

Deployment of the Application

This document describes the basics steps needed to be taken when deploying “**SogeYoung**” Application to a host

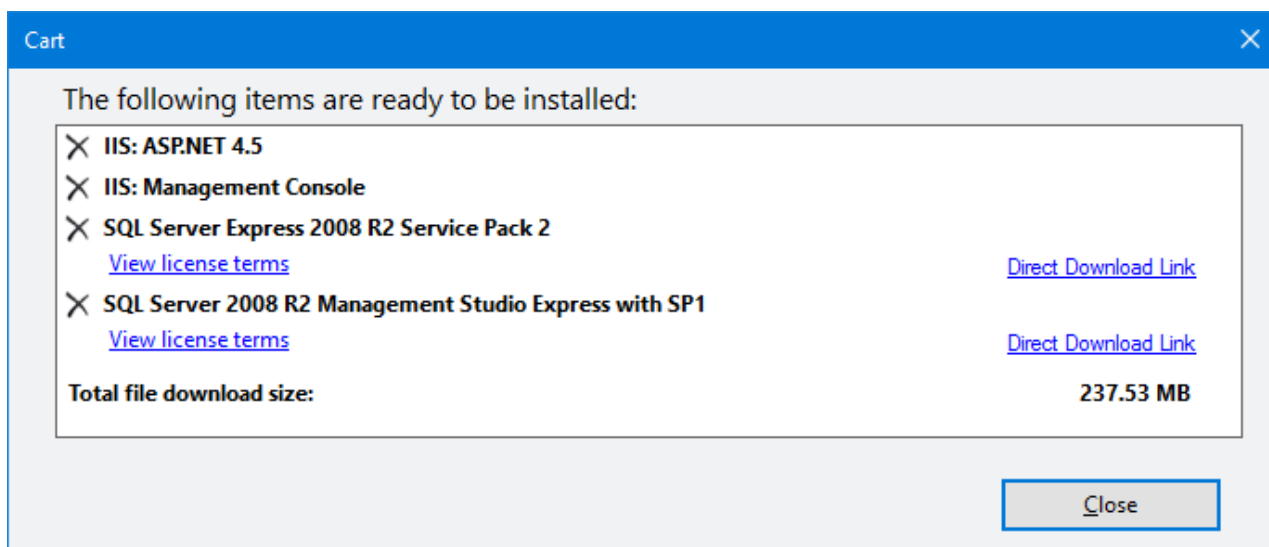
1. Hosting ASP.NET MVC

Building MVC Project produces a **.dll** files. Which need a host process to load them into memory and for websites that host also responsible for delivering HTTP requests to the logic inside that **.dll**. IIS Express is a development host that we use for development so far.

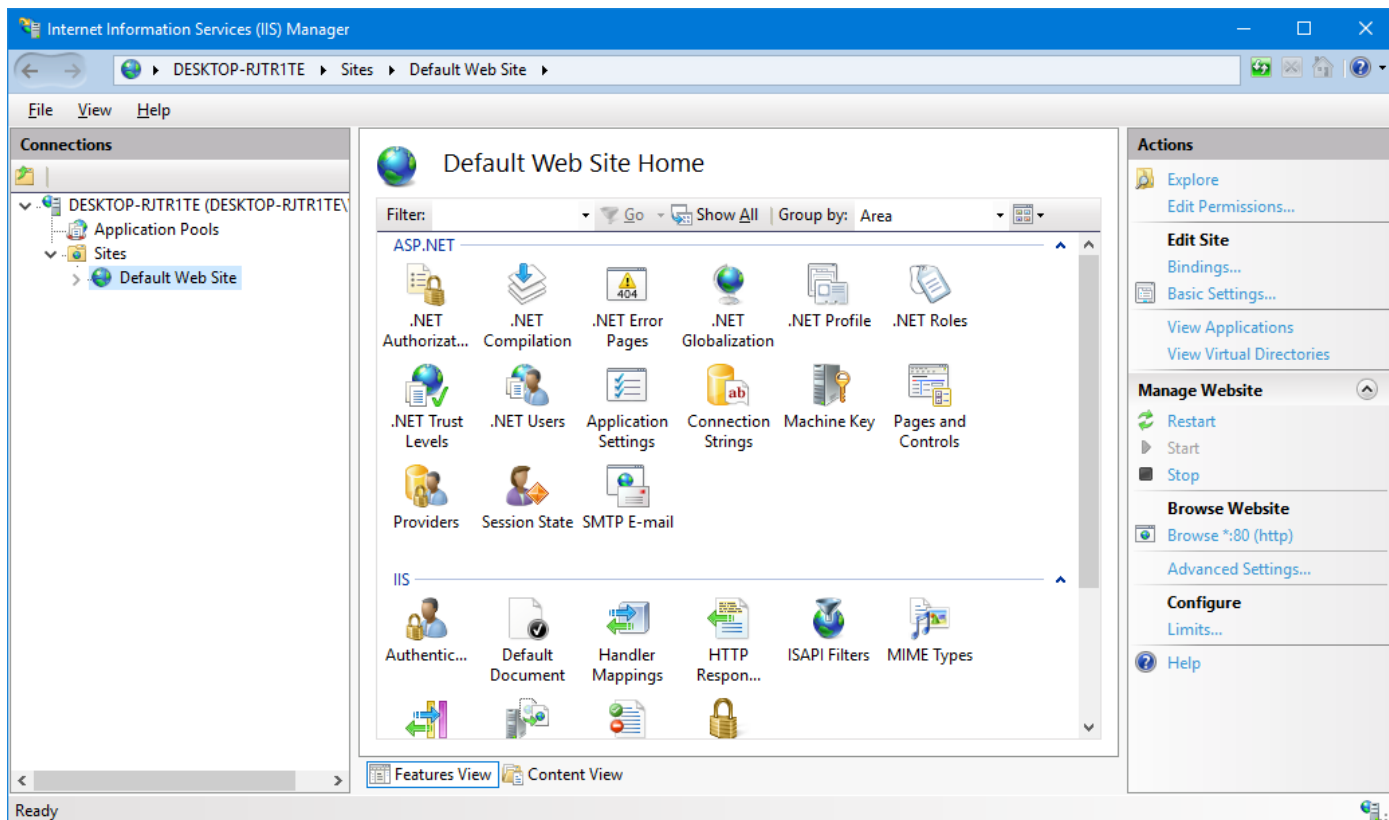
The **IIS Express** makes it very easy for us as developers to develop applications because it runs with our windows identity and can be started or stopped at any time we need to. But if you need to make you application available to the internet you probably would use the **full version** of **Internet Information Services**. IIS can be installed on nearly any version of Windows but by default it is not installed. We could download it and setup our machine as real server machine to **deploy our application**.

To begin you need to download several components. The easiest way to do it is to **download** [Microsoft Web Platform Installer](#) then **install** (if you have not already installed):

