Q·BOT·X

☰ MENU

Laravel 5 on Azure - part 1

05 APRIL 2015 on azure, laravel

Ok, I'll be honest with you: At first I wanted to have a little 5 minute Screencast of me yapping on and on about Azure, Composer, Laravel and the whole configuration thing.

Turns out, I really can't stand my voice "on tape", so here we go with an old school blog entry.

Ok, where to start? At this stage, I assume you have already heard about Laravel, Composer and actually have an Azure subscription/account. If you haven't here's a little list to get you up to speed on the various topics:

- Composer
- Laravel
- Laravel 5 from scratch
- Azure

Let's get started

I won't run you through the process how to create a basic web application in Azure (it's really simple and there's ton's of documentation out there already). So at this stage you should already have a basic web application.I will also use the new Web-Portal of Azure and not the old one. Just to be sure, click on the Browse button next in your main Panel menu. You should see a nice blue web-page stating that this is a new web-app. If not, something is going wrong and you probably do want to check if the site itself is running and everything is like it should.

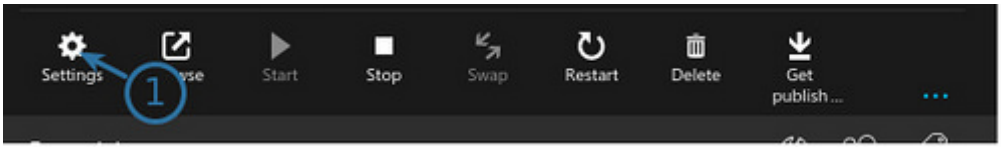
If you aren't quiet there yet, have a look here for a quick run down. The process is practically self explanatory.

Install Composer on Azure

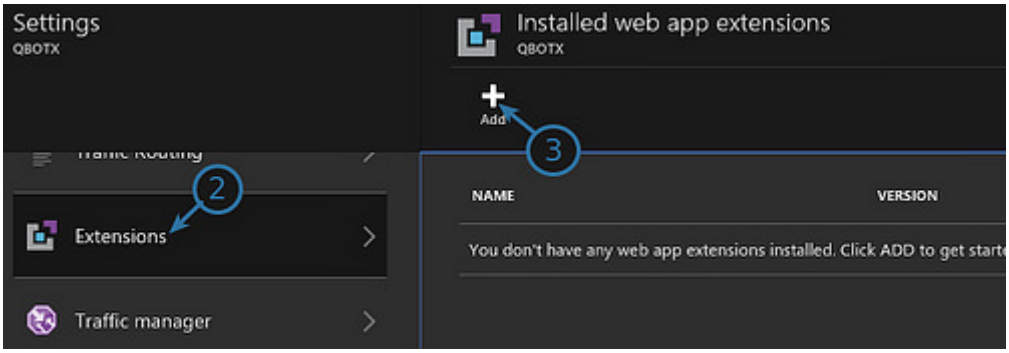
One of the minimum requirements of Laravel is composer. Out of the Box, Azure Web-Apps (or Sites) comes with PHP but without composer support. I hope Microsoft will come up with standard PHP Web-App templates that include composer.
However, composer can be easily installed on Azure via the extension manager. I know that there is already a very nice tutorial how to get extensions like Composer running on your Azure Web-Apps here but for completeness, let me run you through it anyway. As I said before, I will focus on the new web frontend so have a look at the previous link should you get stuck on the old portal.

Installing Azure Extensions

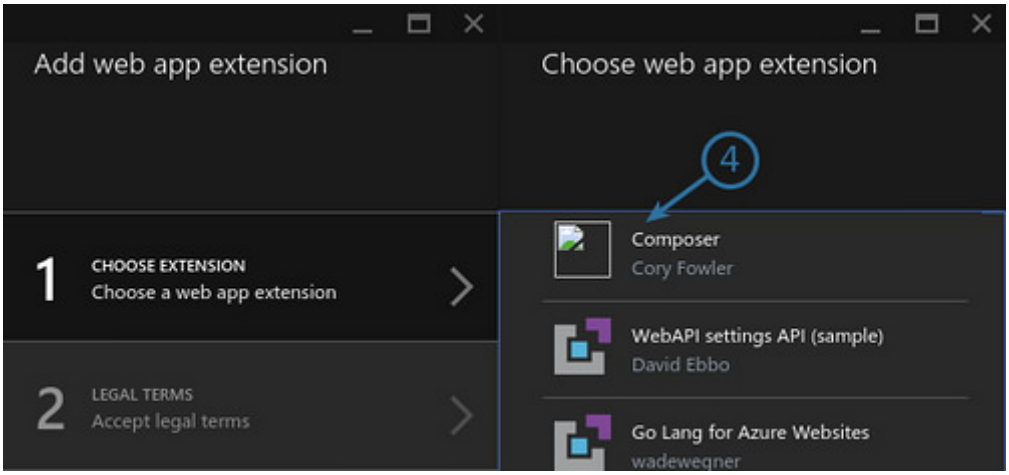
- Log in to your Azure Account, and click on the settings Icon



- Once you're in the "Settings" Sidebar select the "Extensions" menu item to open the "Installed web app extensions" panel
- Click the big + and yet another sidebar with the possible extensions will open for you.



