Entersoft WEB API

# Getting started

This document assumes that you have a fair knowledge on Entersoft Business Suite (EBS), Entersoft Expert (EXP) or Entersoft CRM (CRM). In order to proceed is strongly recommended to have attend the EBS Foundation Seminar so that you have the knowledge required to setup and connect to EBS, design a Scroller, create and execute an automation, and have a decent knowledge on the EBS Object model.

The Architecture behind the Entersoft WEB API is as following:

1. Entersoft Application Server(s) - **EAS**
2. SQL Server RDBMS - **SQLDB**
3. Entersoft WEB API Server (ASP.NET MVC 5 Web Server) - **EWAPIS**
4. Entersoft API Clients **EAPIC**

## Entersoft Application Server(s)

Entersoft WEB API Server can connect to multiple application servers simultaneously either of load-balancing purposes (in case that all registered application servers serve the same entity – database & company) or for multi-tenant purposes where databases and companies belong to different entities. In order for the EWAPIS to connect to a specific EAS, .NET Remoting over TCP/IP is required to be stablished between the EWAPIS and the EAS and that the designated host server (IP or DNS name) and the TCP/IP port (specified in the $/CSConfig/ES00Server.config) should be accessible from the server hosting the EWAPIS:

<configuration>

<system.runtime.remoting>

<application>

<channels>

<channel ref="tcp" useIPAddress="true" BindTo="127.0.0.1" **port="8888"**>

<clientProviders>

<formatter ref="binary"></formatter>

</clientProviders>

<serverProviders>

<formatter ref="binary" typeFilterLevel="Full"/>

</serverProviders>

</channel>

</channels>

</application>

<customErrors mode="off"/>

</system.runtime.remoting>

</configuration>

EAS can be hosted either on-premises or on the cloud such as Windows Azure VM. In any case you should contact your system IT to configure the firewall or any other system management software that controls access to the internal network resources. The process of registering EAS to be accessed by a specific instance of a EWAPIS is described in the EWAPIS setup and configuration chapter.

## Microsoft SQL Server – Database

Entersoft EBS supports all versions of SQL Server from SQL 2005 and later. However, in order to get access and use the Entersoft Public Queries (EPQ) - the new mechanism introduced since EBS version 4.28) for retrieving data from the EBS - you should use SQL Server 2008 R2 and later. SQL Server 2014 is recommended.

Entersoft Public Queries (EPQ) is the evolution of the Scroller in terms of programmatic access to data from EBS through EBS object model. EPQs support for fast execution (minimal overhead compared to scroller execution), paging, Number of records in total and JSON format as return type. EPQs are described in more detail in the corresponding Appendix.

## Entersoft Web API Server

Entersoft WEB API Server is built on ASP.NET MVC 5 technology and is deployed as a Microsoft IIS Application – Web Site.

## Entersoft API Clients

Depending on the programming environment of the client software that needs access to the Entersoft Application Server there are different libraries that can be used. See the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| Environment | Entersoft API Client | Examples | Repository |
| .NET Framework Native Apps on Windows Platforms | Entersoft Bridge Session | Any native .NET application – Windows Forms, WPF, WCF, WF, ASP.NET, etc. DLL or EXE, | <https://www.nuget.org/packages/Entersoft.ERPBridge/> |
| Portable Class Library enabled environments | Entersoft PCL Client Session | Xamarin iOS, Xamarin Android, Xamarin WP, Xamarin Forms, Mono Linux projects | <https://www.nuget.org/packages/Entersoft.Web.Client/> |
| JavaScript, NodeJS, WEB Front-end development | Entersoft WEB API Client | JQuery, AngularJS, NodeJS, JavaScript frameworks, etc. | <https://github.com/entersoftsa/eswebapi> |

This document focus on the JavaScript and the Entersoft WEB API Client framework (EWAPIC). However, most of the document is applicable to the PCL Library environments and the Entersoft PCL Client Session as it is symmetric to the WEB API and requires – assumes the same architecture.

# Entersoft Application Server

In case that you lack of access to a development environment of Entersoft Business Suite (EBS) you may download a demo version of EBS found in <http://www.entersoft.eu/Demos/Entersoft-showcase-download-demos>. The demo installs all the necessary components and subsystems for EBS to work either in all in one PC / Server or in a distributed architecture. A demo database is also a part of the installation as well as a Startup DB. While in demo mode, EBS is fully functional with limitations on the number of records you may create. Should an SQL Server installation is not available you may download the Express Edition of SQL Server 2014 from Microsoft’s Site <http://www.microsoft.com/en-us/download/details.aspx?id=42299>

