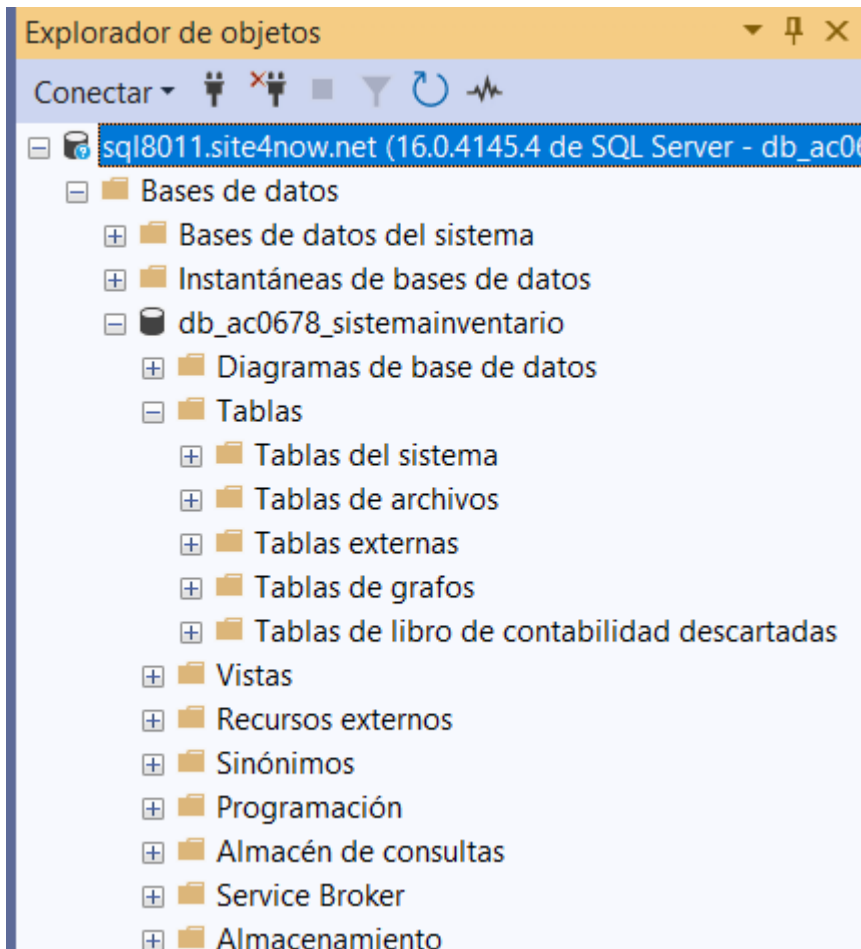
<predeterminado>

Custom...

