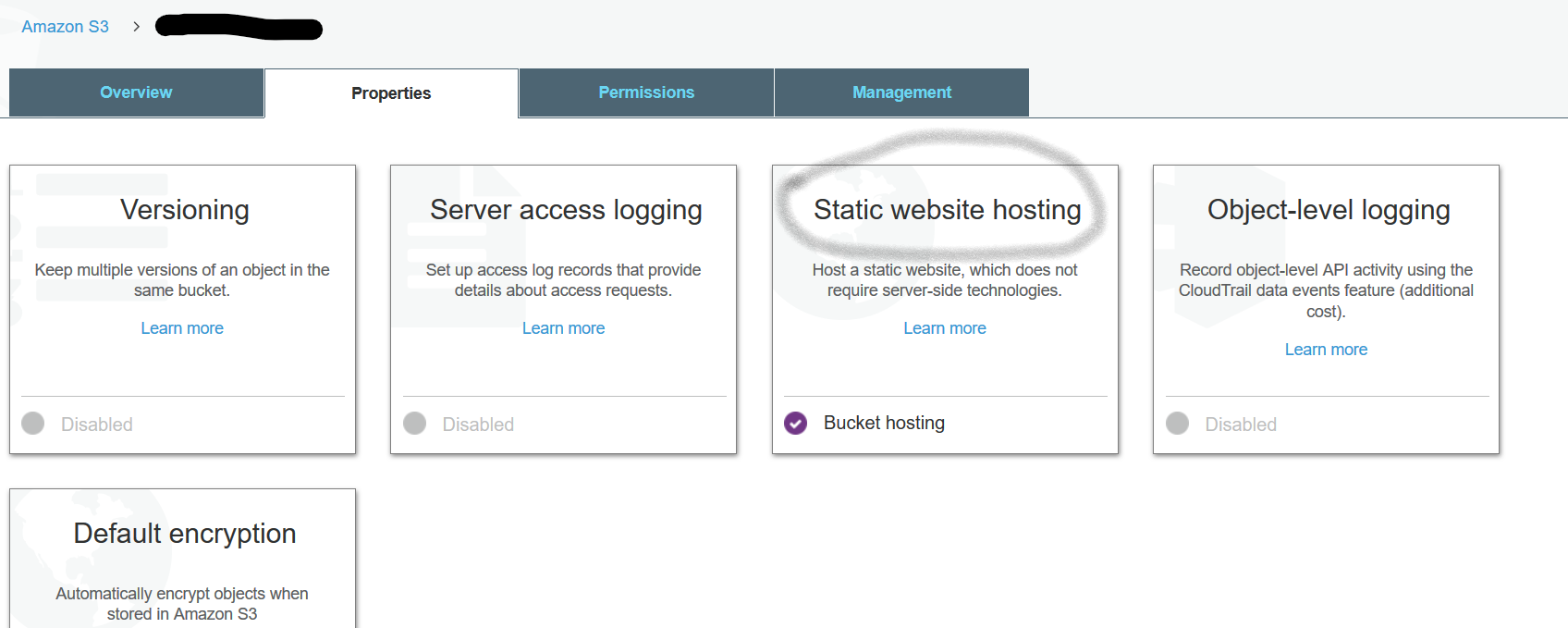


Hello, this is Tim, Unicorn Images programmer plus systems plus troubleshooting plus bao-ga-liao guy. If you are reading this, probably means that the system pecah liao. Don’t worry, this is why the boss asked me to write this document at 4pm on my last day at work.

1. Getting the front end running

This is probably your number one priority so that our customers think we still have a working website. The files are in /frontend folder.

I was told that we should store these files in Amazon S3, and use static web hosting to host the files. So I think you should create a bucket in Amazon S3 in the Singapore region, then upload all the files to in the frontend folder to that bucket and then enable static website hosting.



If it works you should see our website up again!

1. Getting the APIs up and running

But that is just the static website – if customers upload any images it won’t do anything, very embarrassing. So we need to fix up the APIs.

So, as our legendary CEO told me, when a customer uploads an image, we are supposed to apply some filters to the image so it looks cool, and then resize the filtered images down to thumbnails (50x50 pixels). If there is any text in the image, it’s also supposed to be recognised and be displayed.