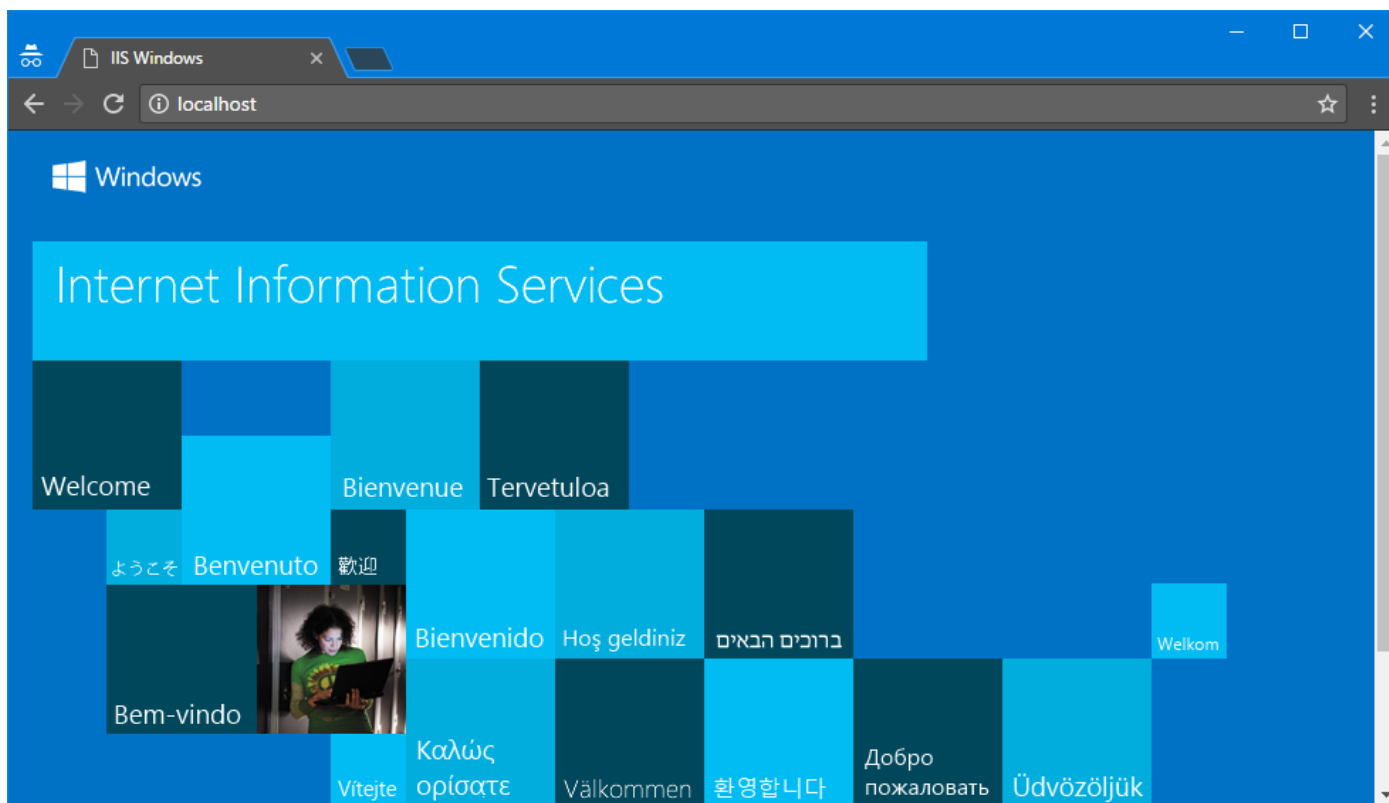
- Visual Studio 2015
- Microsoft .NET Framework 4.6
- IIS ASP.NET 4.5
- IIS: Management Console
- SQL Server Express
- SQL Management Studio



From the **IIS Manager Console**, you can setup the hosting of web applications. By default, there is Default Web Site that is up and running.



To check whether the Default Website is running open a browser and go to **localhost** and you should see the **default welcome page** of the IIS. We will **replace that page** with our application.



2. Preparing for Deployment

Before deploying our project for the first time we should make sure that the **database schema is correct**, **auto migrating is disabled** and **there are scripts for migrating**. Because when we deploy it we will have **live data** and we **do not want to lose it** when we make some changes to the database.

```
internal sealed class Configuration : DbMigrationsConfiguration<
{
    public Configuration()
    {
        AutomaticMigrationsEnabled = false;
    }
}
```

To be sure our migration will go as we expected we would configure it through code. That configuration also can be done in the **Web.config** file.

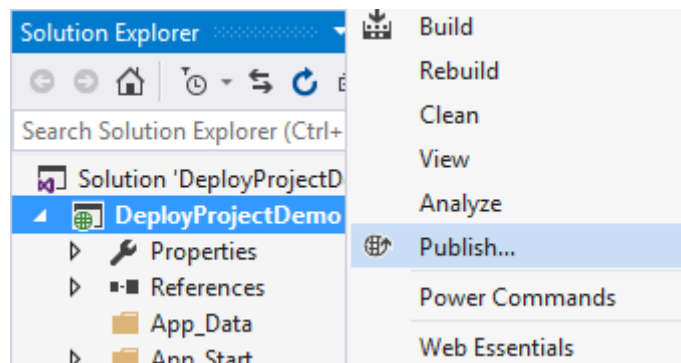
```
public class MvcApplication : HttpApplication
{
    protected void Application_Start()
    {
        var migrator = new DbMigrator(new Configuration());
        migrator.Update();

        AreaRegistration.RegisterAllAreas();
    }
}
```

Also **update the database** from the PM Console with **update-database** command

3. Deploying to IIS

To deploy a project just **right click** on the name of the project and select **Publish...**



Create **new custom publishing profile** and name it the way you want. (for example, release).

Publish Web

Profile

Connection

Settings

Preview

Select a publish target

- Microsoft Azure Web Apps
- Microsoft Azure API Apps (Preview)
- Import
- Custom**

New Custom Profile

Profile name:

release

OK Cancel

< Prev Next > Publish Close

Now we should select the **publish method**. We should also select **destination** where the package will be created. The site **name** would be **Default Web Site**. It can be changed at any time but for simplicity we would leave it as default.

On that screen, you need to select publish method among several of them:

- **Web deploy** – that option can be used to publish your site at your hosting company if it allows it. can be published only as administrator, can be achieved if Visual Studio is started as an administrator.
- **Web deploy package** – create a **.zip** file that can be published to a host
- **FTP** – that method is an option if your hosting provider does not support web deploy
- **File System** - This method publishes the web application to a folder that you specify. You can then use your own FTP tool to transfer the files to a hosting provider.

More information about publishing methods can be found [here](#).

We would create it using the **web deploy package** option and then will **install it manually** on our **local IIS server**.

On the next screen, we set configuration of deploy to **Release**. And should select the **connection to the database**.

IF **Execute Code First migrations** check box is checked the package will configure the migration of the database in the **Web.config** file of the project but since we created it in the code in the **Application_Start()** method we do not need to check it so we leave it unchecked.

Publish Web

Profile

Connection

Settings

Preview

release *

Configuration: Release

File Publish Options

Databases

ApplicationDbContext (DefaultConnection)

Remote connection string

☒ Use this connection string at runtime (update destination web.config)

☐ Execute Code First Migrations (runs on application start)

< Prev Next > Publish Close

We should pick a **connection string to the SQL Server** and that would be to our SQL Server (not the LocalDb). When setting up the connection string you can choose between **SQL Server Authentication** (with username and password) and **Windows Authentication**. For now, we would **select Windows Authentication** method and later we would find **some problem** with it that we would **solve**. Final step in configuring the connection string is to **select name of the database** we would create.

Destination Connection String

Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.

Data source:
Microsoft SQL Server (SqlClient) Change...

Server name:
DESKTOP-RJTR1TE Refresh

Log on to the server

Authentication: Windows Authentication

User name:

Password:

☐ Save my password

Connect to a database

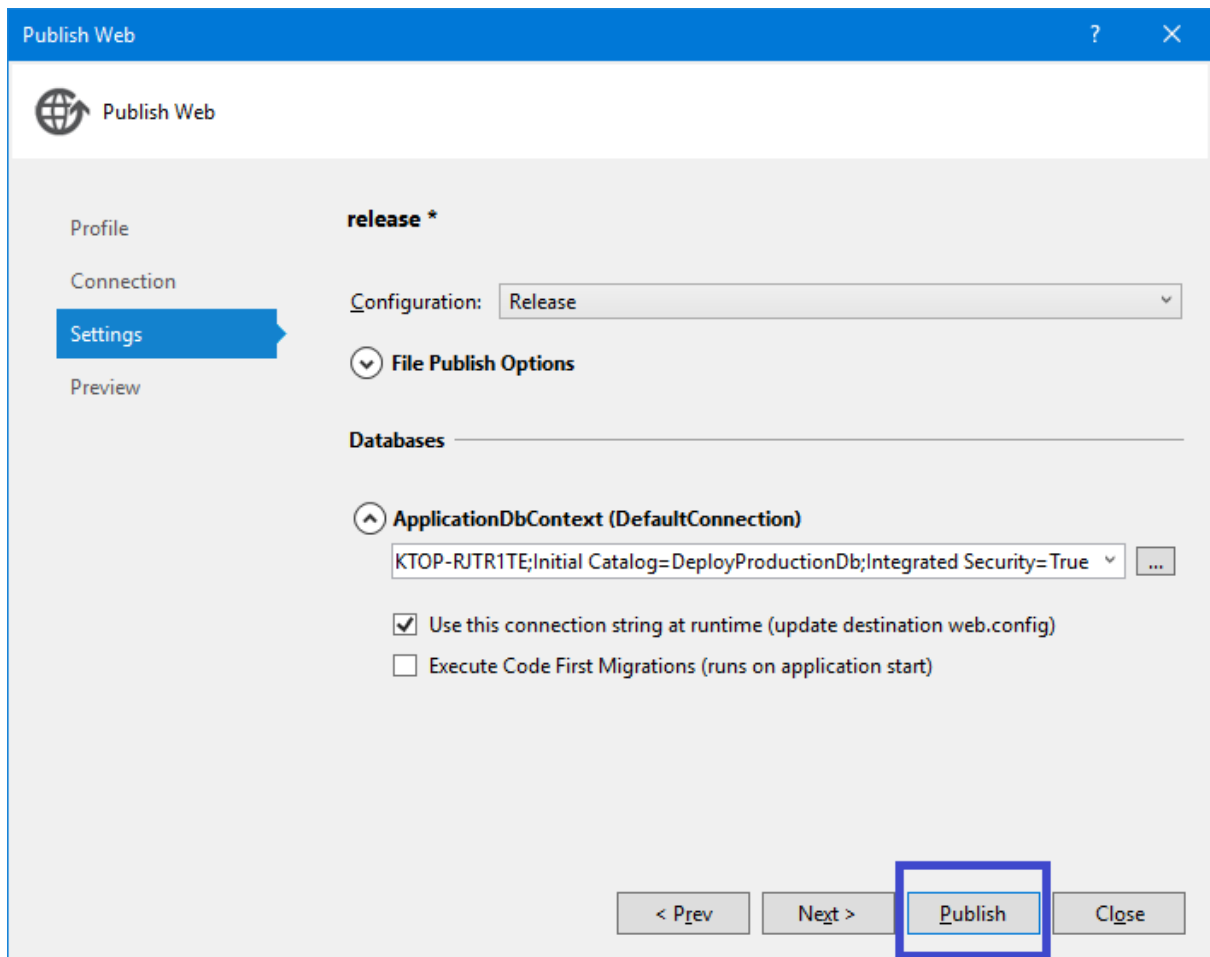
☒ Select or enter a database name:
DeployProductionDb