Conectar

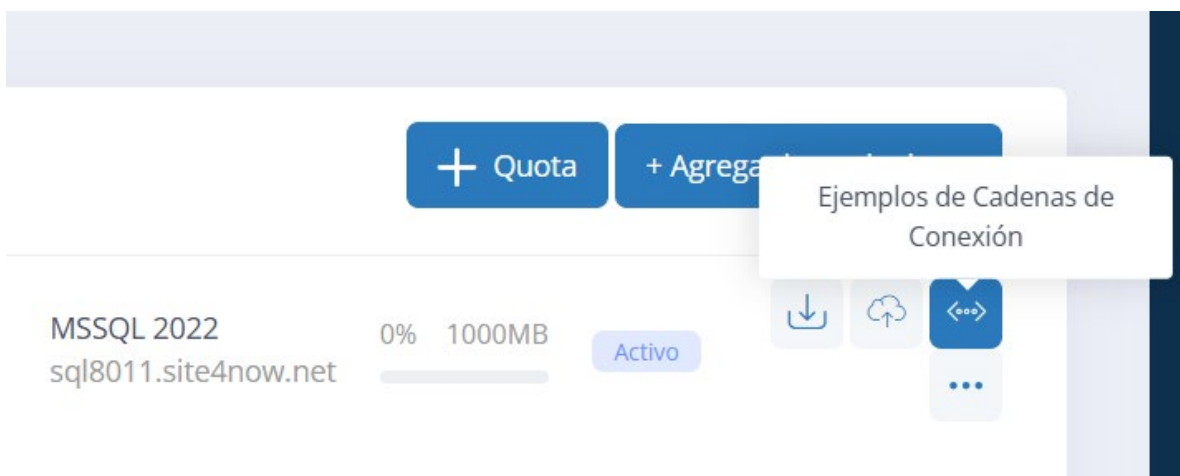
Cancelar

Help



3. Subir la Aplicacion a Hosting MyASP Parte 2

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



