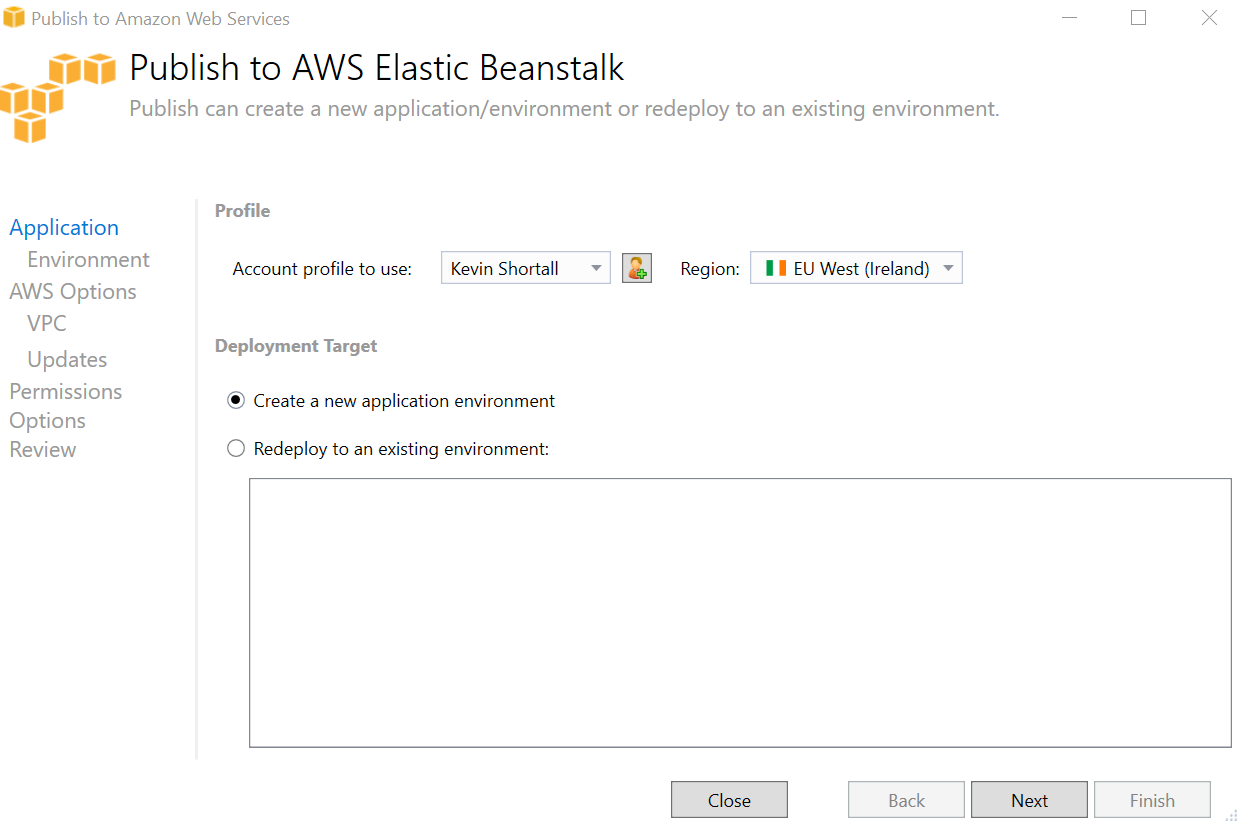
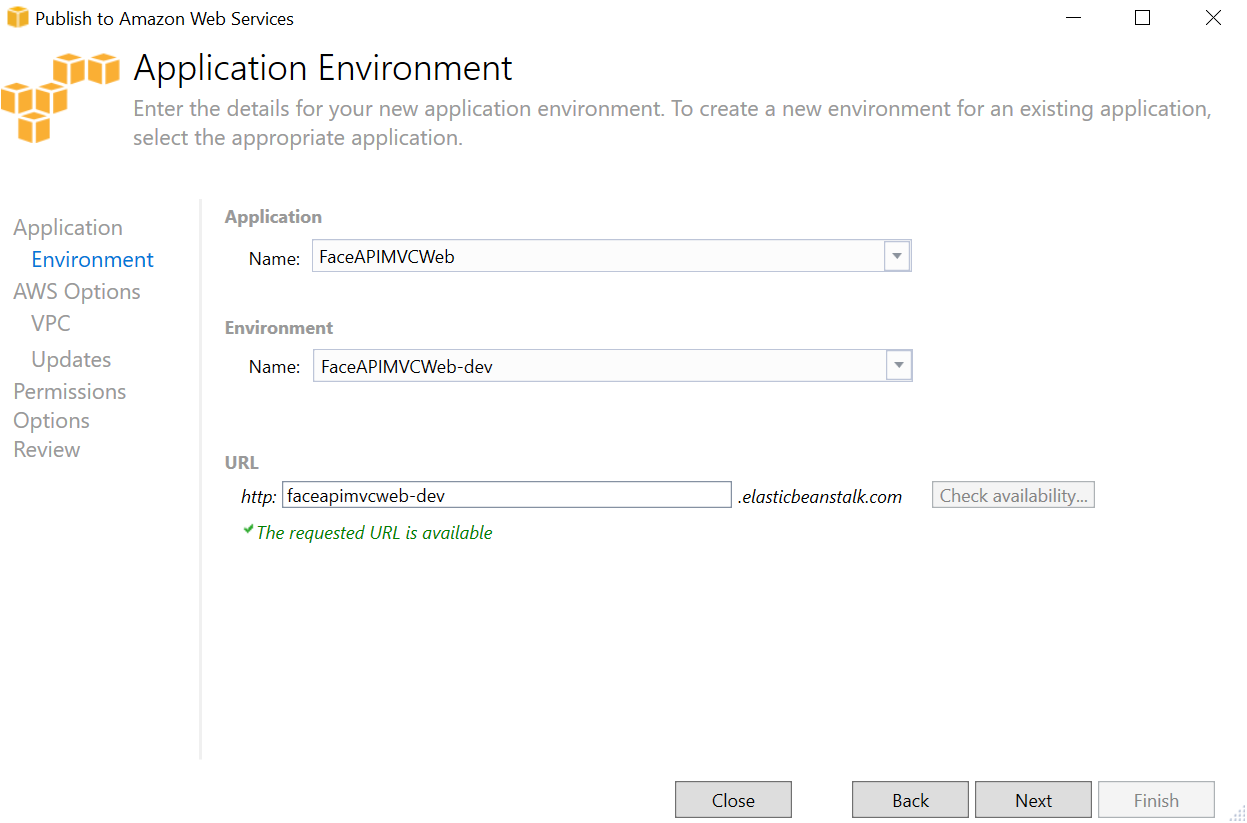
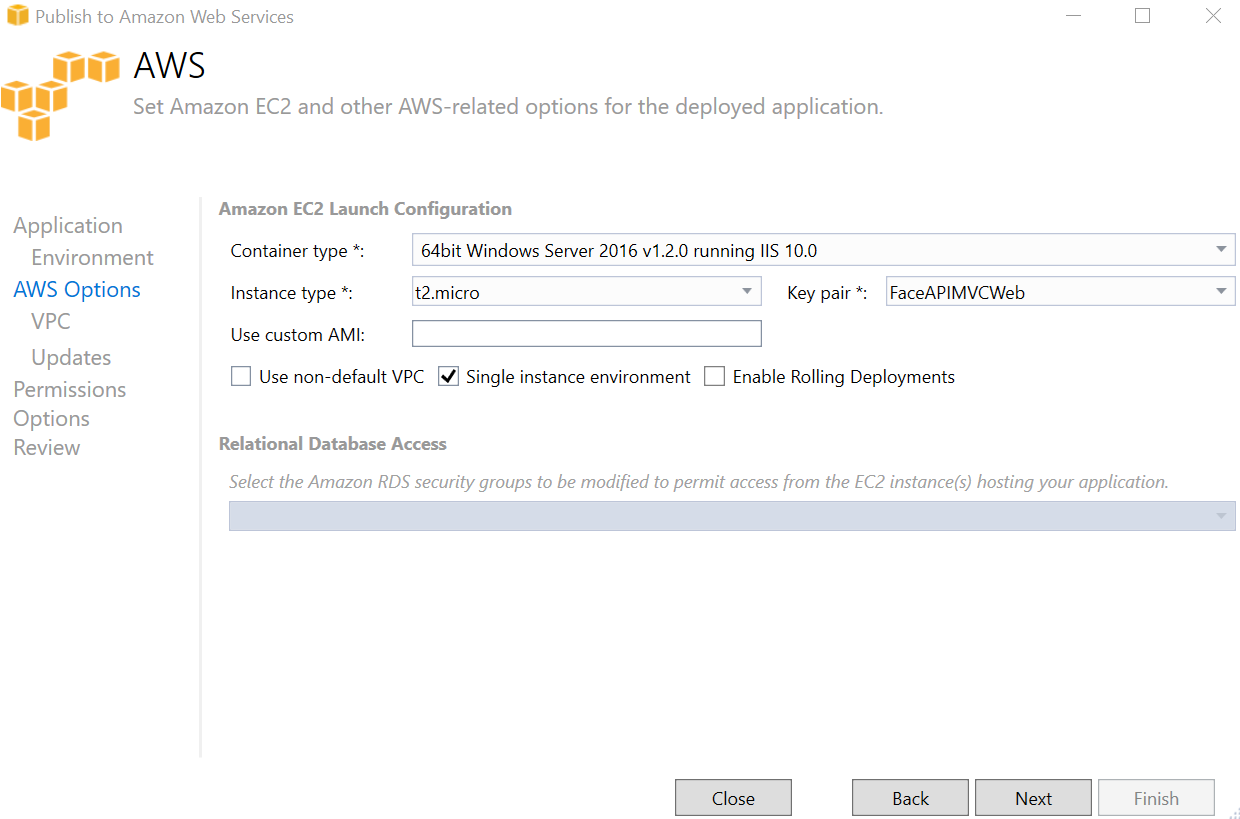