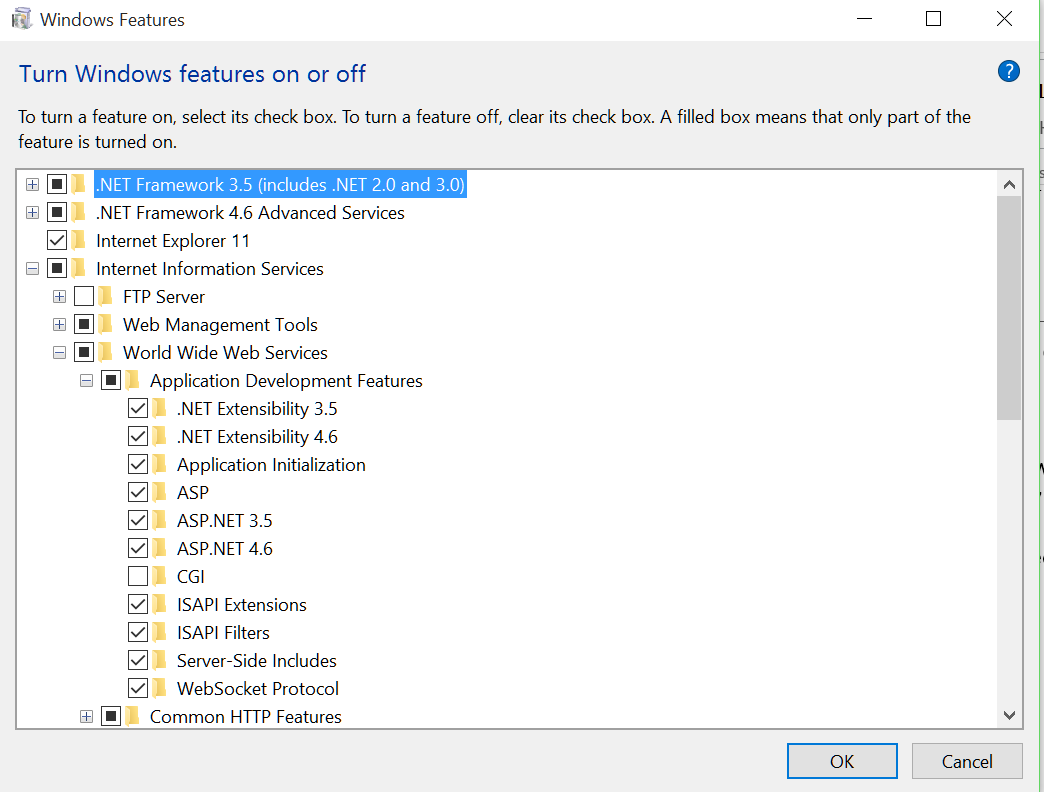
In case you have access to a production environment of EBS, it is strongly recommended to create a development environment either on a side-by-side architecture to your production environment or by installing a local instance of EBS in your development PC.

# Entersoft WEB API Server

The latest version of Entersoft WEB API Server (EWAPIS) can be found on GitHub at: <https://github.com/entersoftsa/eswebapiserver>