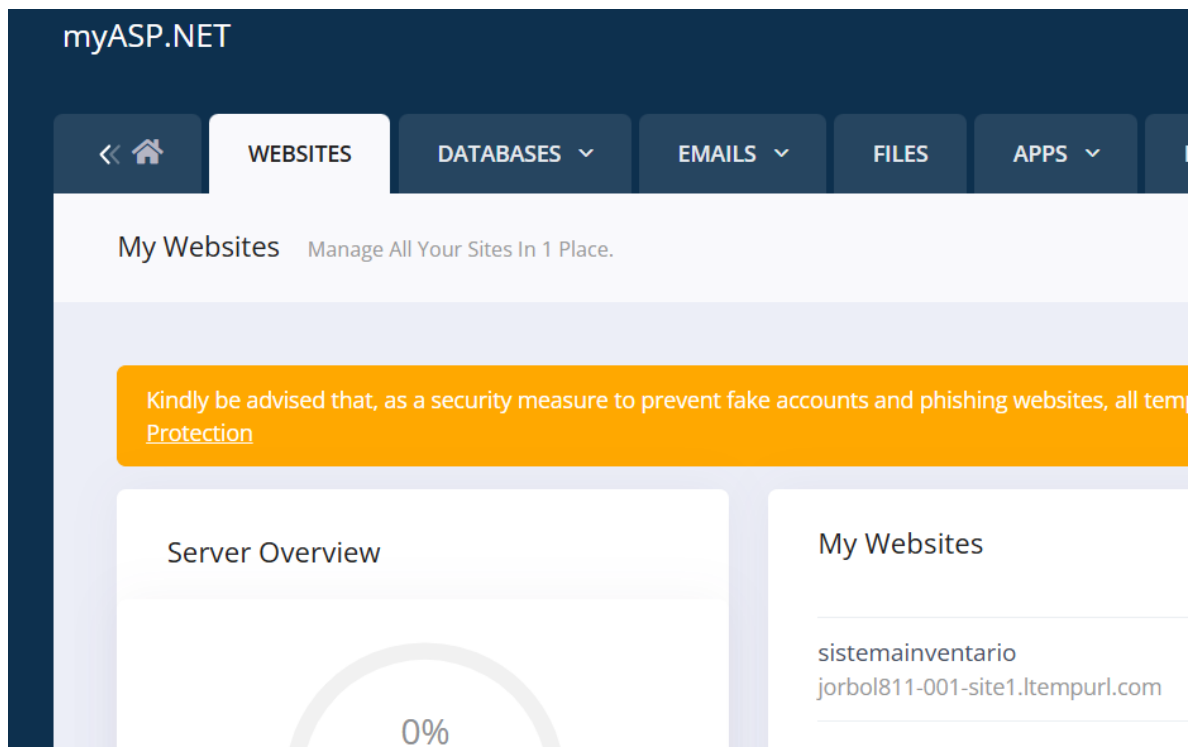
ASP.NET

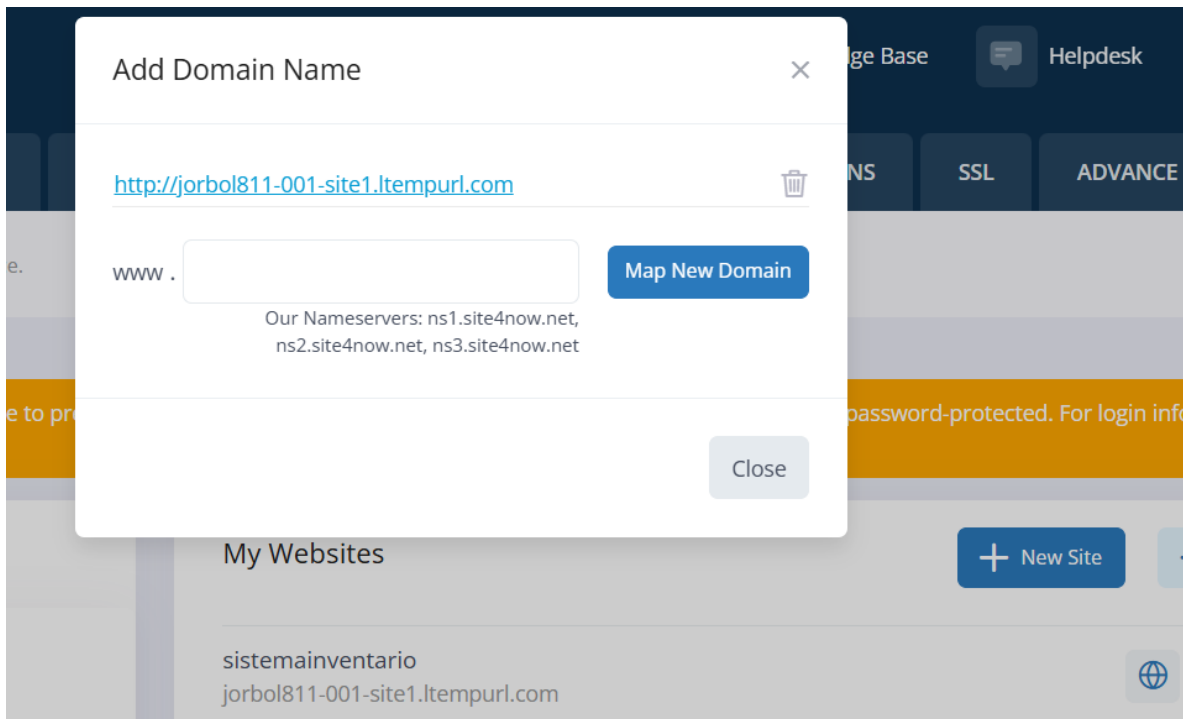
```
"Data Source=SQL8011.site4now.net;Initial Catalog=db_ac0678_sistemainventario;User Id=db_ac0678_sistemainventar;
```