# 22/11/2017: Starting again from scratch. Settings.

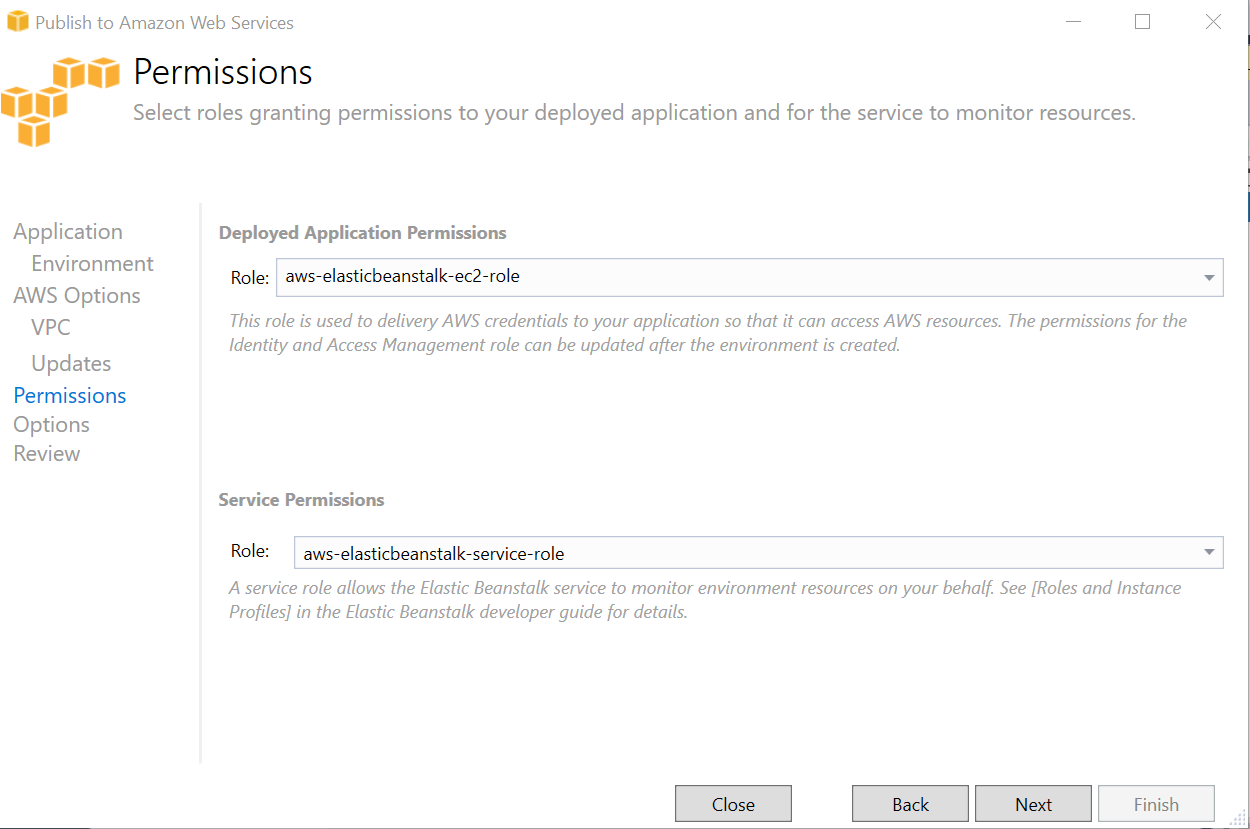