So we have 3 APIs to build:

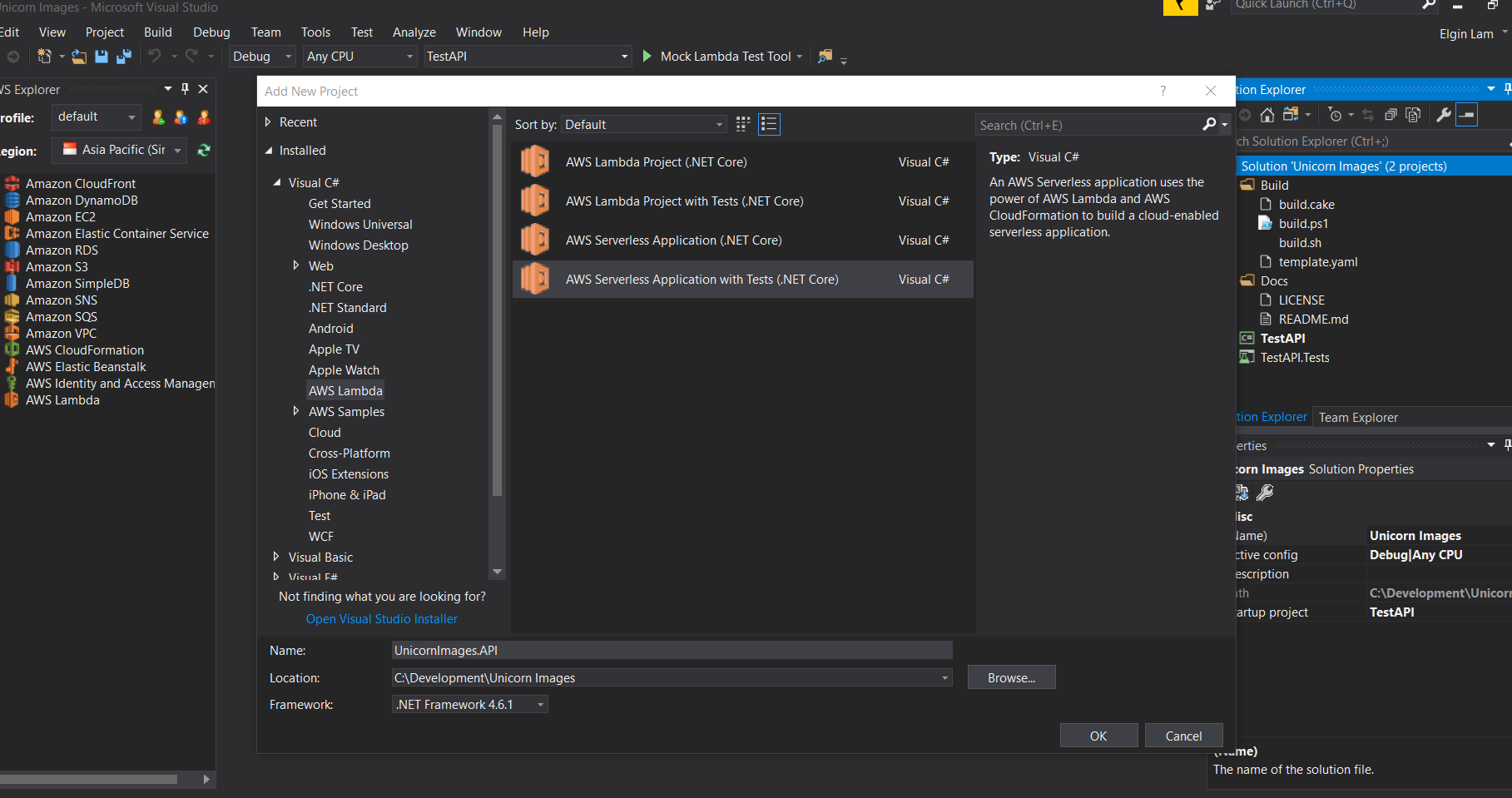
* Image Filter API
* Resize API
* Text Recognition API

The resize API will be called by the image filter API.