☐ Attach a database file:
 Browse...
Logical name:

Advanced...

Test Connection OK Cancel

Finally, just click on **Publish button**



Now it is time to **deploy the package to the IIS**. Open the **Command Prompt as Administrator**. Then navigate to the folder where we deployed our package in our case **C:\deploy**. In that folder, there is a file called **release.deploy.cmd**.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd \deploy

C:\deploy>dir
Volume in drive C has no label.
Volume Serial Number is A6EA-DA5B

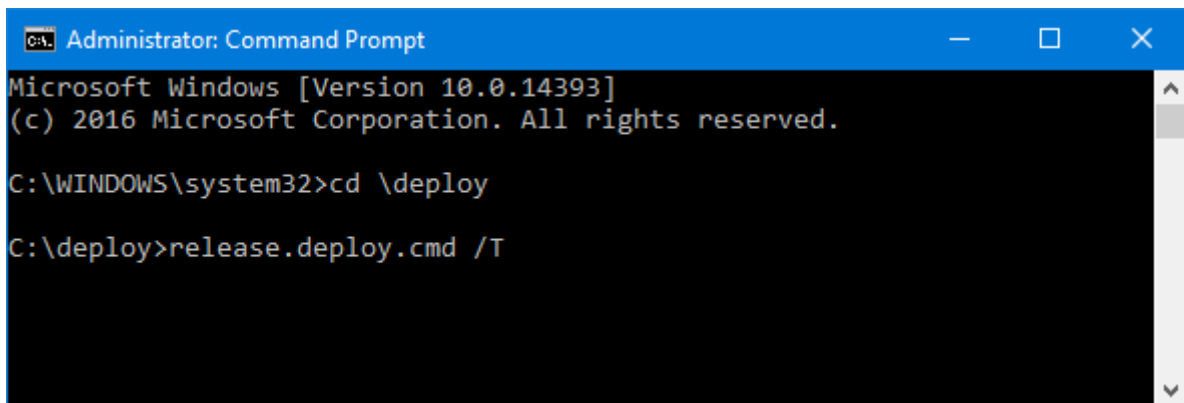
Directory of C:\deploy

26/04 10:32 <DIR>          .
26/04 10:32 <DIR>          ..
26/04 10:32             4,000 release.deploy-readme.txt
26/04 10:32          14,365 release.deploy.cmd
26/04 10:32             317 release.SetParameters.xml
26/04 10:32             510 release.SourceManifest.xml
26/04 10:32      8,273,651 release.zip
               5 File(s)      8,292,843 bytes
               2 Dir(s) 10,697,478,144 bytes free

C:\deploy>
```


When you run it, it opens a help file with instructions how to run the script file in details with all options. But just to be short you need just **2 commands**:

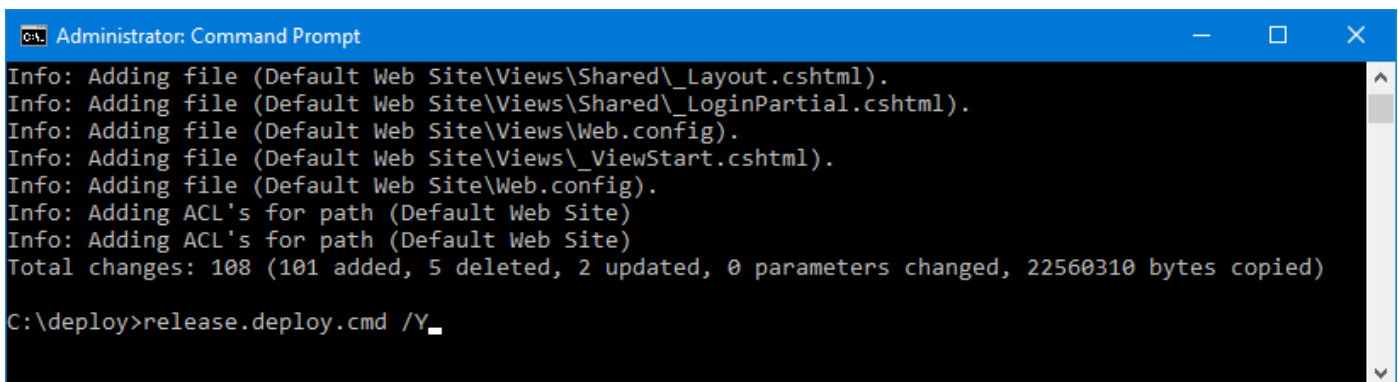
- **release.deploy.cmd /T** – that would **Test the script if can be executed without any problems**. If there are any problems they would appear in red color on the console and you have to fix them so you can deploy the package to the IIS



```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd \deploy
C:\deploy>release.deploy.cmd /T
```

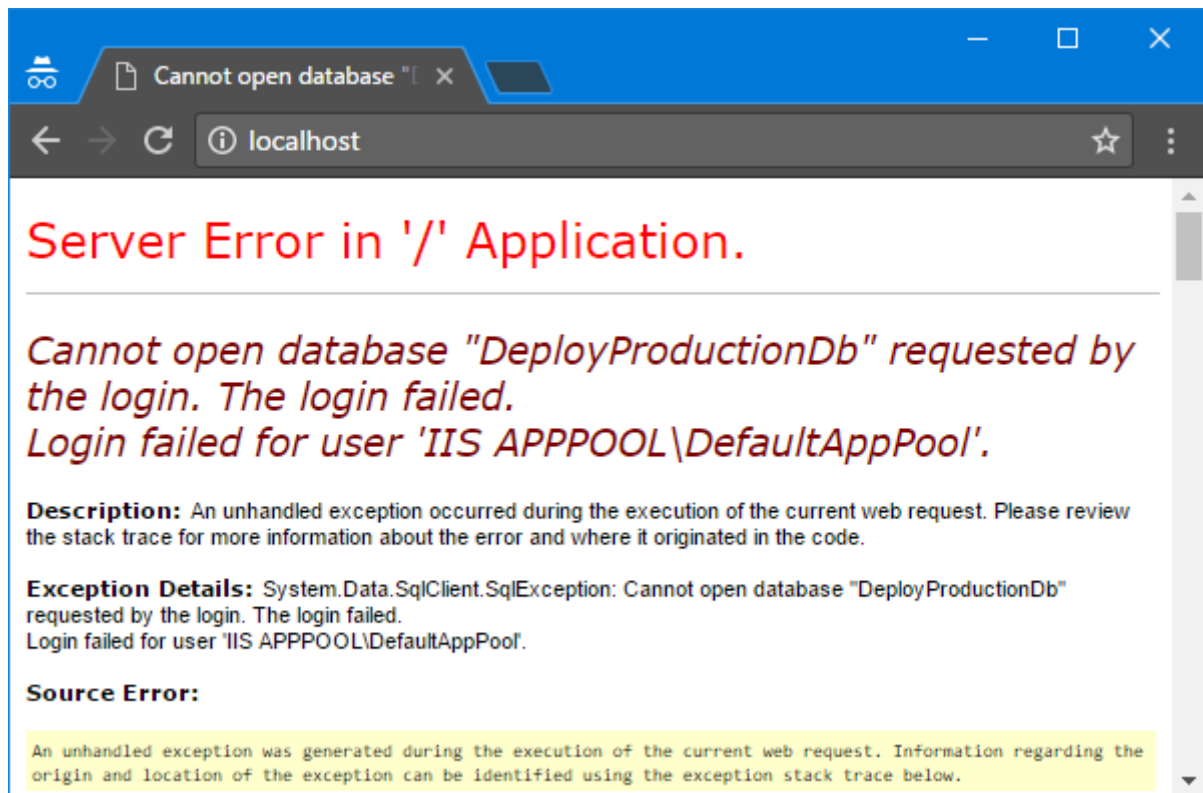
- **release.deploy.cmd /Y** – if there are no errors that command would **deploy the package to our IIS**.