# Attempt 1:

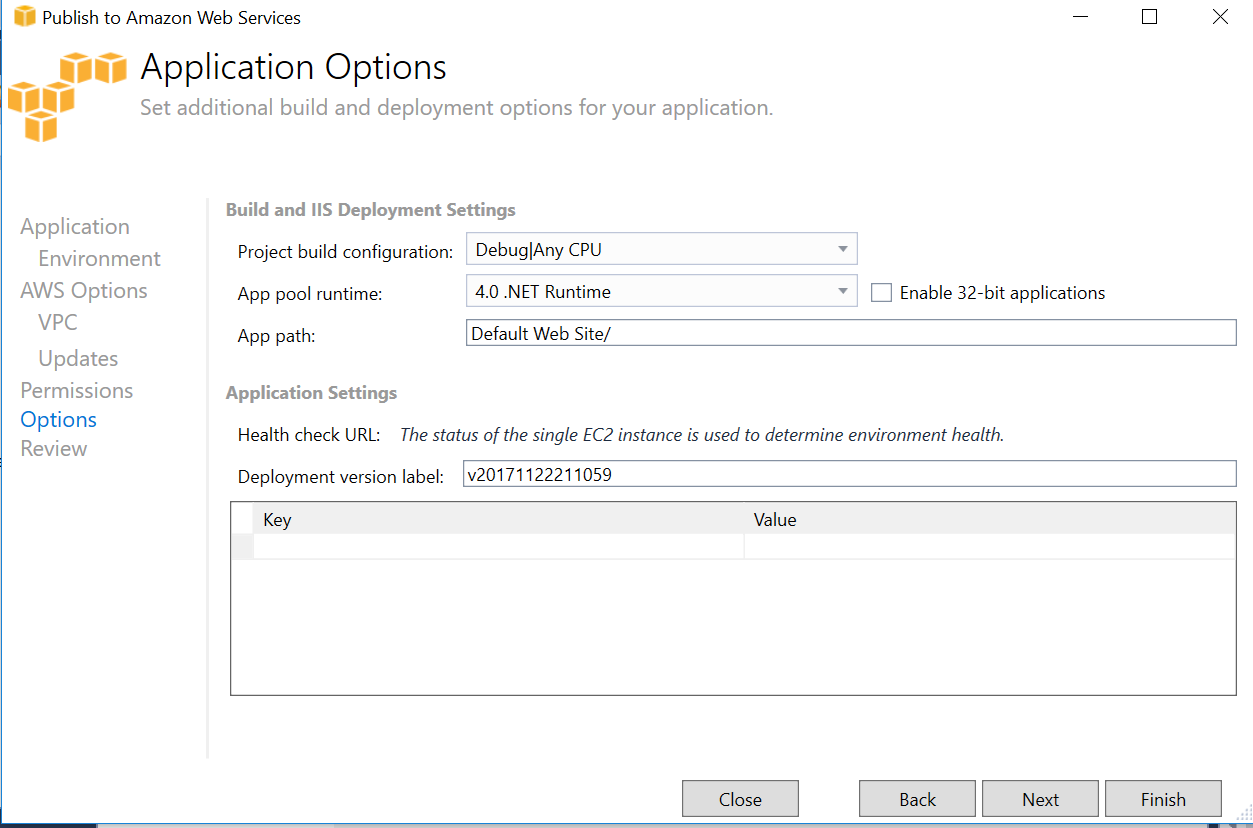
Note – source FaceAPI files are stored on the laptop desktop in this attempt. They are not stored in the Cloud anywhere. This may be a factor versus earlier attempts.