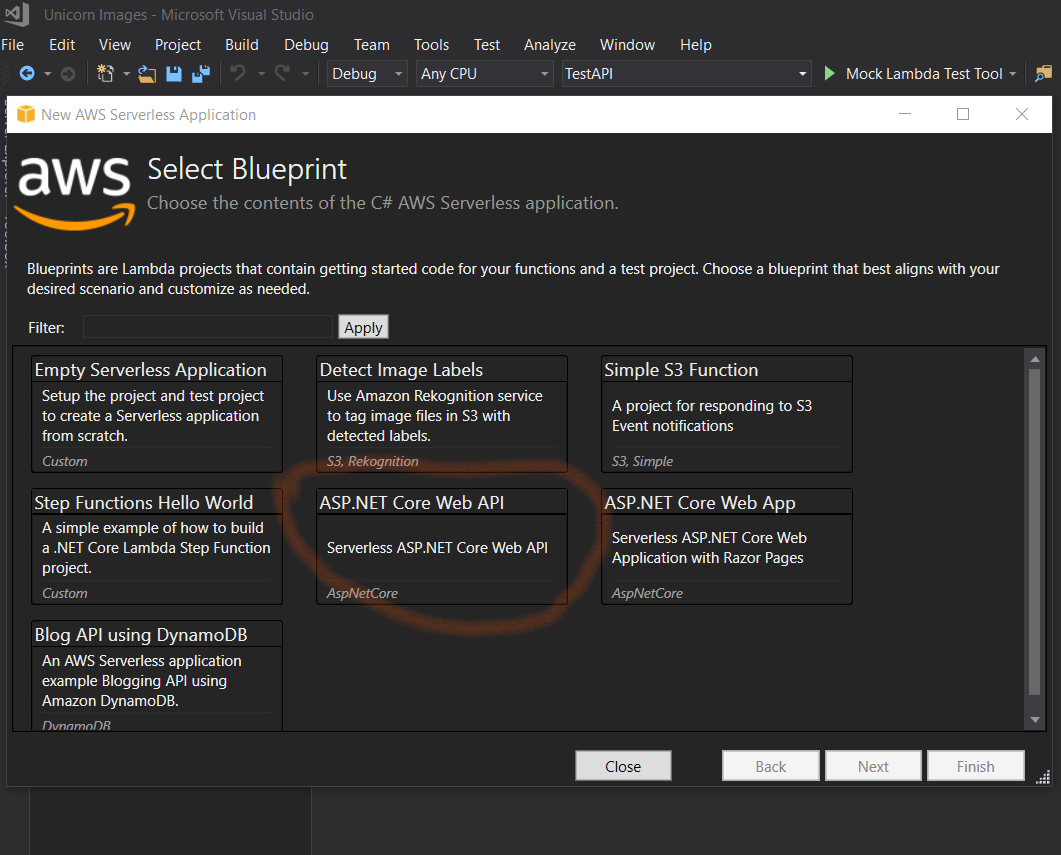
Remember, the investors want us to use “microservices”, so we should treat each service independently. So let’s tackle each service independently and once the service is passing your tests, you can integrate everything into one application. And then our customers will stop sending us angry emails like “your app doesn’t work” and “I should have stuck with Instagram”.

**Start Building**

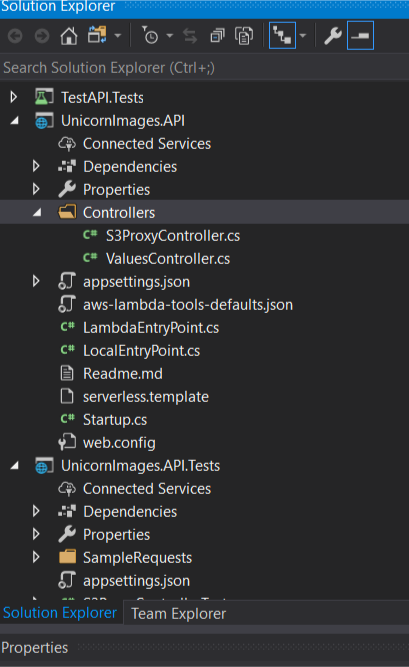
Since we’re going to use API Gateway and Lambda (serverless) for these, let’s create a serverless project. Call it UnicornImages.API or something cooler if you want. Why serverless? Because we can use .NET Core for these APIs, which means a lot less heavy lifting for us and we don’t need to worry about provisioning any infrastructure.