- Select Composer from the list and go through the install process (you would have to agree to licenses etc.).



Install any additional extensions you might need (phpMyAdmin, Site Admin Tools or PHP Manager). We will use the Site Admin tools later on- so it won't hurt to install them now.

That's it- you're done. You have everything installed you need to run a Laravel Website on Azure. From this point onwards we will only configure our existing configuration.

Get your hands dirty...

At first, get your local Laravel installation going. I usually have the laravel installer on the system, but this should work with the standard composer way as well. Install it, then open a console/terminal window in your new Laravel application directory and check if the basics are running by typing:

```
php artisan serve
```

if you haven't changed anything in your application, it should come back with something like:

Laravel development server started on http://localhost:8000/

If you browse to this URL, you should see the Laravel Welcome-Page. And this is exactly what we want to see when we deploy to Azure.

We deploy our application to Azure using git. There are a lot of different ways to deploy to Azure, but I believe that git is possibly the simplest way to deploy (not just to Azure). Much like other services (openShift for example), Azure will build your application when you push your git repo to the cloud.

Let's start by stopping your development server if it's still running. Now, if you haven't already initialized the git repository do it now:

```
git init
git add .
git commit -m "Hey, Oh - Let's go!"
```

Once you've done this, add the cloud as one of your deploy end-points by typing the following in your terminal:

```
git remote add <endpoint name> <your git clone url from the
Essentials panel>
```

Sidenote: *Make sure you pick a sensible endpoint name, that will make your life easier in the long run. For example, if you have a staging/dev Server, just call it like that to avoid confusion in the future. Simple conventions can make your life better!*

Do not push your application just yet ... we need to do something else first!

Public Directory

There are a few things that are a bit different on Azure to what you're used to if you're usually deploying to a linux environemnt. You will be used to different conventions like public or public_html as the standard http serving directories for Web-Servers like Apache. The same thing is true for your root directory, by default Azure deploys everything you push to it into the wwwroot directory. Laravel doesn't like this! So let's go and make Laravel happy with a few little tweaks to our configuration:

1. Click on **Settings** in the main panel for the web-app
2. Pick **Application Settings** from the Sidebar and the **Web app settings** sidebar will open.
3. In the new Sidebar scroll down to **App Settings** create two new entries
4. `SCM_REPOSITORY_PATH` with the value `..\repository`
5. `SCM_TARGET_PATH` with the value `..`

once you have done this scroll down to **Virtual applications and directories** and change the public directory value from `site\wwwroot` to `site\public`

If you now click the Browse button, you will see that your pretty page from before now has been replaced with a rather disapointing message: *The page cannot be displayed because an internal server error has occurred.*. This should be no surprise, as we

haven't deployed anything to our precious new web app yet. Let's do that now!

Let's deploy

Alright, so we have covered a lot of the things we should do, except for one thing (but more on that later). Let's go ahead and try deploying our first laravel app to Azure. Open a Terminal in your Laravel app directory and type:

```
git push <endpoint name> master
```

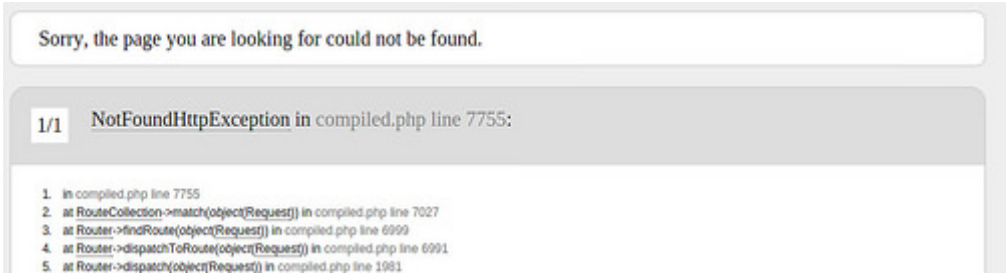
You will see a lot of things happening in your console after you've hit enter. This is because your composer is pulling everything it needs together to put together your first Laravel app on Azure. Give it a few seconds and wait till the Autoload file has been generated.

Are we there yet?!

We are... kind of. Just one more thing and then we're done, I promise! Remember how I said that we have covered nearly everything we should do to get our Laravel app going? Ok let's find out what we are missing.
If you point your web browser to your azure URL, you will see Laravel's welcome page. That's awesome! But what happens when you go to an unconfigured page on your system?