```
Administrator: Command Prompt
Info: Adding file (Default Web Site\Views\Shared\_Layout.cshtml).
Info: Adding file (Default Web Site\Views\Shared\_LoginPartial.cshtml).
Info: Adding file (Default Web Site\Views\Web.config).
Info: Adding file (Default Web Site\Views\_ViewStart.cshtml).
Info: Adding file (Default Web Site\Web.config).
Info: Adding ACL's for path (Default Web Site)
Info: Adding ACL's for path (Default Web Site)
Total changes: 108 (101 added, 5 deleted, 2 updated, 0 parameters changed, 22560310 bytes copied)

C:\deploy>release.deploy.cmd /Y_
```

After the package is loaded let's try to reach the website at localhost.



Since we selected to login with our **Windows Authentication** the IIS tries to login with the user '**IIS APPPOOL\DefaultAppPool**'. But in our SQL Server there is no such user so we need to **create** it and give him some **privileges** for the database of the application:

- **db_datareader** – so that user can read data
- **db_datareader** -so that user can add new data to the database
- **db_ddladmin** – so that user can create new tables and modify the database schema

Login - New

Select a page:

- General
- Server Roles
- User Mapping
- Securables
- Status

Connection

Server: DESKTOP-RJTR1TE

Connection: appUser

[View connection properties](#)

Progress

Ready

Script Help

Login name: IIS APPPOOL\DefaultAppPool Search...

☒ Windows authentication
☐ SQL Server authentication

Password:

Confirm password:

☐ Specify old password

Old password:

☒ Enforce password policy
☒ Enforce password expiration