We’re creating an API, so select the ASP.NET Core Web API blueprint.

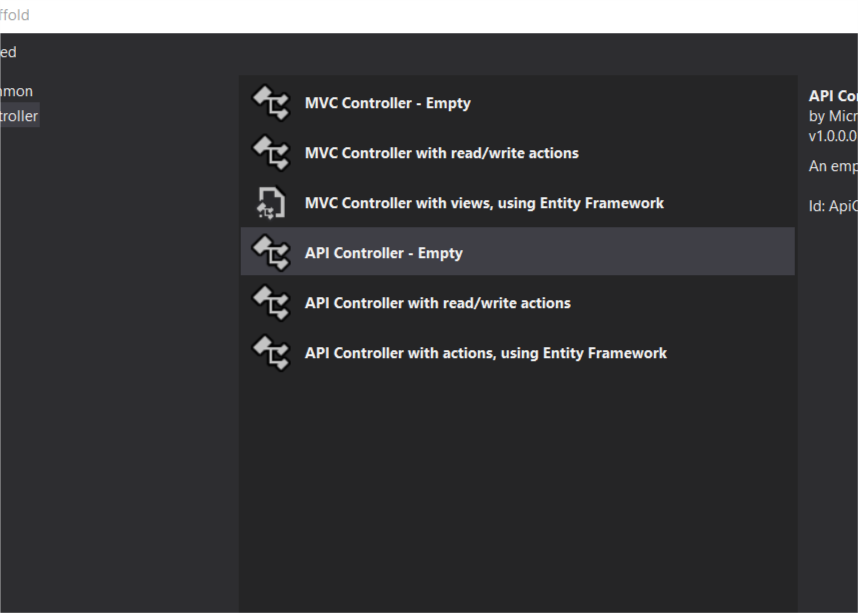


You are going to end up with a very familiar-looking ASP.NET Web API project.



Go ahead and add more controllers, maybe call them ResizeController, FilterController and RecogniseController? Up to you.

Select an empty controller when prompted, since we don’t expect to do any CRUD:

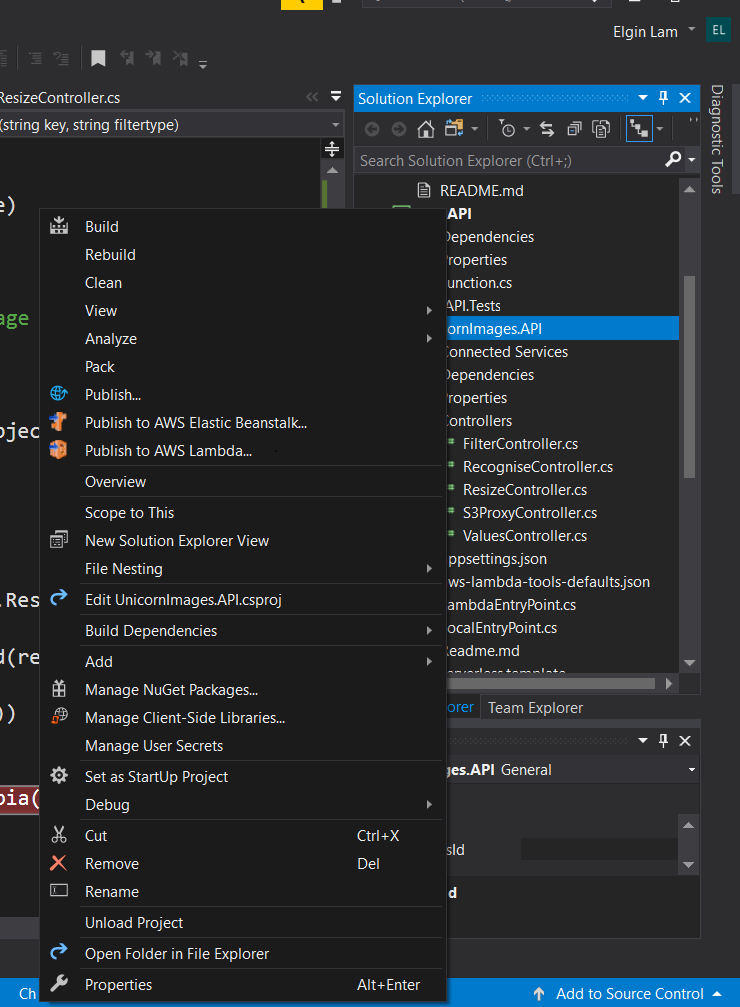


One controller you’ll notice is the S3ProxyController; this is used to proxy calls to Amazon S3 so that we don’t need to expose the user to our S3 buckets directly. You’ll want to use this functionality so go ahead and modify the bucket name in Startup.cs to a bucket that is actually in your account. When you actually deploy your application, the S3 bucket will be created for you automatically.

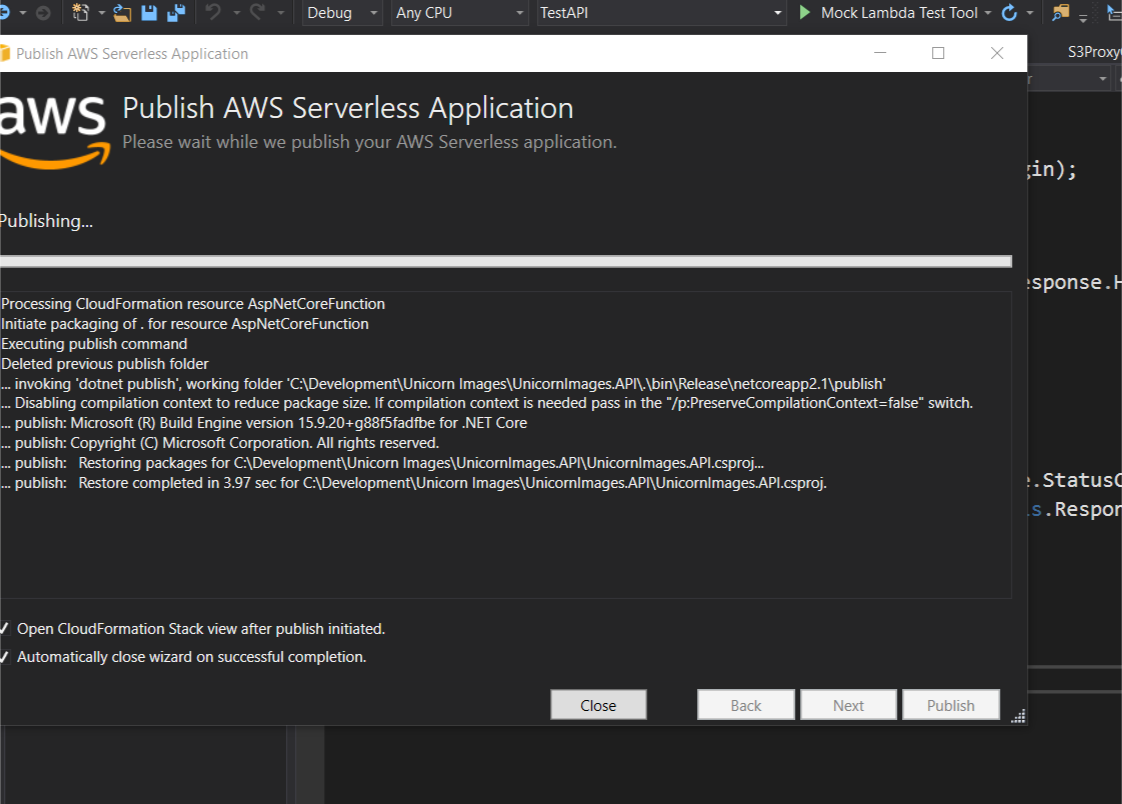
For the image manipulation, because Microsoft hasn’t released .NET Core compatibility for System.Drawing on Linux yet, we will need to use a third party library. Let’s go to manage NuGet packages for the project, select and install the **SixLabors.ImageSharp** and **SixLabors.ImageSharp.Drawing** packages.

Some filters that our users have asked for include sepia, gaussian blur, Kodachrome, Lomograph…you can play around with more and make Instagram really angry that we offer more filters!