OK, let's assume for a moment that you want to go to a URL on your Site that hasn't been setup yet... for example **http://yourapp.azurewebsites.net/contact**. By default Laravel

would refer to it's standard 404 page, which result in a message somewhat like this:



But what we see now is different (and even more disapointing in a way) because all we see is this:

The resource you are looking for has been removed, had its name changed, or is temporarily unavailable.

So, why is that? Well,- Laravel assumes that it's dealing with an Apache Web-Server, therefore it comes with the mod_rewrite rules specifically for that. Azure Web Apps (or Sites) is using Microsoft's IIS. This means that we need to tweak a little bit to get the URL rewrite working. Create a new file called **web.config** in your public directory, and paste the following content into that file:

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.webServer>
    <urlCompression doDynamicCompression="true"
doStaticCompression="true"
dynamicCompressionBeforeCache="true"/>
    <staticContent>
      <remove fileExtension=".svg" />
      <mimeMap fileExtension=".svg" mimeType="image/svg+xml" />
      <mimeMap fileExtension=".woff"
```

```
mimeType="application/font-woff" />
      <clientCache httpExpires="Sun, 29 Mar 2020 00:00:00 GMT"
cacheControlMode="UseExpires" />
    </staticContent>
    <httpProtocol>
      <customHeaders>
        <add name="Strict-Transport-Security" value="max-
age=31536000; includeSubDomains"/>
        <add name="Access-Control-Allow-Origin" value="*" />
        <add name="Access-Control-Allow-Headers" value="X-
Requested-With,Content-Type" />
        <add name="Access-Control-Allow-Methods"
value="POST,GET,OPTIONS,DELETE,PUT,PATCH" />
      </customHeaders>
    </httpProtocol>
    <rewrite>
      <rules>
        <rule name="Laravel5" stopProcessing="true">
          <match url="^" ignoreCase="false" />
          <conditions logicalGrouping="MatchAll">
            <add input="{REQUEST_FILENAME}"
matchType="IsDirectory" negate="true" />
            <add input="{REQUEST_FILENAME}" matchType="IsFile"
negate="true" />
          </conditions>
          <action type="Rewrite" url="index.php"
appendQueryString="true" />
        </rule>
      </rules>
    </rewrite>
```

```
  </system.webServer>
</configuration>
```

Alright, safe the file and add it to your repository, and push the changes to the cloud:

