Corporación Brothersoft 2020

Instalación y configuración

Contenido

[1. Requisitos el servidor 2](#_Toc53431998)

[2. Compilar la aplicación y publicarla 3](#_Toc53431999)

[3. Configuración de la aplicación Web 4](#_Toc53432000)

[4. Configuración del WebApi 6](#_Toc53432001)

[5. Generar un sitio en el IIS 7](#_Toc53432002)

[6. Visor de eventos 7](#_Toc53432003)

# Requisitos el servidor

* Para poder publicar nuestra aplicación debemos tener un servidor Windows Server 2012 o de preferencia superior.
* Debemos instalar IIS para poder desplegar nuestra aplicación.
* Debemos instalar un certificado de seguridad para tener https.
* Como estamos usando la última versión de .Net que es ASP Core 3.1 demos descargar los siguientes paquetes:

ASP .Net Core 3.1 SDK

<https://dotnet.microsoft.com/download/dotnet-core/thank-you/sdk-3.1.302-windows-x64-installer>

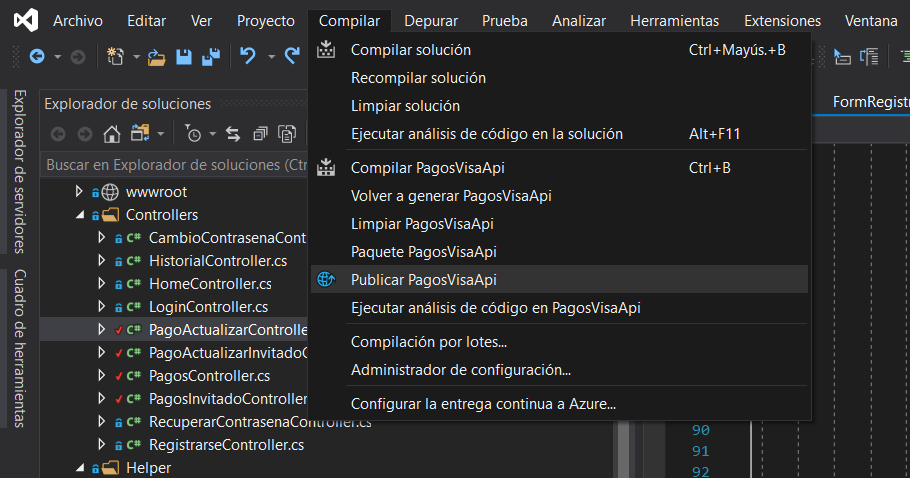
ASP .Net Core Hosting Bundle

<https://dotnet.microsoft.com/download/dotnet-core/thank-you/runtime-aspnetcore-3.1.6-windows-hosting-bundle-installer>

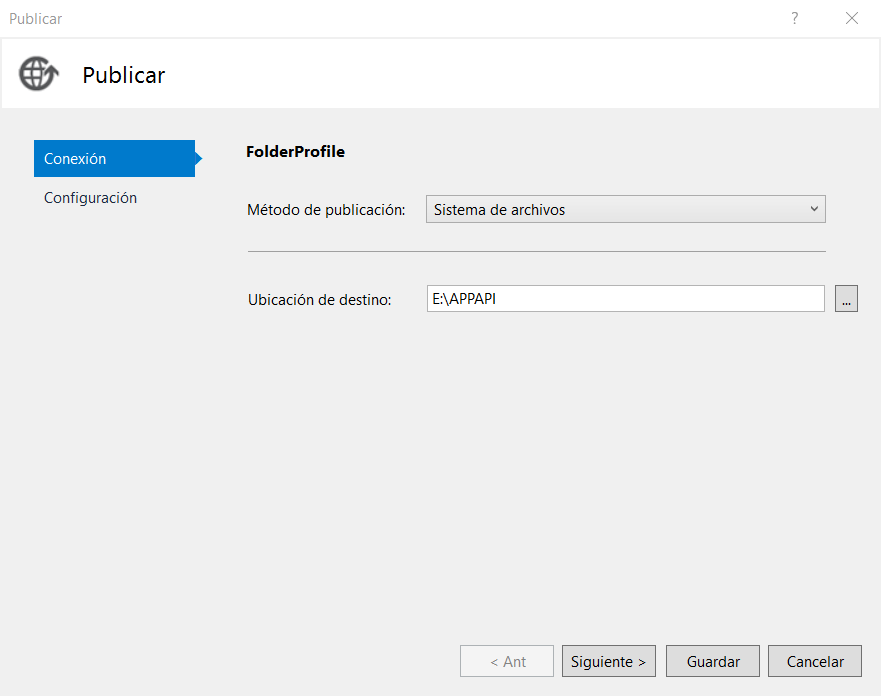
Al ser una aplicación ASP Core también puede funcionar en un servidor Linux.