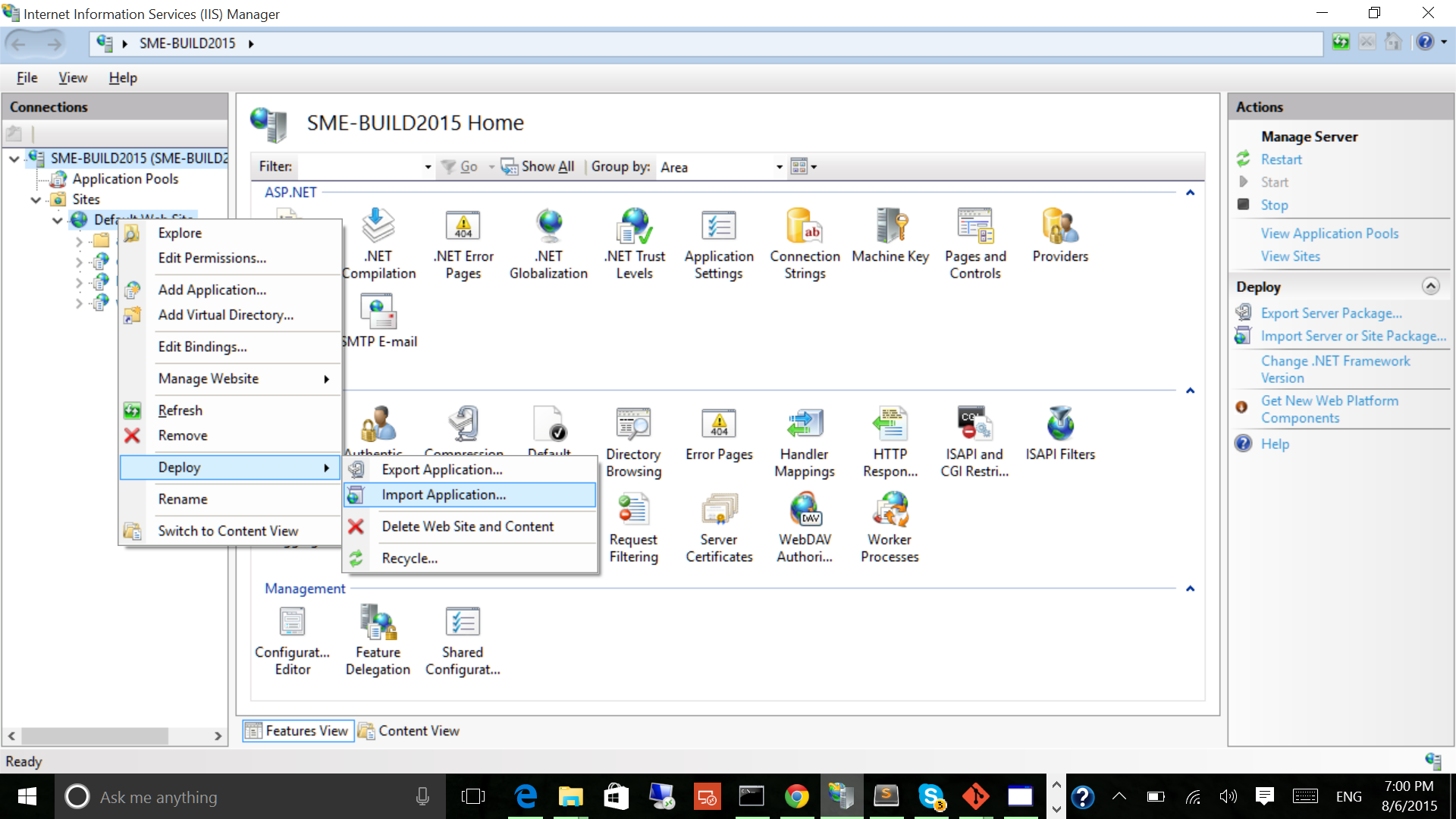
Installation Requirements:

1. Microsoft Windows (Windows 7, Windows 8.1, Windows 10, Windows Servers Family)
2. Microsoft Internet Information Server (MS IIS) 64-bit version 7.5 or higher. It is recommended to use IIS 8.0 or IIS 10.0
3. Prior to start installation please make sure that you have added all the Application Development Features of IIS as shown in the screenshot below.