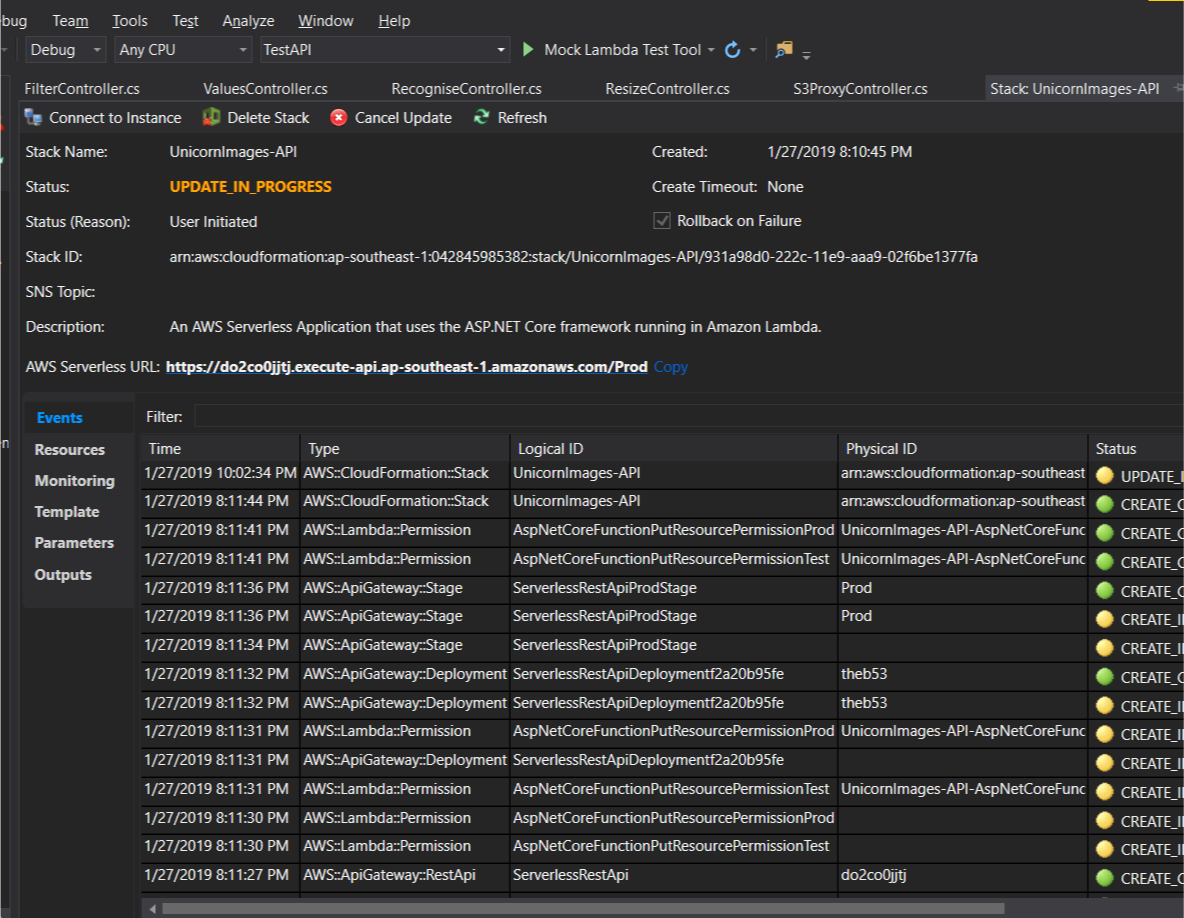
Once you’ve actually gotten the solution up and running, you should think about doing a build. To do this, right click on the project and select “Publish to AWS Lambda”.



Once you have filled in the options, you should be able to get a window showing you the publishing progress:



Next, you will see the progress of CloudFormation as it publishes the stack:



**Testing your API**

You can test the APIs that you are writing by using Unit Tests, or by going to tools like Postman or SoapUI to test via REST.

Hey look, it’s my last day so I’m gonna just scoot okay?

Good luck with it all I’m sure you’ll be fine!

**Tim**