# Compilar la aplicación y publicarla

* Abrimos el Visual Studio 2019 y cargamos la aplicación Web o Web Api ambas se despliegan de la misma forma.
* Seleccionamos la opción en el menú “Compilar -> Publicar nombre de la aplicación”:

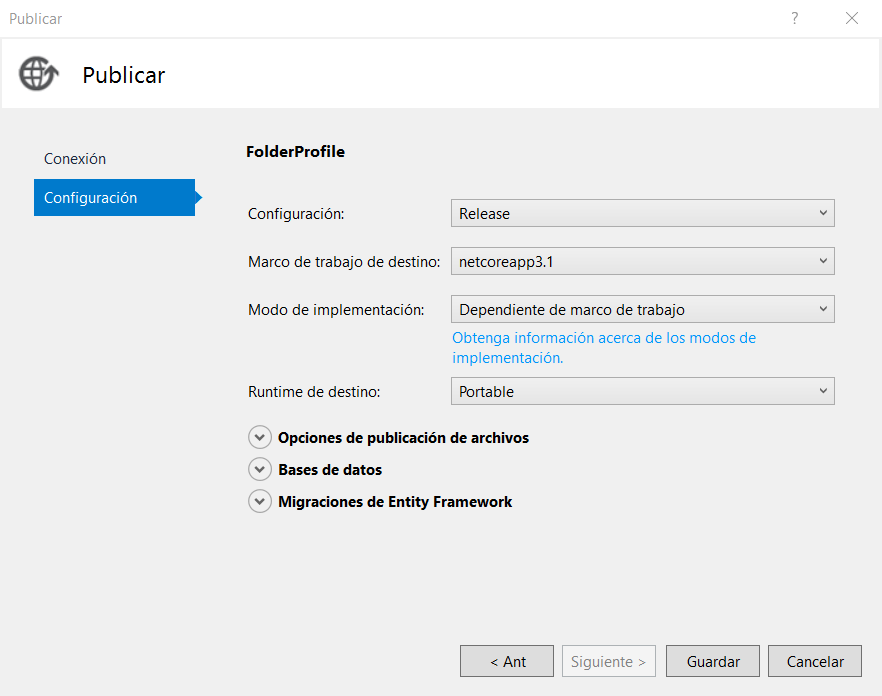


* Nos envía a la ventana de publicación, hacemos clic en la opción nuevo y nos muestra la siguiente ventana si seleccionamos el método de publicación “sistema de archivos” y escogemos la ubicación.



Hacemos clic en siguiente y dejamos las opciones por defecto:

* Configuración: Release.
* Framework: netcoreapp3.1
* Runtime: Portable



# Configuración de la aplicación Web

* En cada aplicación hay un archivo appsettings.json que funciona como el web.config; este archivo podemos configurar la cadena de conexión, parámetros de visa y otros valores
* El archivo appsettings.json :

|  |
| --- |
| {  "Logging": {  "LogLevel": {  "Default": "Information",  "Microsoft": "Warning",  "Microsoft.Hosting.Lifetime": "Information"  }  },  "AllowedHosts": "\*",  "ConnectionStrings": {  "ElectrosurDB": "Server=ELSX007;Database=dbPagosVISA;user id=UsuarioPagosVISA;password=PagosVIS@;"  },  "Datos": {  "Estado": "Desarrollo",  "IpServidor": "209.45.40.148"  },  "VisaProduccion": {  "MerchantId": "",  "AccessKeyId": "",  "SecretAccessKey": "",  "VisaUrlSeguridad": "https://apiprod.vnforapps.com/api.security/v1/security",  "VisaUrlSession": "https://apiprod.vnforapps.com/api.ecommerce/v2/ecommerce/token/session/",  "VisaUrlAuthorization": "https://apiprod.vnforapps.com/api.authorization/v3/authorization/ecommerce/"  },  "VisaDesarrollo": {  "MerchantId": "522591303",  "AccessKeyId": "integraciones.visanet@necomplus.com",  "SecretAccessKey": "d5e7nk$M",  "VisaUrlSeguridad": "https://apitestenv.vnforapps.com/api.security/v1/security",  "VisaUrlSession": "https://apitestenv.vnforapps.com/api.ecommerce/v2/ecommerce/token/session/522591303",  "VisaUrlAuthorization": "https://apitestenv.vnforapps.com/api.authorization/v3/authorization/ecommerce/522591303"  },  "WebServices": {  "AgenciaEntidadConsulta": "VISANET",  "MedioPagoEntidadConsulta": "Ventanilla",  "UsuarioConsulta": "visanet",  "ClaveUsuario": "Cas@2020"  }  } |

Parámetros para modificar:

* ConnectionStrings: Cadena de conexión para nuestra base de datos.
* "Datos": {

"Estado": "Desarrollo",

"IpServidor": "209.45.40.148"

},

Estado debe pasar a “Produccion” y IpServidor es la IP pública del servidor donde está la aplicación.