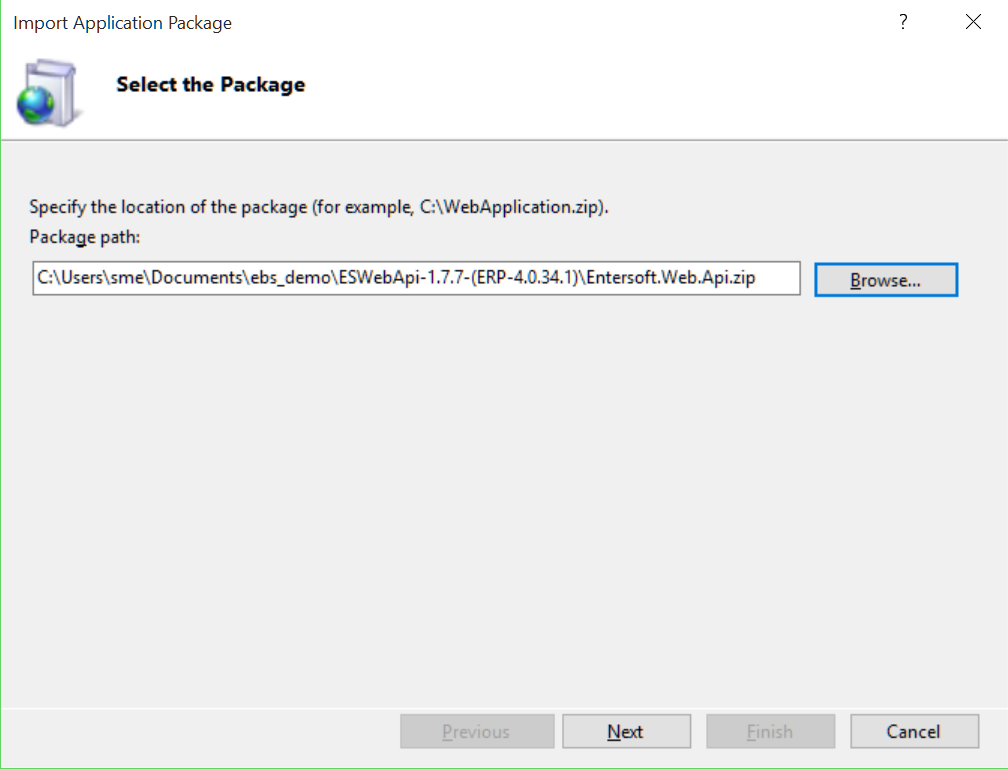


Installation Instructions

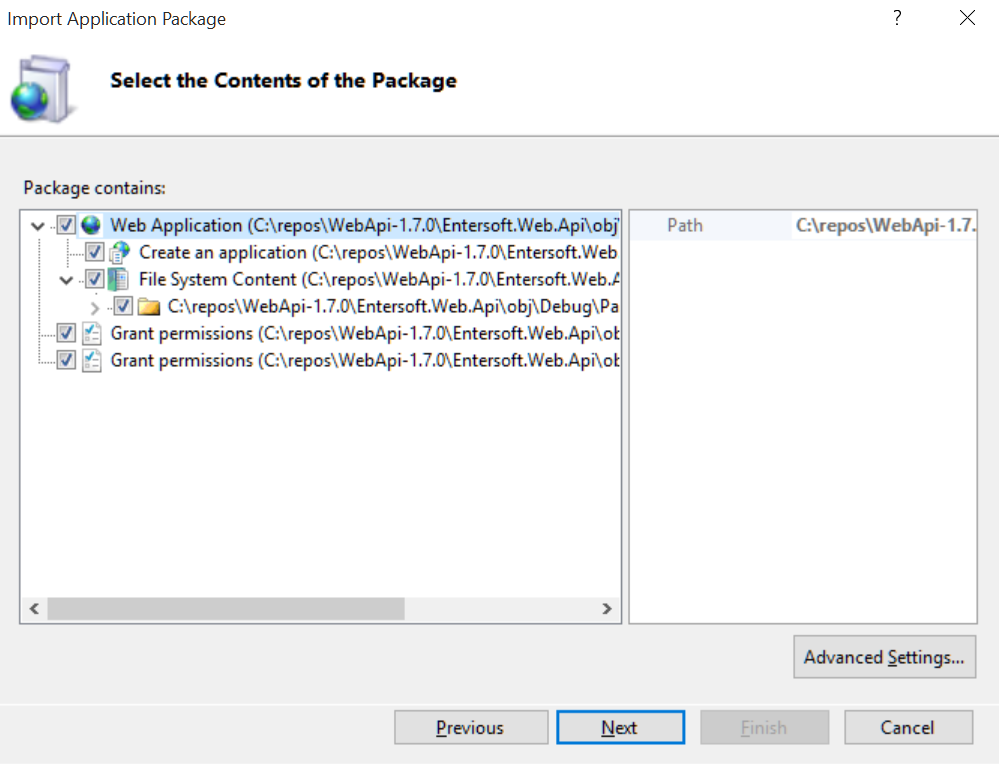
1. Download Entersoft WEB API Server from GitHub: <https://github.com/entersoftsa/eswebapiserver>
2. Open IIS Manager
3. Go to the node “Default Web Site” and choose Deploy -> Import Application



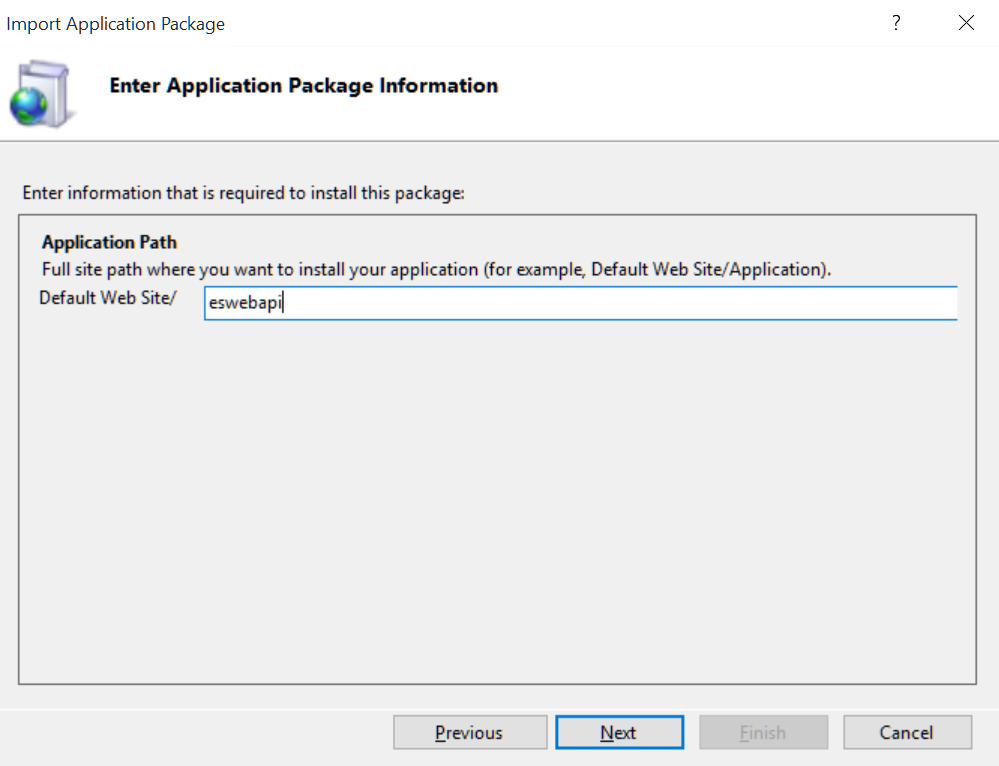
1. Select the zip file that you downloaded from step a)