☒ User must change password at next login

☐ Mapped to certificate
☐ Mapped to asymmetric key
☐ Map to Credential Add

Credential	Provider

Remove

Default database: master
 Default language: <default>

OK Cancel

Login - New

Select a page

- General
- Server Roles
- User Mapping
- Securables
- Status

Script **Help**

Users mapped to this login:

Map	Database	User	Default Schema
<input type="checkbox"/>	AdventureWorks		
<input type="checkbox"/>	AdventureWorksLT		
<input type="checkbox"/>	Adventureworks2008		
<input checked="" type="checkbox"/>	DeployProductionDb	IIS APPPOOL\Default...	
<input type="checkbox"/>	master		
<input type="checkbox"/>	model		
<input type="checkbox"/>	msdb		
<input type="checkbox"/>	tempdb		

☐ Guest account enabled for: DeployProductionDb

Database role membership for: DeployProductionDb

- ☐ db_accessadmin
- ☐ db_backupoperator
- ☒ db_datareader
- ☒ db_datawriter
- ☒ db_ddladmin
- ☐ db_denydatareader
- ☐ db_denydatawriter
- ☐ db_owner
- ☐ db_securityadmin
- ☒ public

Connection

Server: DESKTOP-RJTR1TE

Connection: appUser

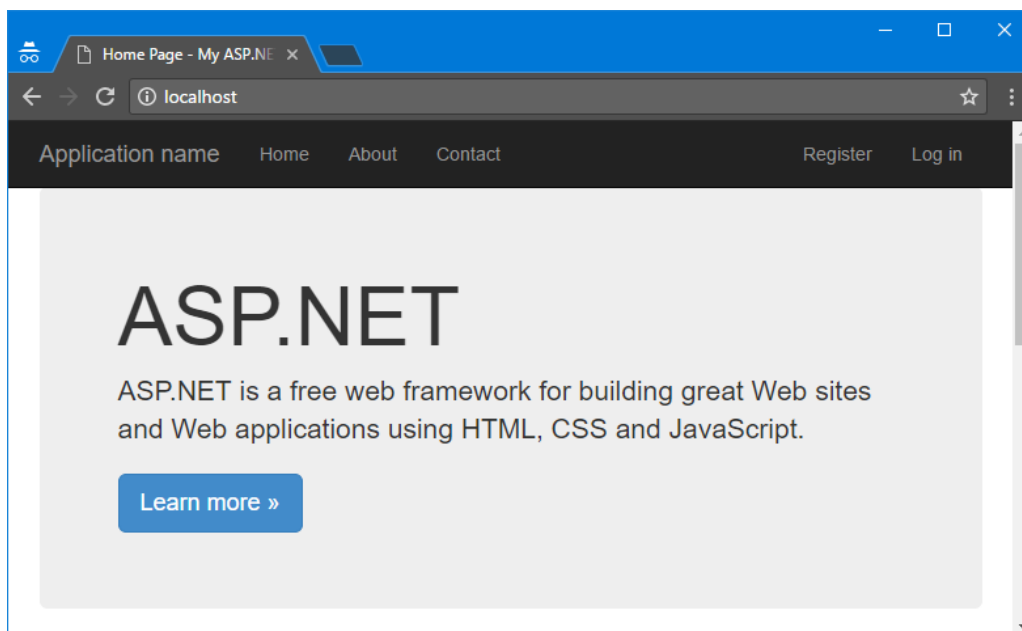
[View connection properties](#)

Progress

Ready

OK **Cancel**

Now everything should be fine and our site should be **loaded correctly** as expected.

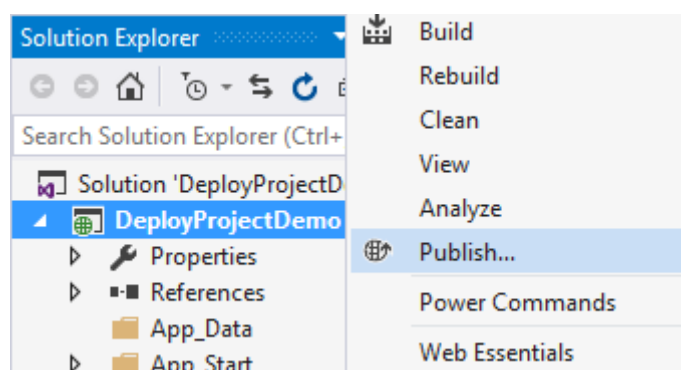


In the beginning if you have **not selected Windows Authentication** but instead a **SQL Authentication** with user that have those privileges **that error might not occur**.

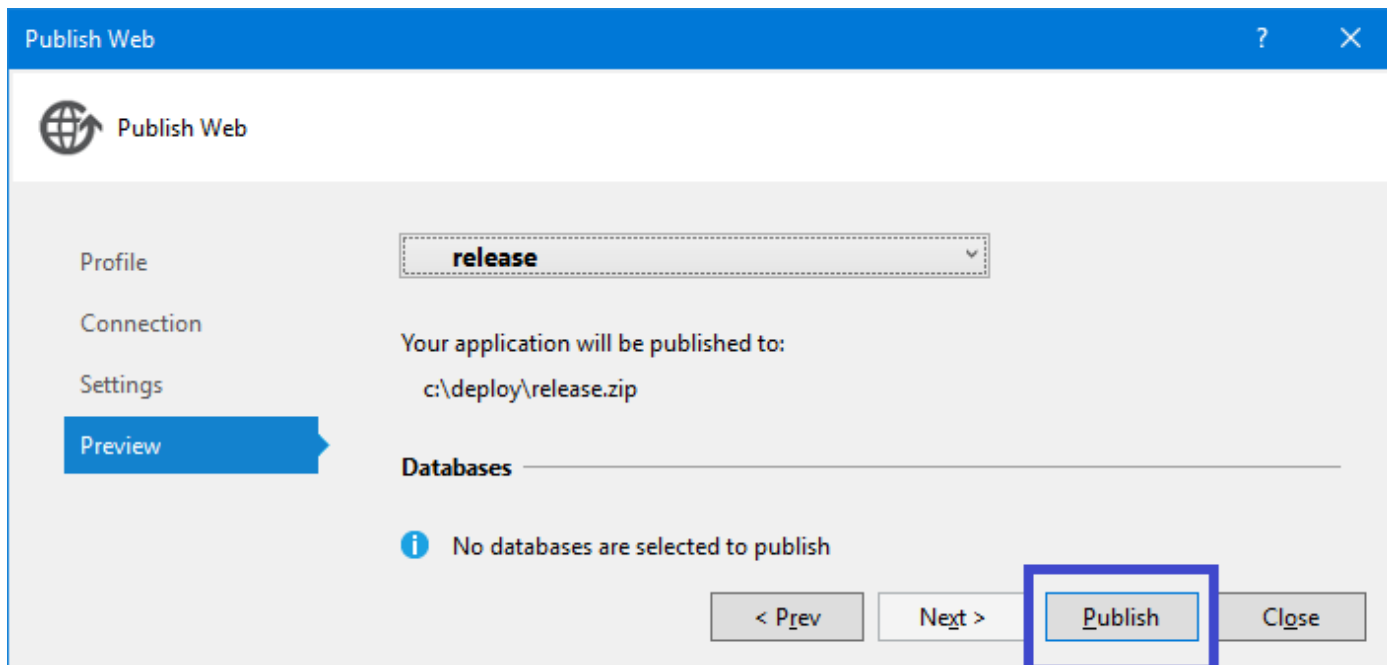
4. Second Deployment