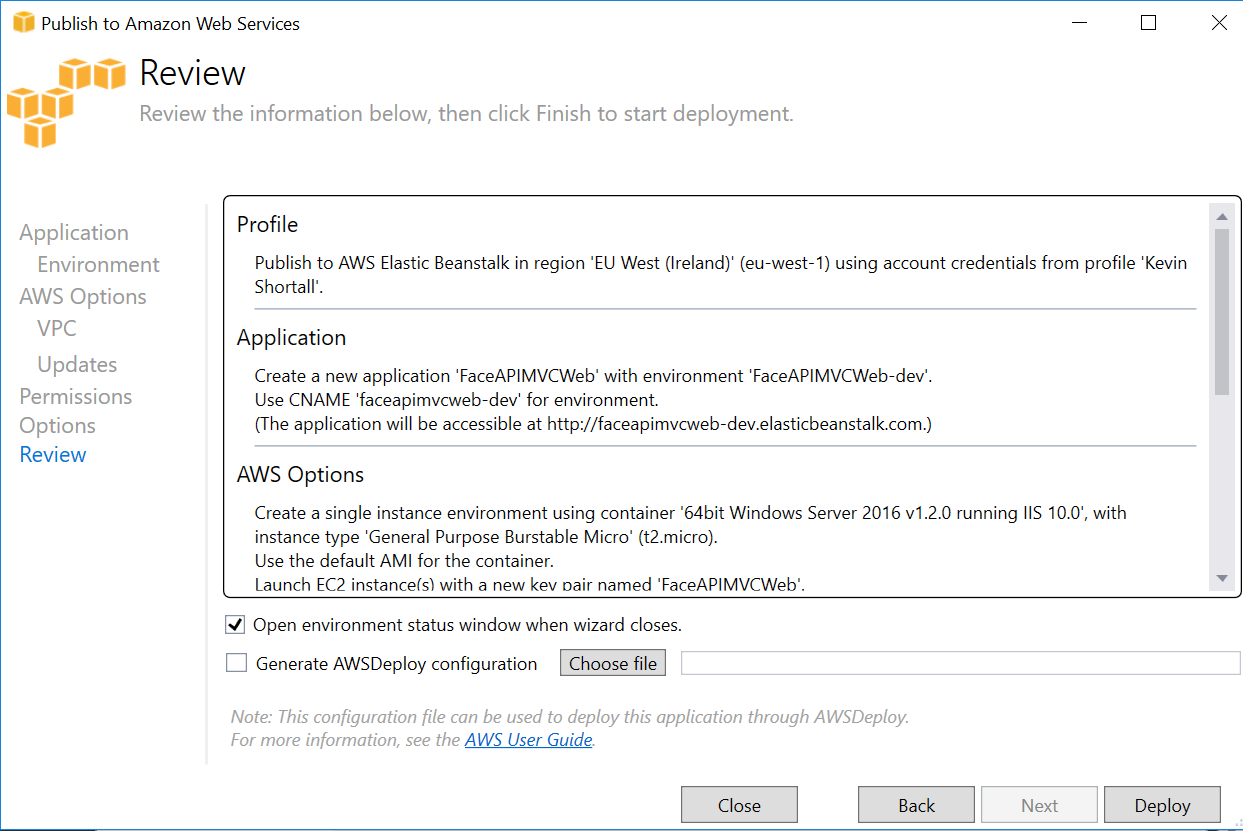




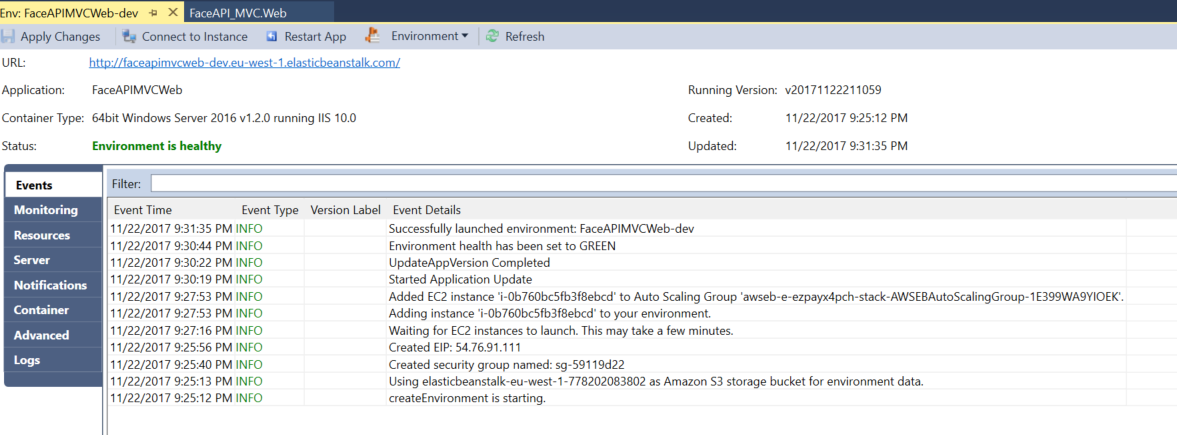