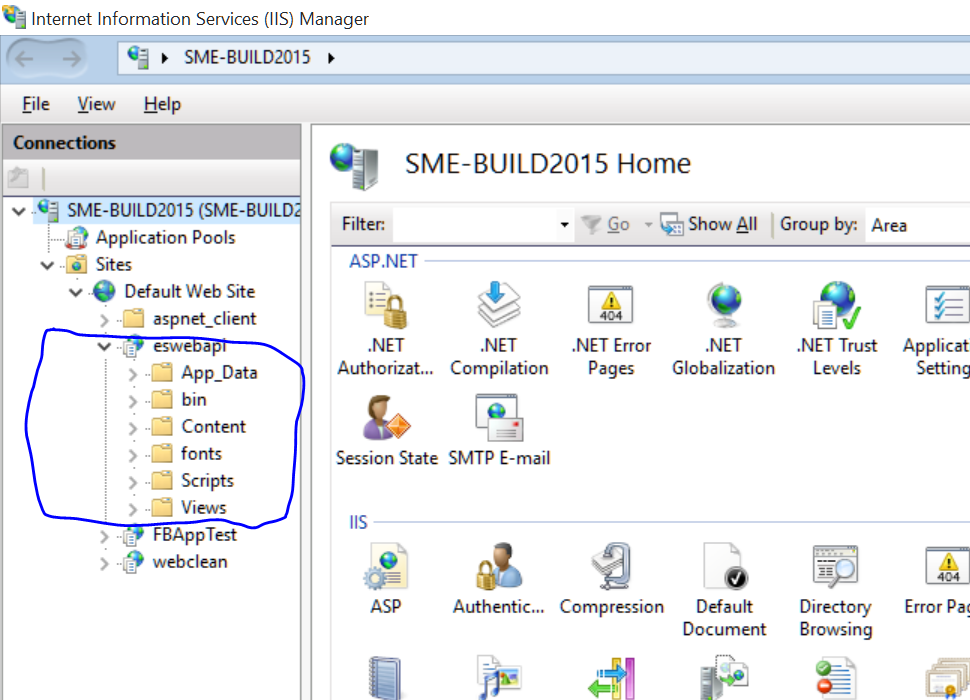
1. Press Next



1. Unless you have the experience and want to change the default settings, which is recommended, Press Next
2. Enter the name of the web site that will expose the Entersoft WEB API service



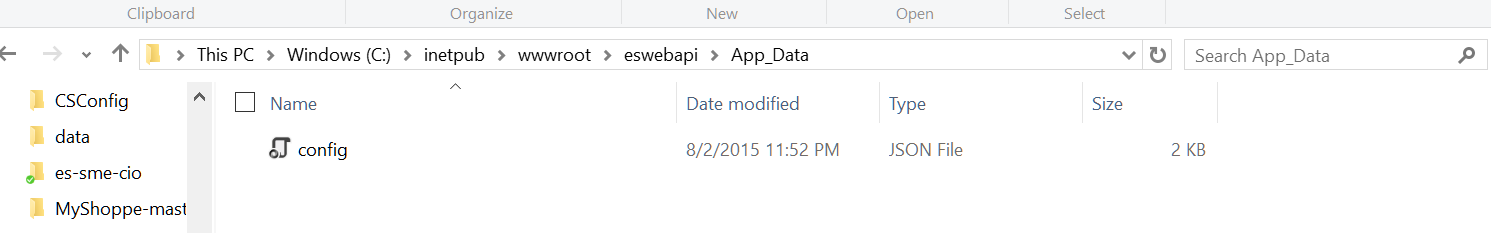
1. Press Next and the installation will begin
2. After successful installation the IIS Manager will have the new application added under the Default Web Site as shown in the picture:



Note: If you are installing the Entersoft WEB API Server on top of an existing application, you will be prompted to specify whether you want a fresh install i.e. remove all existing files first and then install the application, or an update that will leave intact any unnecessary files i.e. not included in the zip being installed.