```
(assuming that you are in the laravel public directory)
git add web.config
git commit . -m 'IIS Tweaks'
git push <end point name> master
```

If you point the web-browser to an unconfigured route after the deploy, you will see standard laravel error page.

Well, that's it for now. There is not much more to it, but we haven't touched environment configuration and databases yet. I will do this in part 2 in a few days.

28 Comments qbotx 1 Login

Recommend 3 Share Sort by Best

Join the discussion…

Rares • a year ago
Thanks for the tutorial ! It really helped me.
1 • Reply • Share ›

Johnny • a year ago
Really helpful tutorial. Thanks **@robopetr**
1 • Reply • Share ›

Johnny Ng • 4 months ago
Worked well for me :) Thanks for the tut
• Reply • Share ›

Patryk F Szady • 8 months ago
I'm pulling my hair out... the welcome screen works but I get the browser default 404 for everything else. I've followed all the steps and added the web.config file into my /public folder. Any ideas?
• Reply • Share ›

Nguyen Quy Hy • 10 months ago
Do you have any problem running "php artisan optimize" on Azure. Both the Kudu console and Git deployment log stop after "Generating optimized class loader" and only services.json is created (compiled.php is missing).
• Reply • Share ›

1actobacillus ➚ Nguyen Quy Hy • 8 months ago
I have the same problem. "php artisan optimze" appears to hang both in Kudu and it git deployment. Did you manage to solve it?
• Reply • Share ›

Nguyen Quy Hy ➚ 1actobacillus • 8 months ago
If you have load of time for each deployment, you can disable timeout of composer. I did that and the deployment succeeded every time, but it took 3 hours each. Later on, I chose to remote "php artisan optimize" out of the automatic process and run it manually in Kudu only after I change dependencies. CI is blazing fast now. I still cannot find a better solution.
• Reply • Share ›

1actobacillus ➚ Nguyen Quy Hy • 8 months ago

I replaced "php artisan optimize" with "composer dump-autoload -o" in my composer.json file, in the post-install-cmd step. It works for me now.

▲ | ▼ • Reply • Share ›

George Imerlishvili • a year ago

As I see it's for azure web app... Did you try run laravel 5 as PHP cloud service?

▲ | ▼ • Reply • Share ›

robopetr Mod ➚ George Imerlishvili • a year ago

Sorry- I don't get the question- could you explain?

▲ | ▼ • Reply • Share ›

George Imerlishvili ➚ robopetr • a year ago

In azure you can host site as web app or cloud service (look at screenshot), and you made Laravel setup using web app, I was interested if you tried to setup Laravel using cloud service...

▲ | ▼ • Reply • Share ›

Tyrion Lannister • a year ago

Very helpful. I normally use IIS and import the .htaccess to a web.config and found that the web.config from my local machine works on Azure as well.

Something like:

<configuration>

<system.webserver>

<rewrite>

<rules>

<rule name="Imported Rule 1" stopprocessing="true">

<match url="^(.*)/$" ignorecase="false"/>

<conditions logicalgrouping="MatchAll">

<add input="{REQUEST_FILENAME}" matchtype="IsDirectory"

see more

• Reply • Share ›

Justin • a year ago

Thanks for the tut. I followed your instructions with an app I've been developing (so not a fresh install of laravel) but I'm struggling with a permissions issue, when I deploy I just get a blank screen which suggests that the storage folder is not writable. After a bit of research it looks like Azure webapps won't allow folders outside of the webroot (or /public after we changed it) to be written to. Did you come across this issue, if so how did you get around it?

• Reply • Share ›

Antonio Bojorges ➚ Justin • a year ago

I'm having the same prob as Justin. Just checking if he might have solved it any how.

• Reply • Share ›

robopetr Mod ➚ Justin • a year ago

Hi Justin - sorry for the late reply, but I am very busy at the moment and only slowly catching up :). If all breaks down, I try to install a blank Laravel 5.x application as described above (I never had any storage-folder-rights problems with this approach). After I've seen the correct page Laravel welcome page I would try to migrate bit by bit. Sorry if that sounds cumbersome- but if I don't see the actual code- I won't be able to help you. You could also go and set this environment temporarily to "dev" to get the propper error messages. - Hope that helps somehow.

• Reply • Share ›

Franck Garcia • a year ago

Did you experience performance problems with Laravel on Azure?

• Reply • Share ›

robopetr Mod ➚ Franck Garcia • a year ago

No, I can't say I have. What pricing tier are you on, and what are you using as db?

• Reply • Share ›

Franck Garcia ➚ robopetr • a year ago

I'm on the basic pricing tier and I'm using mysql (clearDB) for testing.
I also have a version using SQL Azure (local app, db on azure), and it's much slower than local mysql (even with the premium pricing).

• Reply • Share ›

robopetr Mod ➚ Franck Garcia • a year ago

That is very odd. I have had nothing of that kind.

To be fair, I use the latest version of PHP they offer, and I have a small VM that runs a MySQL Database (I've got a couple of other small things running on that too). I did this because the the driver they are currently offering for Azure SQL is just for the older versions of PHP (have a look at the other tutorials I have on this site).

When you say Local App, do you mean that the app is living in your Azure environment as well, or are you referring to your development environment?

^ | v • Reply • Share ›

Franck Garcia ➚ robopetr • a year ago

Ok I'll check the tutorials.
I mean that the app is living on my development environment.

^ | v • Reply • Share ›

ramseylove • a year ago

The welcome view loaded just fine so i thought was all good but when going to the built in home view with auth controller. I get a 404 error and this message: The resource you are looking for has been removed, had its name changed, or is temporarily unavailable.

^ | v • Reply • Share ›

ramseylove ➚ ramseylove • a year ago

My web.config file was in the wrong directory, i moved it to the public directory and everything was happy

^ | v • Reply • Share ›

robopetr Mod ➚ ramseylove • a year ago

Thanks for sharing!

^ | v • Reply • Share ›

Infacq • a year ago

look forward your part 2. Since using Azure maybe you can explore how can we make use their database especially SQL Database and laravel. How's that

^ | v • Reply • Share ›

robopetr Mod ➚ Infacq • a year ago

Part 2 is online now. I have had a brief look at the MS SQL Database option, but I didn't get it to run. I've ended up using mySQL again.

I might go back and try it again.

• Reply • Share ›

Infacq ➚ robopetr • a year ago

am also struggling to get it connect. Why is it so hard? hahhaha

• Reply • Share ›

robopetr Mod ➚ Infacq • a year ago

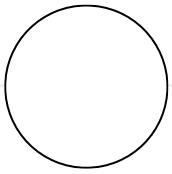
OK, so I've got it! Have a look here -> http://blog.qbotx.com/laravel-...

• Reply • Share ›

robopetr Mod ➚ Infacq • a year ago

No Idea :) I think the core problem is the Driver difference (between Linux/Windows as they seem to have a different one on Windows). Another possibility is that I'm just stupid. ;-P

• Reply • Share ›



Peter Katelaan

Dad, Geek, maybe even a bit nerdy from time to time. Movies, Robot's and Tech of all kind. OS Agnostic (Trust me, I really don't care).

Auckland *http://www.qbotx.com*

Share this post