Let's make **some changes in the project and rebuild it**. For example, in our test project I would change the text in the jumbotron section on the home page and rebuild the project.

Now select again the **Publish** option from the **context menu** of the project



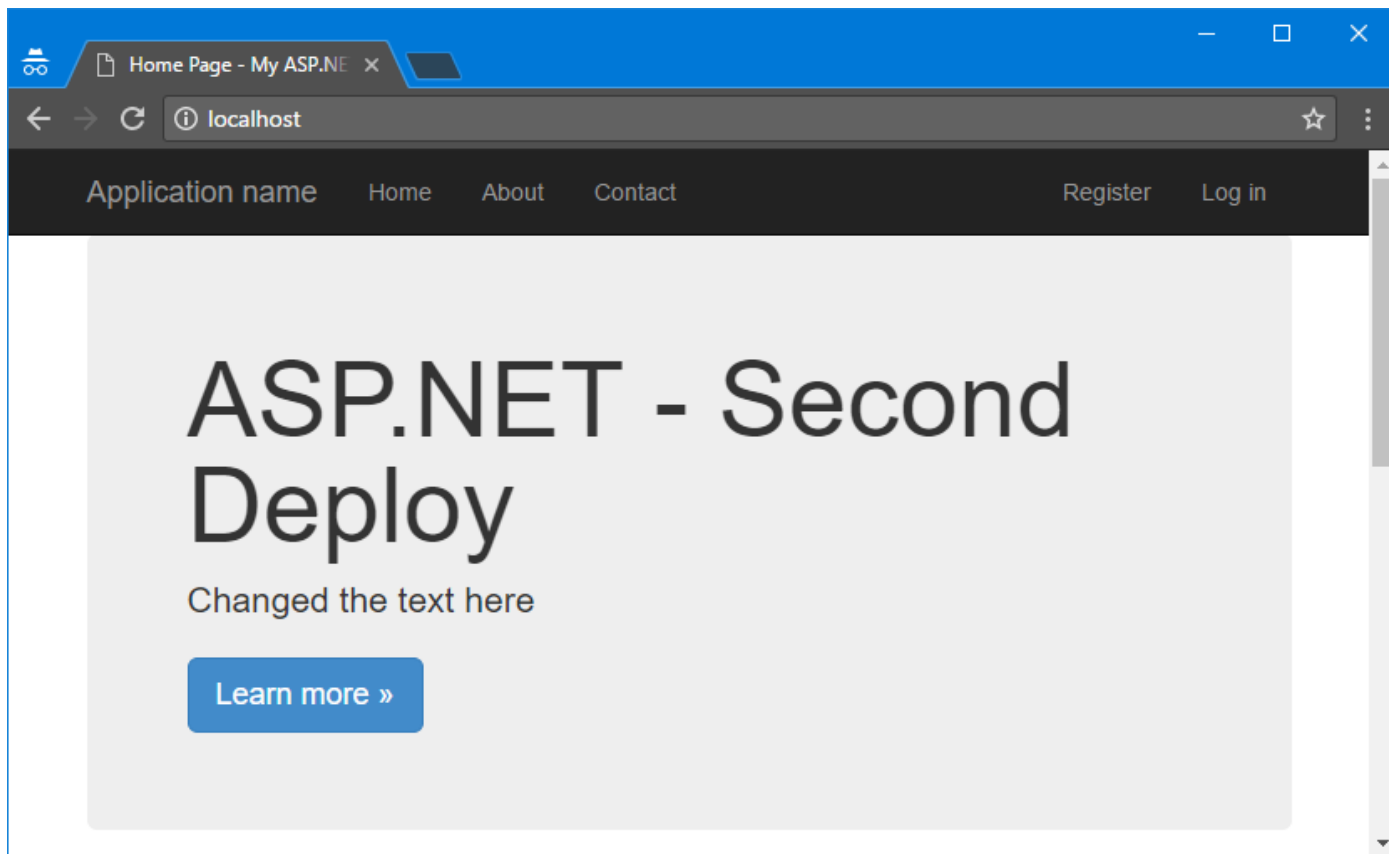
Then just click on the **Publish button**, because the Visual Studio remembers our publishing profile and there is no need to make other configurations for now.



Now to deploy it we should run the **command prompt as an administrator**, navigate to the folder with the **deploy package** and its deploy script and run it by **release.deploy.cmd /Y**

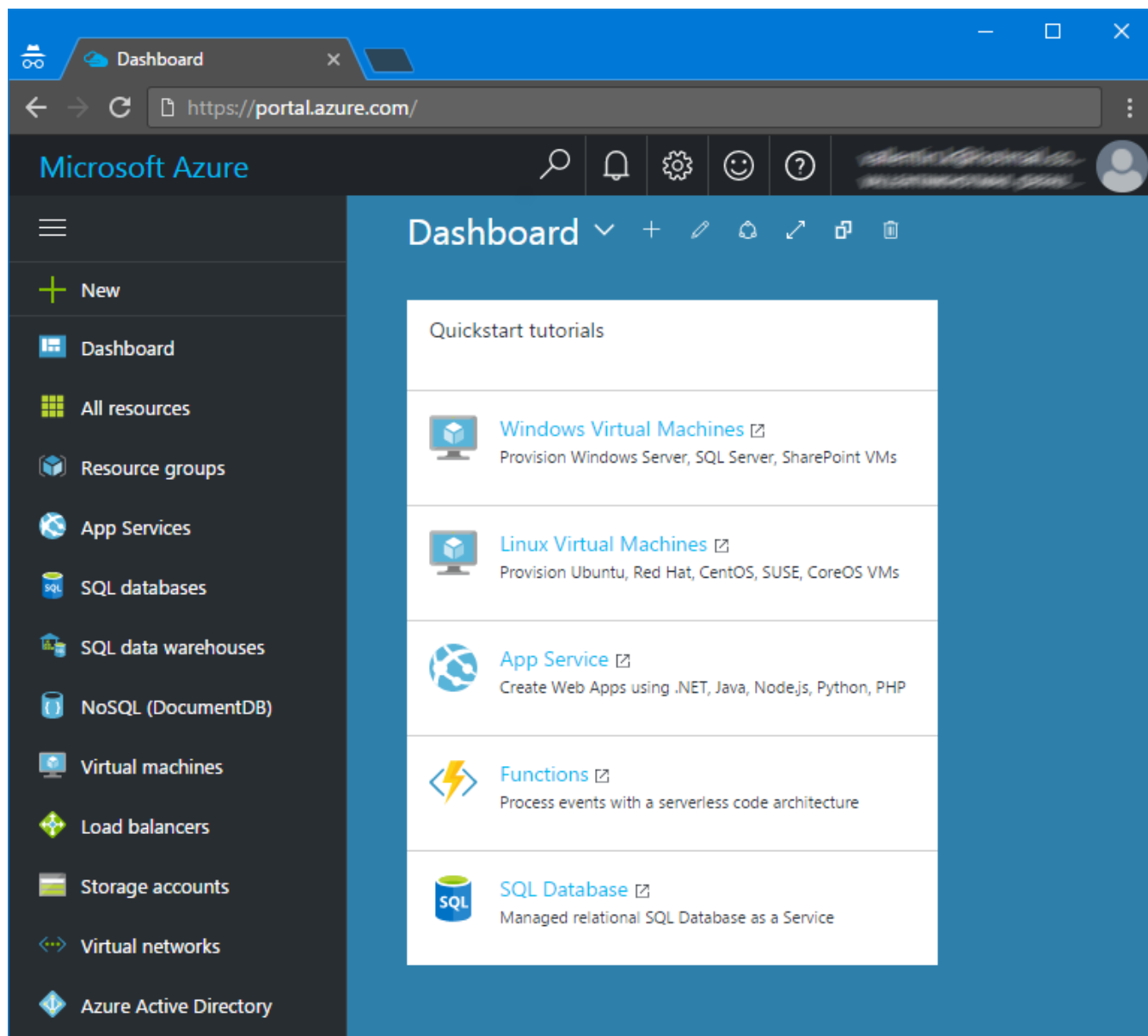
```
Administrator: Command Prompt
C:\deploy>release.deploy.cmd /Y
SetParameters from:
"C:\deploy\release.SetParameters.xml"
You can change IIS Application Name, Physical path, connectionString
or other deploy parameters in the above file.
-----
Start executing msdeploy.exe
-----
"C:\Program Files\IIS\Microsoft Web Deploy V3\msdeploy.exe"
-source:package='C:\deploy\release.zip' -dest:auto,includeA
cls="False" -verb:sync -disableLink:AppPoolExtension -disabl
eLink:ContentExtension -disableLink:CertificateExtension -se
tParamFile:"C:\deploy\release.SetParameters.xml"
Info: Updating file (Default Web Site\Views\Home\Index.cshtm
l).
Info: Adding ACL's for path (Default Web Site)
Info: Adding ACL's for path (Default Web Site)
Total changes: 3 (0 added, 0 deleted, 3 updated, 0 parameter
s changed, 1400 bytes copied)
C:\deploy>release.deploy.cmd /Y
```

After the script finishes the application should be live.