Classic ASP

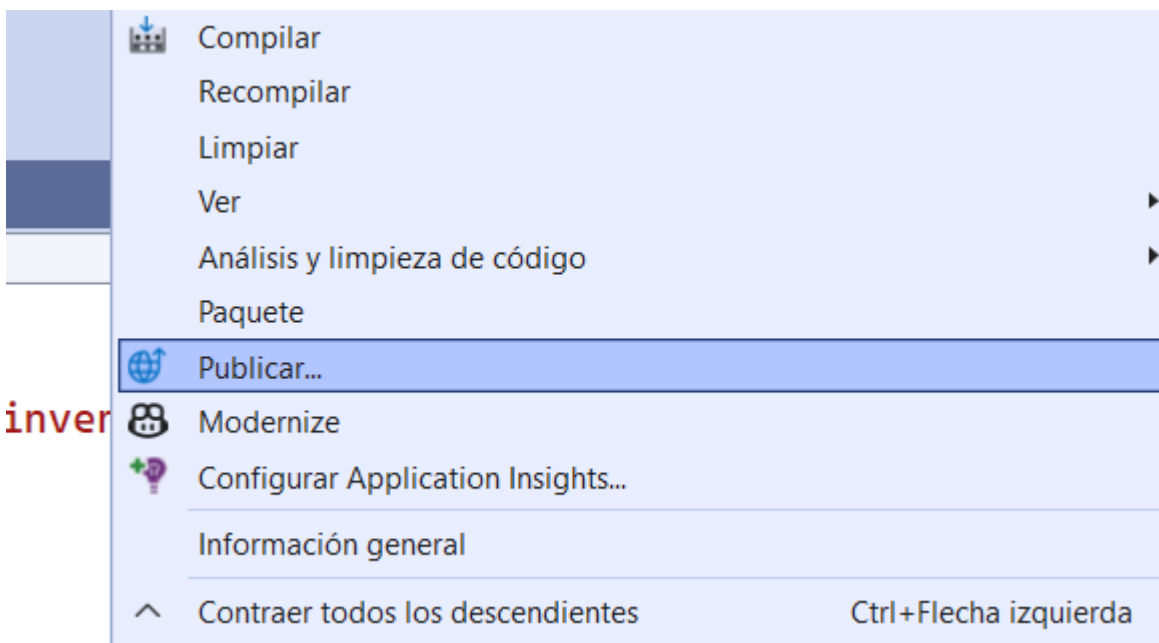
```
"Provider=SQLOLEDB;Data Source=SQL8011.site4now.net;Initial Catalog=;User Id=db_ac0678_sistemainventario_admin;f
```

- 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

Publicar

¿Dónde publica hoy?

Destino

**Azure**
Aloje su aplicación en la nube de Microsoft

**Container Registry para Docker**
Publique la aplicación en cualquier instancia compatible de Container Registry que funcione con imágenes de Docker.

**Carpeta**
Publique su aplicación en un recurso compartido de archivos o una carpeta local.

**Servidor FTP/FTPS**
Publique la aplicación en un servidor FTP/FTPS.

**Servidor web (IIS)**
Publica la aplicación en IIS mediante Web Deploy o un paquete de Web Deploy.

**Importar perfil**
Importe la configuración de publicación para implementar la aplicación.

Atrás

Siguiente

Finalizar

Cancelar

Publicar

Importar archivo de configuración de publicación

Destino

Archivo de configuración de publicación

Importar perfil

<Obligatorio>

Examinar...