* VisaProduccion y VisaDesarrollo agregamos las credenciales de Visa.
* WebServices Credenciales para usar con el webservice del Sielse.

Configuracion del web services vamos al siguiente archivo después de publicar la aplicación web:

“9.Codigo fuente\Publicado\web\Connected Services\WSSielse\ ConnectedService.json”

Encontramos la siguiente seccion:

|  |
| --- |
| "ExtendedData": {  "inputs": [  "https://elsx007.electrosur.com.pe/WSCobranzaOnline/SCobranza.svc?Wsdl"  ], |

Donde cambiamos el url de Wsdl.

# Configuración del WebApi

Se realizan las mismas configuraciones anteriores, pero demos cambiar el siguiente texto en el appsettings.json:

|  |
| --- |
| {  "Jwt": {  "Key": "BGy+7Eb7cyqgZy?N6Sw7Z!&dC95Utjk#xVp8yTxWNUMqd%f\*4T$kMuk!jAJMyUVg+R=+3EkL374$Q4vJ=&2$HRV&fu3sZxFZ6S?M+SZK\*bM\*Nq^KpphsguWczRR\*MjcHCbRkQ$V#BDKykQW=e%H=L&5FYf%$3jbK^gGrWvXnYkCM5duj!A6A#EhUjp\*aQWnP2NYRBVDs8xNYNwmJ+8jmwucG\*qc$fZdk3VGKR@\_9x@xK7BAK$#aSq9\_zMqFTeP=cE$VqrqMks4DBzgtbCTQVAVEeg%wRYfg9qY=C$ge8v!Dr\*ZfRbEsNv?pvrjCnK+jqgZXYpXgnettxpsq3\_!-2KFRdGuKWu@6&Szm&2c\*2\*\*^R!mzwtXWUg#Ce%DW\*=G7^#n?A+3#26BPCgvHSfuk$-#kSGVXh4#!a\_VWts&Je$mDBtR+8f@4C\*@f=XfMRJBkys^buWNS8w=MbRRKzZAU%R%Qy+LH5$%y6m%-am7vv!D47KmRe@G@UKF=g\*\_xpv3\*Q",  "Issuer": "Electrosur"  },  "Logging": {  "LogLevel": {  "Default": "Information",  "Microsoft": "Warning",  "Microsoft.Hosting.Lifetime": "Information"  }  },  "AllowedHosts": "\*",  "ClaveSecreta": "zxczc2vz3v215fg17",  "ClaveSecretaMovil": "zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17",  "ConnectionStrings": {  "ElectrosurDB": "Server=ELSX007;Database=dbPagosVISA;user id=UsuarioPagosVISA;password=PagosVIS@;"  },  "WebServices": {  "AgenciaEntidadConsulta": "VISANET",  "MedioPagoEntidadConsulta": "Ventanilla",  "UsuarioConsulta": "visanet",  "ClaveUsuario": "Cas@2020"  }  } |

* Jwt:Key : esta es unca clave q se utiliza para generar la sesión entre el aplicativo móvil y el web api puede cambiarla.
* ClaveSecreta: clave secreta del aplicativo nos sirve apara encriptar y desencriptar la data que se envía al servidor.
* ClaveSecretaMovil: esta es una clave en común que hay entre el servidor y el móvil, ambas aplicaciones deben tener la misma clave.
* Al realizarse el pago en el móvil no configuramos visa.

# Generar un sitio en el IIS

* Generamos un sitio por aplicación y copiamos lo generado por el Visual Studio.
* Si hay una aplicación dentro de otra cada una debe tener su propio hilo de ejecución (Pool V4.0).
* Le asignamos el certificado digital y ya estaría la aplicación.
* **La aplicación web debe colocarse en la raíz de la aplicación porque el direccionamiento de las imágenes y link están gestionados de la manera.**
* La aplicación web api puede ser gestionado con un dominio alternativo o también puede estar dentro de la aplicación principal como otra aplicación, cada aplicación debe tener su propio Pool.

# Visor de eventos

* Para hacer seguimiento de la aplicación podemos utilizar el visualizador de eventos
* Vamos a Windows Logs- > Aplicación.