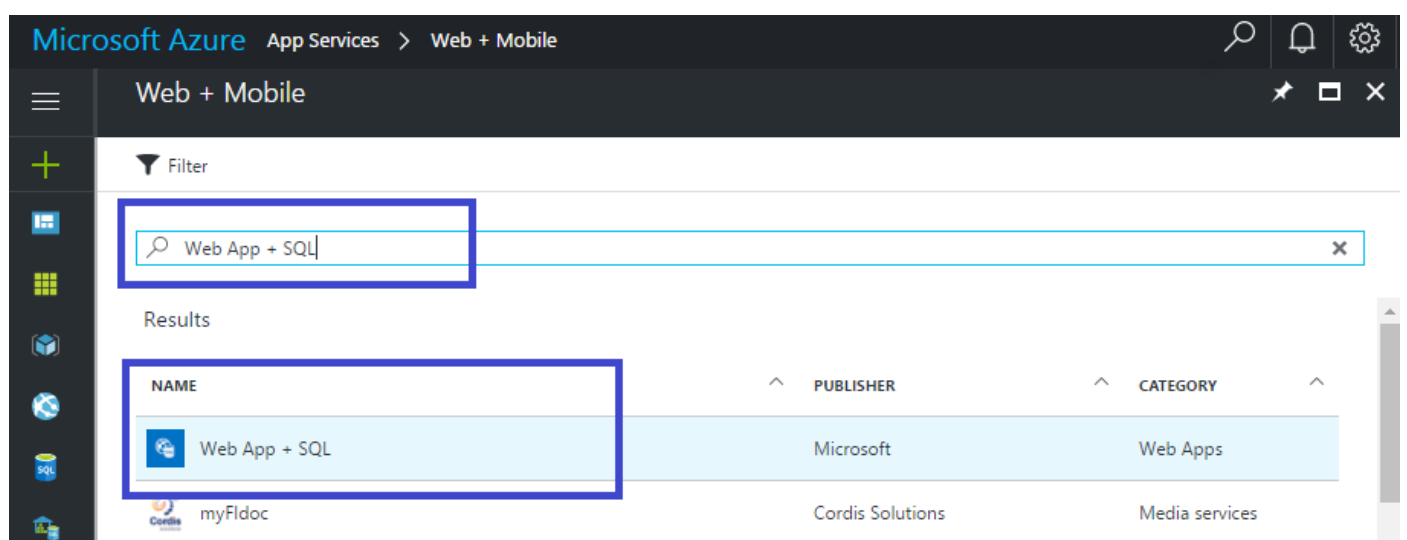
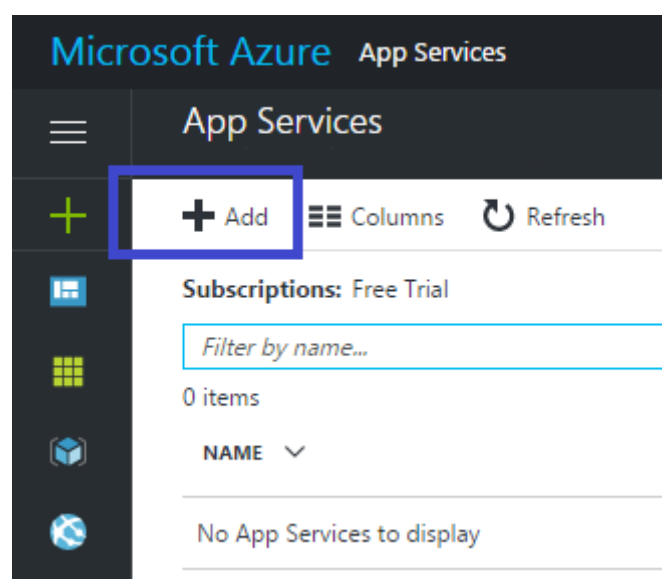
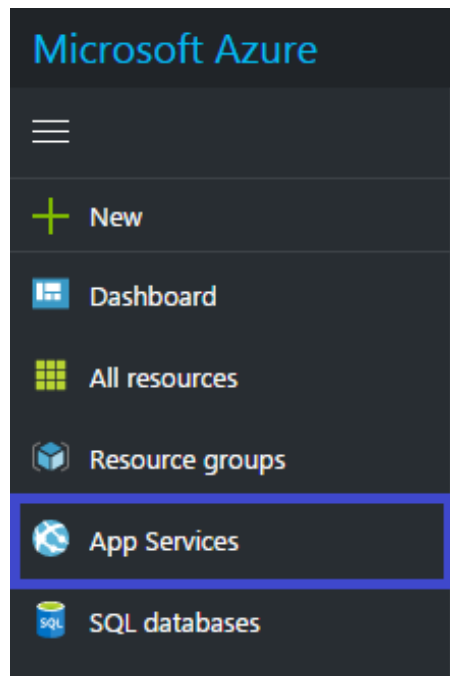


5. Deploying to Windows Azure

To deploy your project to **Windows Azure** you need to **have an account** there and **login in the Azure Portal**.



Then you need to **create a new App Service with Database**.



Web App + SQL
Microsoft

HOME
NOTIFICATIONS
OVERVIEW
JOURNALS
BILLING

Summary

MyTestGroup

MyTestSite1

MyTestFL...

Properties

QUICK START

Monitoring

Events in the past week

20
15
10
5
0

Alert rules

1 rules

Billing

Resource Costs

NAME	TYPE	CURRENT SPEND

Summary

MyTestGroup

MyTestSite1

MyTestFL...

Properties

QUICK START

Monitoring

Requests and errors today

100
80
60
40
20

0

0

PUBLISHER

Microsoft

Documentation

Solution Overview

Solutions you can deliver

Pricing Details

USEFUL LINKS

Create

Now we need to select the **type of subscription**, give name to the **resource group** that our application will use, **create database** for our application and select **App Service Plan**. In our case, we would use the **free service plan**.

SQL Database

*

Name

DeployAzureDb

✓

*

Target server

deployazuredb (West Europe)

>

*

Pricing tier ⓘ

Free: 5 DTU, 32 MB

>

*

Collation ⓘ

SQL_Latin1_General_CP1_CI_AS

Select

New App Service Plan

Create a plan for the web app

*

App Service plan

FreeTestPlan

✓

*

Location

West Europe

▼

*

Pricing tier

F1 Free

>

OK

Microsoft Azure App Services > Web + Mobile

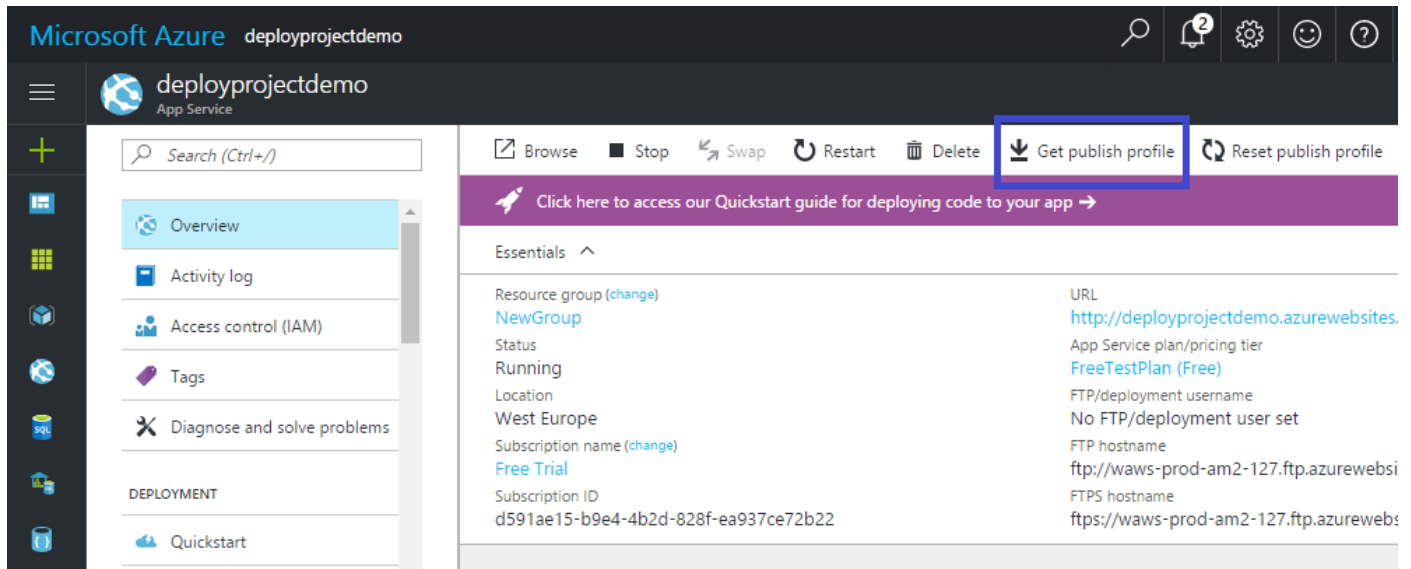
Web App + SQL

Create

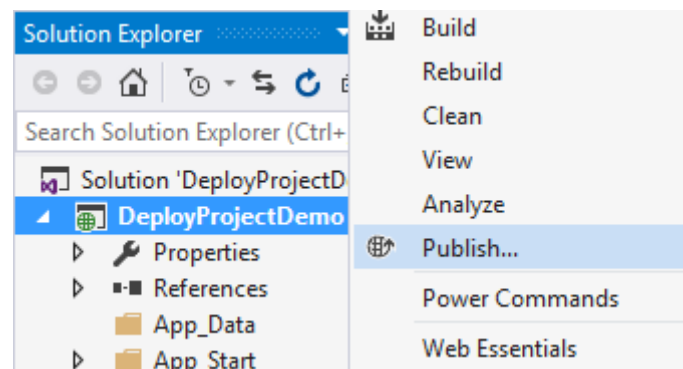