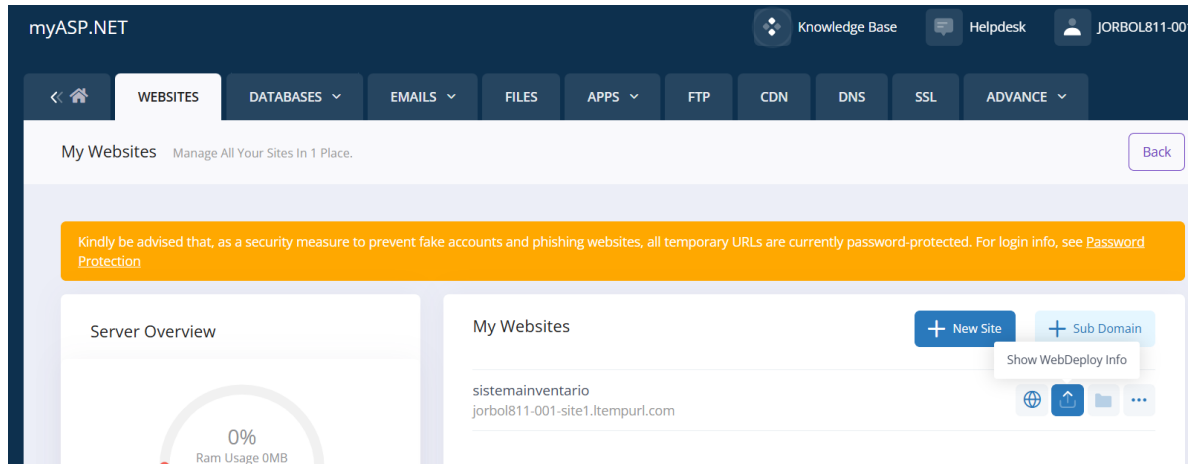
Atrás

Siguiente

Finalizar

Cancelar

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



VS Webdeploy



STATUS	ON TURN OFF
Service URL	https://win8114.site4now.net:8172/MsDeploy.axd?site=jorbol811-001-site1
Site Name	jorbol811-001-site1
Username	jorbol811-001
Password	Modify
Publishing XML	Get Publish Setting
	Get Publish Setting XML
[KB Article]	How to publish ASP.NET site?
Fix ACL	Fix ACL

Publicar

Importar archivo de configuración de publicación

Destino	Archivo de configuración de publicación	
Importar perfil	C:\Users\User\Downloads\site1.PublishSettings	Examinar...

- Damos Click en Mostrar todas las configuraciones



Publicar

Conexión

Configuración

jorbol811-001-site1 - Web Deploy *

Configuración:	Release
Marco de trabajo de destino:	net8.0
Modo de implementación:	Dependiente de marco de trabajo
	Obtenga información acerca de los modos de implementación.
Runtime de destino:	Portable



Opciones de publicación de archivos

☒ Quitar archivos adicionales en el destino



Bases de datos



Migraciones de Entity Framework

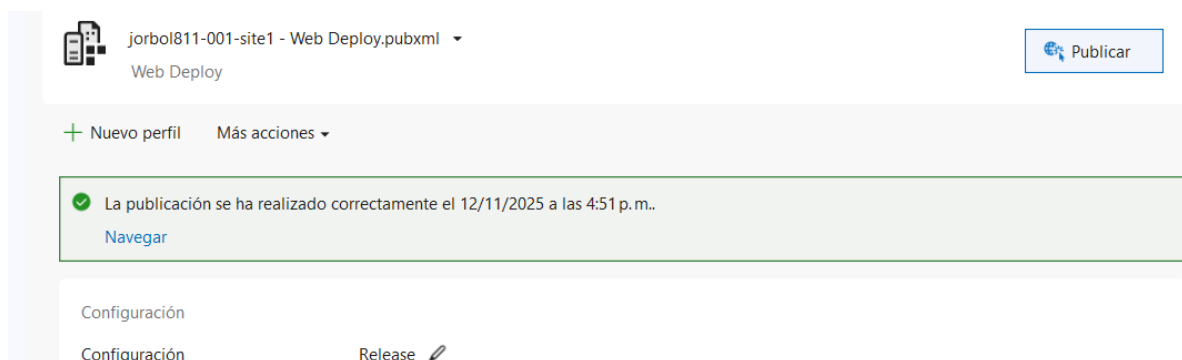
- Presionamos el botón publicar.



The screenshot shows the Visual Studio Web Deploy interface. At the top, there is a header with a server icon, the text "jorbol811-001-site1 - Web Deploy.pubxml", and a "Publicar" button. Below the header, there is a section with a "+ Nuevo perfil" link and a "Más acciones" dropdown. A blue notification bar states "Listo para publicar." Below this, a "Configuración" section displays a table of deployment settings.

Configuración	
Configuración	Release 
Plataforma de destino	net8.0 
Modo de implementación	Dependiente del marco 
Tiempo de ejecución de destino	Portable 

- Ingresamos la contraseña del sitio y comienza el despliegue.



The screenshot shows the Visual Studio Web Deploy interface after a successful deployment. The "Publicar" button is still present. Below the header, the same "+ Nuevo perfil" and "Más acciones" section is visible. A green notification bar with a checkmark icon states: "La publicación se ha realizado correctamente el 12/11/2025 a las 4:51 p.m.." with a "Navegar" link. Below this, the "Configuración" section is partially visible, showing the "Configuración" row and the "Release" setting with an edit icon.

- Verificamos la base de datos en el SQL Server Management