If it is a fresh install or a fresh re-install, the **directory App\_Data** might not exist and you will create at the next steps. This directory contains all instance related configuration files and settings and thus is never part of the product installation, otherwise your contents will be overwritten on your next update.

1. Once installed, right-click on the *eswebapi* application and choose explore in order to go to the directory where the eswebapi is installed. If the directory App\_Data does not exist, create it with the exact name. Copy the file “**configSample.json**” found in the application’s directory into the **App\_Data** directory and rename it to **config.json**. So the directory structure would look like as in the picture below:



1. Edit the file App\_Data\config.json in order to register the Entersoft Application Server(s) that this WEB API Server will serve. For a single EAS the settings will be as following:

{

"TokenLifetimeInMinutes": 100,

"AdminPassword": "4T0WVxO/s4A=",

"AllowInsecureHttp": true,

"Subscriptions": [

{

"SubscriptionID": "",

"SubscriptionPassword": "CgJHAQYVg+iChKZvXcEKHA==",

"Bridges": [

{

"BridgeID": "",

"DatabaseID": "ES",

"CompanyID": "ES",

"DefaultUserCredentials": {

"UserID": "sme",

"Password": "4T0WVxO/s4A=",

"BranchID": "ΑΘΗ",

"LangID": "el-GR"

},

"Services": [

{

"ServiceID": ""

}

],

"ERPServers": [

{

"Host": "127.0.0.1",

"Port": 8888

}

]

}

]

}

]

}

Where:

TokenLifetimeInMinutes is the number in minutes that the generated token will be valid. After that period of time the token will expire and any subsequent calls using this access token will be denied. It should be greater or equal to 1.

AdminPassword

AllowInsecureHttp if set to true, this instance of the EWAPIS will respond to calls through plain HTTP, otherwise HTTPS (secure HTTP) will be required and no calls (except one that queries the EWAPIS for its capabilities) will be accepted unless they are under https. If in a production environment the use of HTTPS is strongly recommended because sensitive information could be travelling between the client and the server. In order to deploy an HTTPS solution you will need a valid certificate to be installed on the IIS Server, issued by one of the valid and trusted security authorities (VeriSign, GoDaddy, etc.). In case that for any reason you want to bypass the HTTPS requirement, it is strongly recommended to go for alternative solutions in order to secure the communication between the client and the EWAPIS such as VPN, IP security firewalls, etc., assuming that we are referring to intranet applications – clients that are well known to the system and the access can be controlled.

An instance of EWAPIS can serve one or more EAS either of the same Tax Accounting Entity or of different entities (multi-tenant mode). Each EAS should be registered in the config.json file. For this registration we encounter the following concepts / nodes:

* A list of Subscriptions.
* Each Subscription has SubscriptionID which must be unique and a hashed value for a password stored in the SubscriptionPassword field.
* Each Subscription has a list of Bridges that represent a specific connections to an EAS under a unique combination of DatabaseID and CompanyID as specified in the EAS.
* Each Bridge has the following attributes that need to be specified:
* BridgeID a unique identifier for the specific Bridge
* DatabaseID the EBS database id as specified in the ESDBDEF.XML found in the $\CSConfig directory of the EAS. This is equivalent to the DatabaseID a user selects in the EBS ESMainApp.EXE full login dialog.
* CompanyID the EBS Company that this bridge connects to within the companies registered in the DatabaseID. This is equivalent to the CompanyID a user selects in the EBS ESMainApp.EXE full login dialog.
* DefaultUserCredentials: Depending on the user login mode that will be selected at the WEB API client application (ERP Mode, Impersonation Mode, B2C mode) the EWAPIS needs to have registered a valid EBS User so that this user will be used by the EWAPIS either at the Impersonation Mode as well as for any operation that can be executed against the EAS without any prior user login (e.g. FetchUserSites). Typically, this user should be an administrator but for security and audit trail purposes it is strongly recommended this user not be the default admin users found in any EBS installation such as “esmaster”, “admin”. It is strongly recommended the Sys Admin of EBS installation to create a new user e.g. **eswebapi** with a strong password and give this user access to all companies and branches from within the EBS User Management screens (for more information please refer to the EBS Manual). In case that you need more elaborate audit trail from all the EWAPIS calls against a specific DatabaseID / CompanyID defined in a Bridge instance, it is recommended to create different EBS users for each specific BridgeID. DefaultUserCredentials include the following attributes need to be specified as in a typical EBS ESMainApp.EXE full login dialog:

UserID

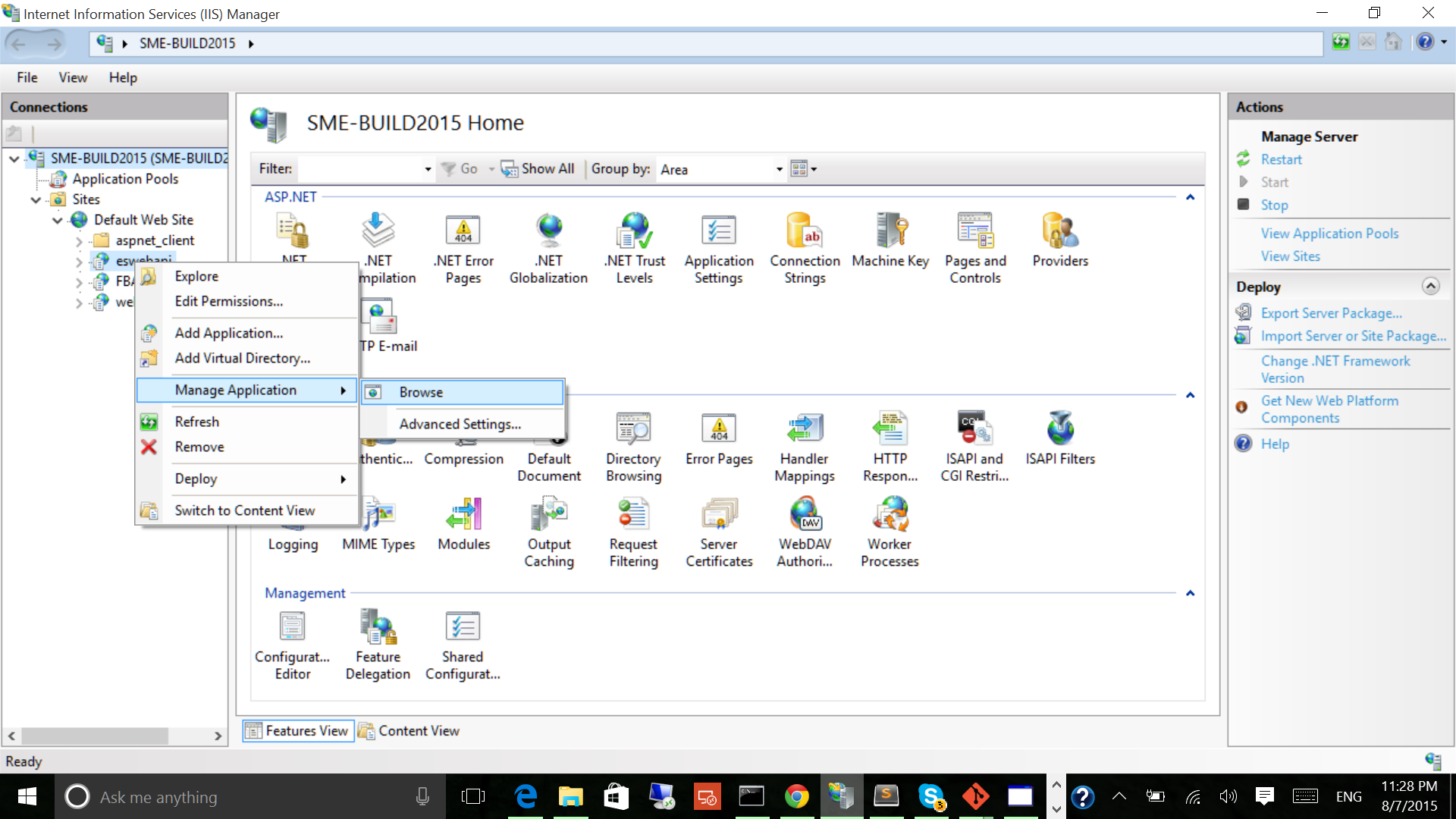
Password this the hashed value stored in the database under the ESGOUser Table in the corresponding record.