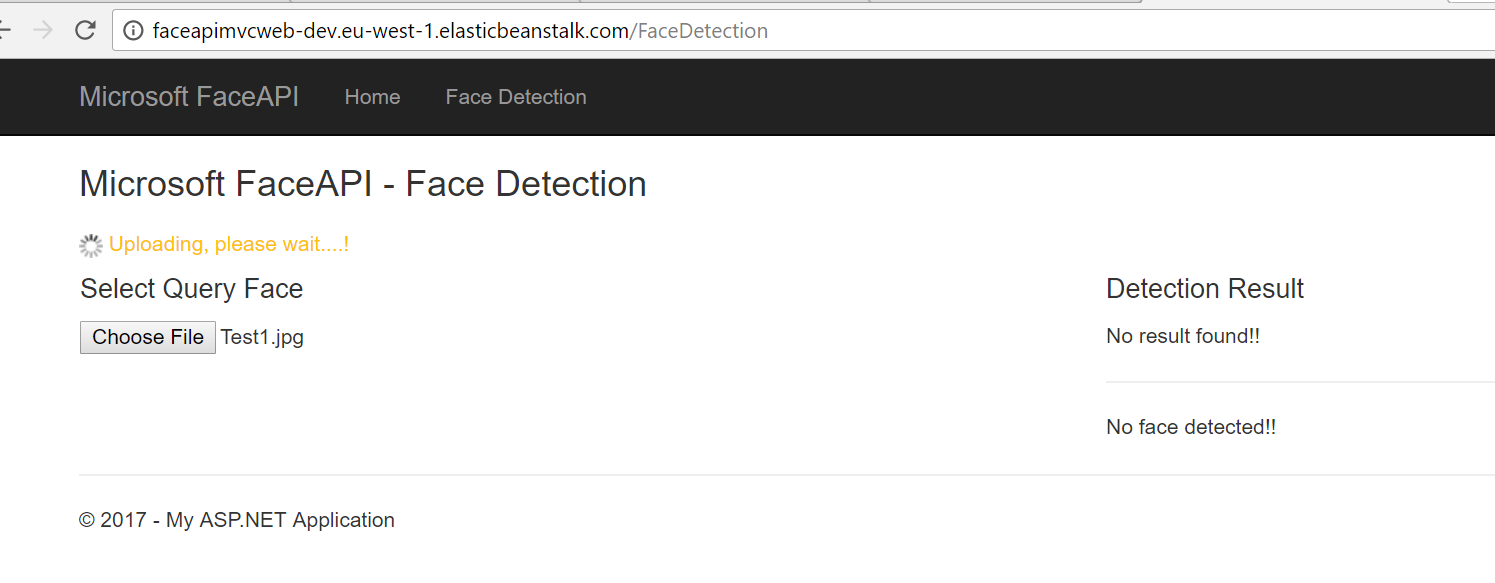




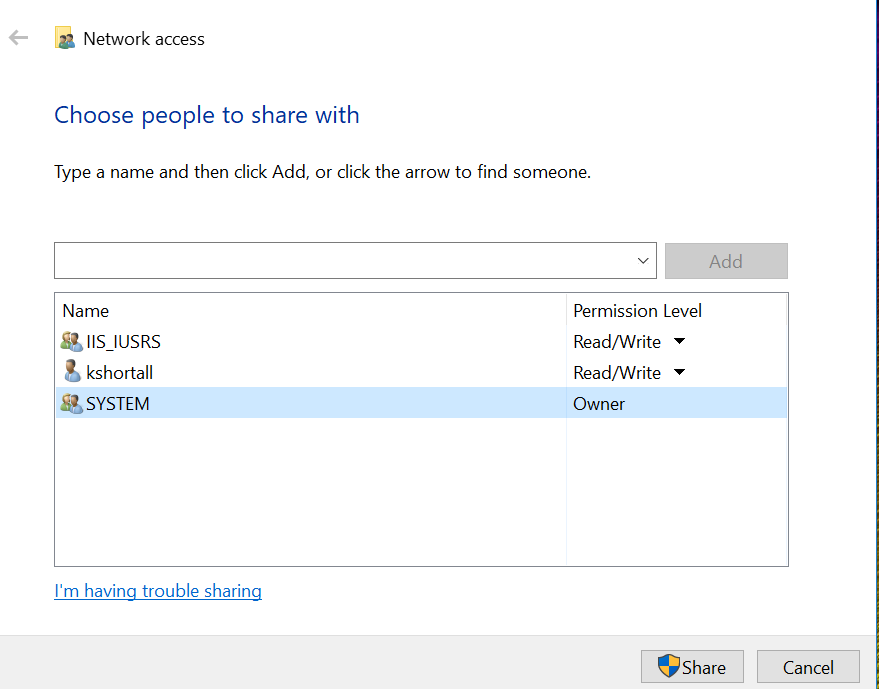