- * App name
deployprojectdemo ✓
.azurewebsites.net
- * Subscription
Free Trial ▼
- * Resource Group ⓘ
☒ Create new ☐ Use existing
NewGroup ✓
- * App Service plan/Location
FreeTestPlan(West Europe) >
- * SQL Database
DeployAzureDb >
- Application Insights ⓘ
- ☒ Pin to dashboard

[Automation options](#)

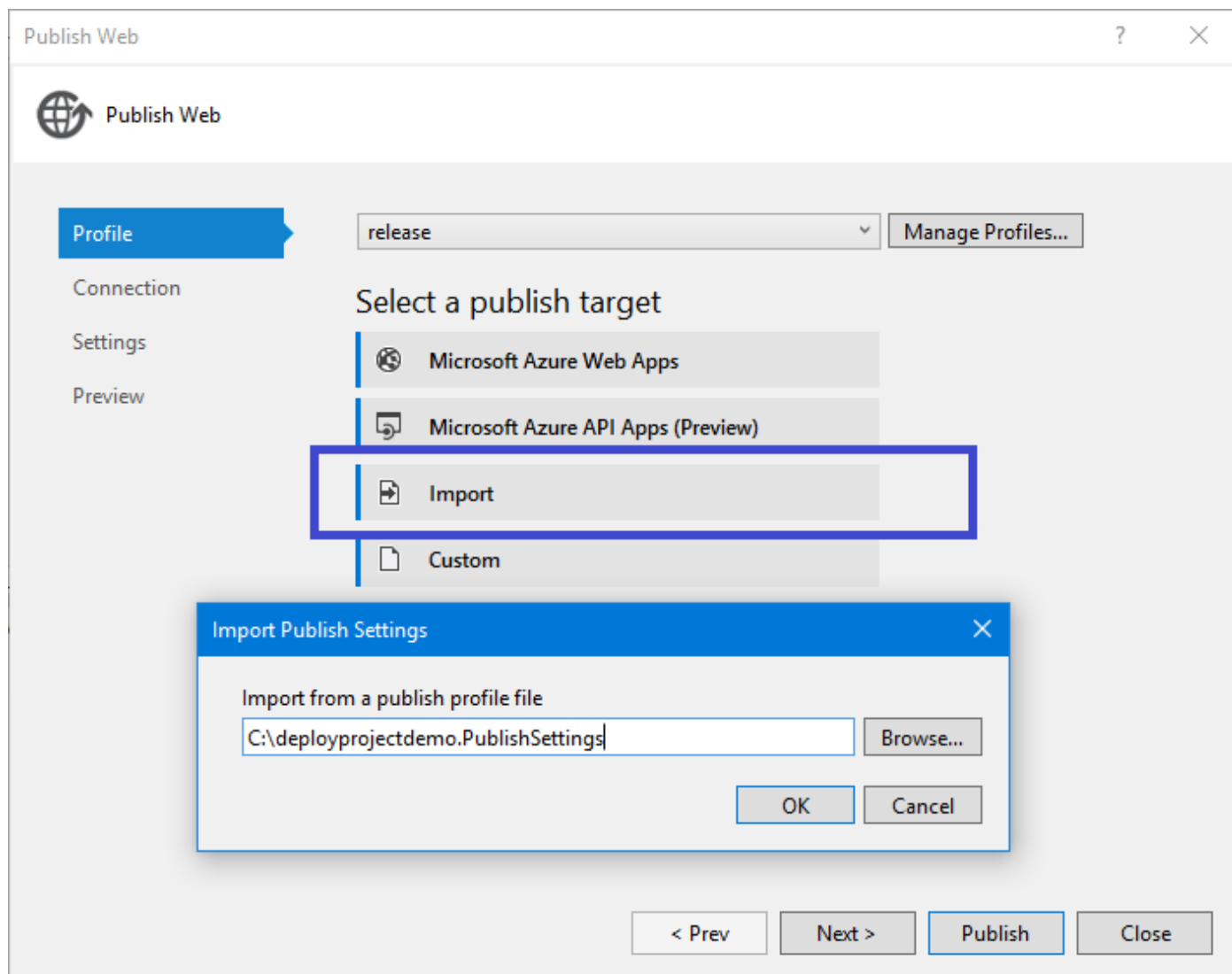
It might take several minutes for the application to become online. When it is running we should download, its publish profile.




Go back to Visual Studio and choose **Publish...** from the project's context menu.



This time we will not create custom publishing profile but we will **import** the one we've just downloaded from Azure Portal.



Publish Web ? X

 Publish Web

Profile
Connection
Settings
Preview

deployprojectdemo - Web Deploy

Publish method: Web Deploy

Server: deployprojectdemo.scm.azurewebsites.net:443

Site name: deployprojectdemo

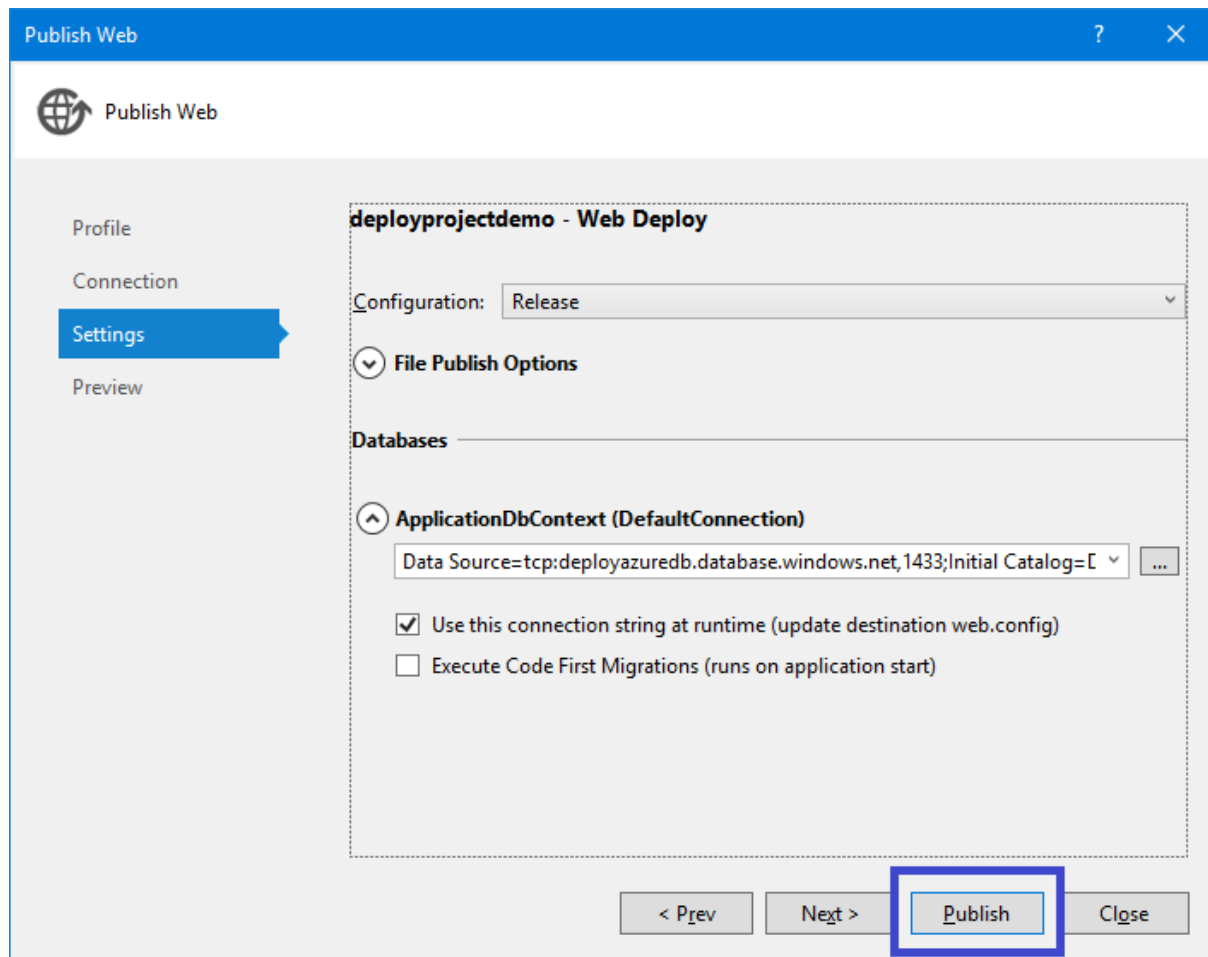
User name: \$deployprojectdemo

Password: Save password ☒

Destination URL: http://deployprojectdemo.azurewebsites.net

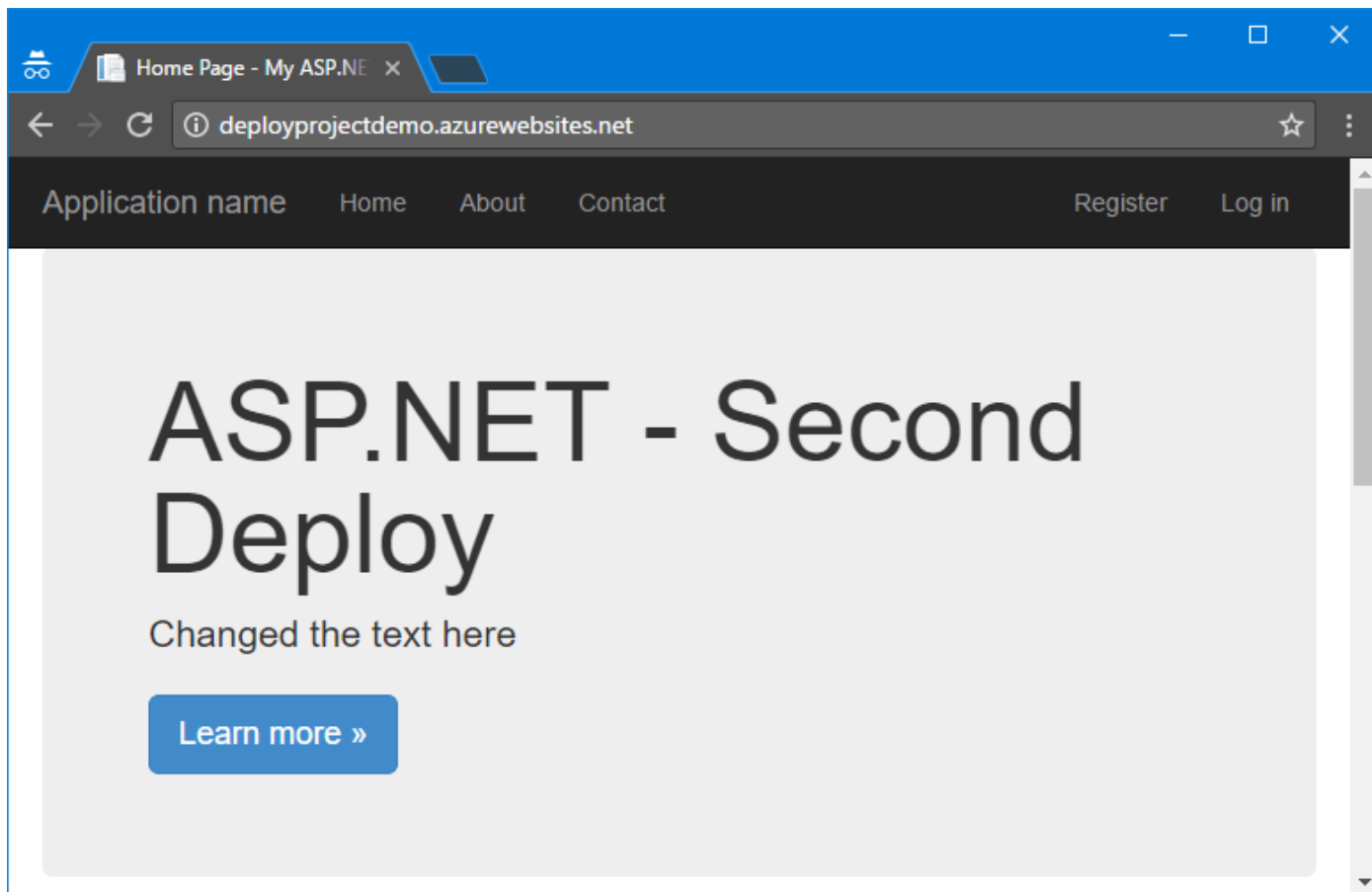
Validate Connection

< Prev Next > Publish Close



As you can see all of the settings we've created earlier this time are **automatically created** for us when importing that profile.

Finally, our web application is **up and running in the Azure**.



For future deployments to Azure you just **right click** on the project and choose **Publish...** then select the Azure profile and click on **Publish button**. It becomes truly one-click deployment.