BranchID

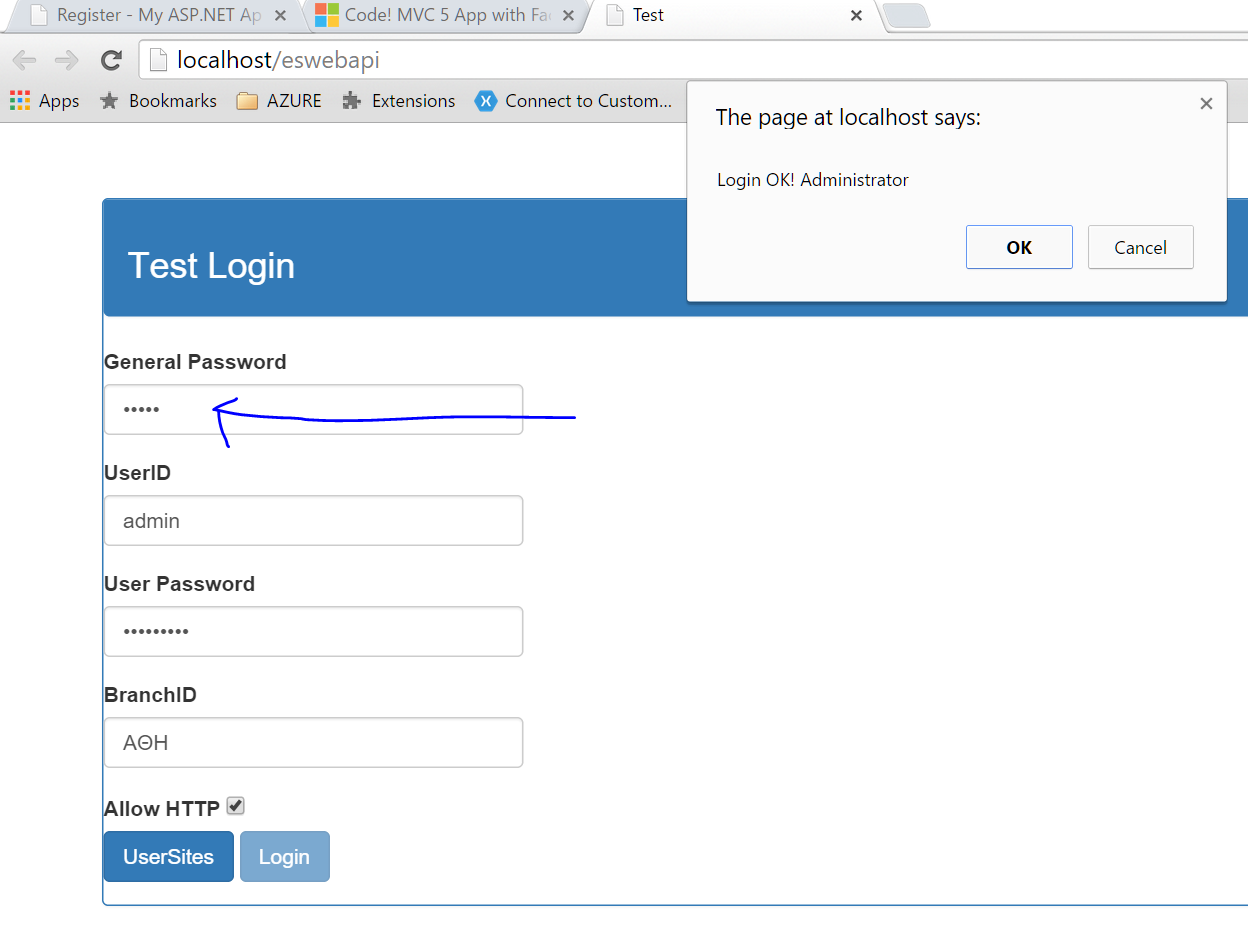
LangID

* Services Each Bridge can support any number of services i.e. Applications (RFA, Expenses, etc). Every WEB API client connects to the EWAPIS in the context of a specific Service. This version of EWAPIS does not fully support Services, so it is recommended to leave this section as in the template.
* ERPServers Each Bridge should have at least one or more EAS servers that should be registered under the ERPServers section. Should more than one ERPServers are declared then these servers are required to fully support the DatabaseID, CompanyID and DefaultCredentials that specified under the Bridge definition and will be used in a Round-Robin form for Load Balancing and Fault Tolerant access to the services.
* HostID the IP or DNS name of the EAS
* Port the port to which the given EAS is listening to

Once config.json is correctly configured you should restart the EWAPI application / site through the IIS Management console and then you can test the connectivity by browsing to the test page:



If installed in the localhost then the test page is accessible by any browser by navigating to <http://localhost/eswebapi> (where eswebapi is the application name used when installing the EWAPIS)



The General Password refers to the SubscriptionPassword which if left unchanged is **passx**

AllowHTTP should be clicked in case that you deployed the EWAPIS in a non-secure server i.e. NOT using HTTPS. If everything is OK then you will see the Alert Message “OK” along with the name of the given user.

# Entersoft WEB API AngularJS Library

**Module: es.Services.Web**

Function

openSession(credentials)

Parameters:

credentials is an object of the following type

{

UserID: string,

Password: string,

BranchID: string,

LangID: string

}

Description

openSession returns a promise that once completed it will return either a valid access token that will be held and stored internally or an error. User login credentials should be given in a JSON object as

Example

$scope.credentials = {

UserID: 'admin',

Password: 'entersoft',

BranchID: 'ΑΘΗ',

LangID: 'el-GR'

};

esMessaging.subscribe("ES\_HTTP\_CORE\_ERR", function(rejection, status) {

var s = esGlobals.getUserMessage(rejection, status);

$scope.esnotify.error(s);

});

$scope.doLogin = function() {

esWebApiService.openSession($scope.credentials)

.then(function(rep) {

$location.path("/pq");

});

}