Loaded successfully in Visual Studio:

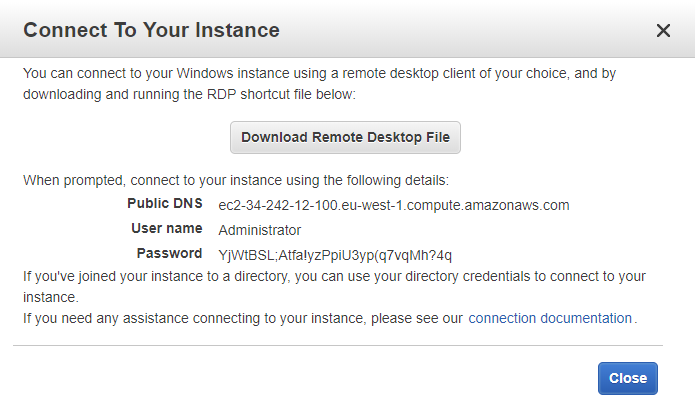


Same uploading problem as previously:

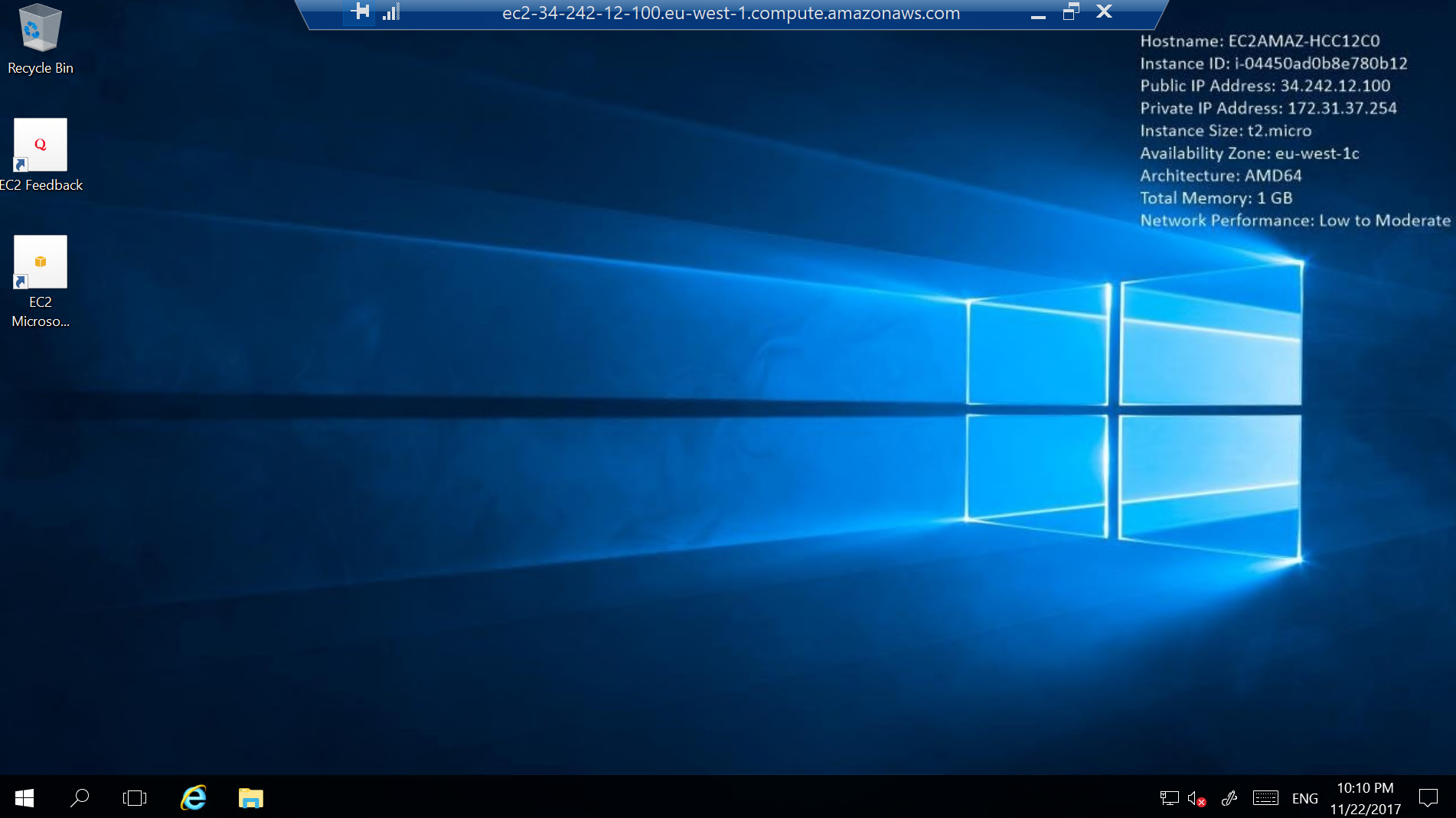


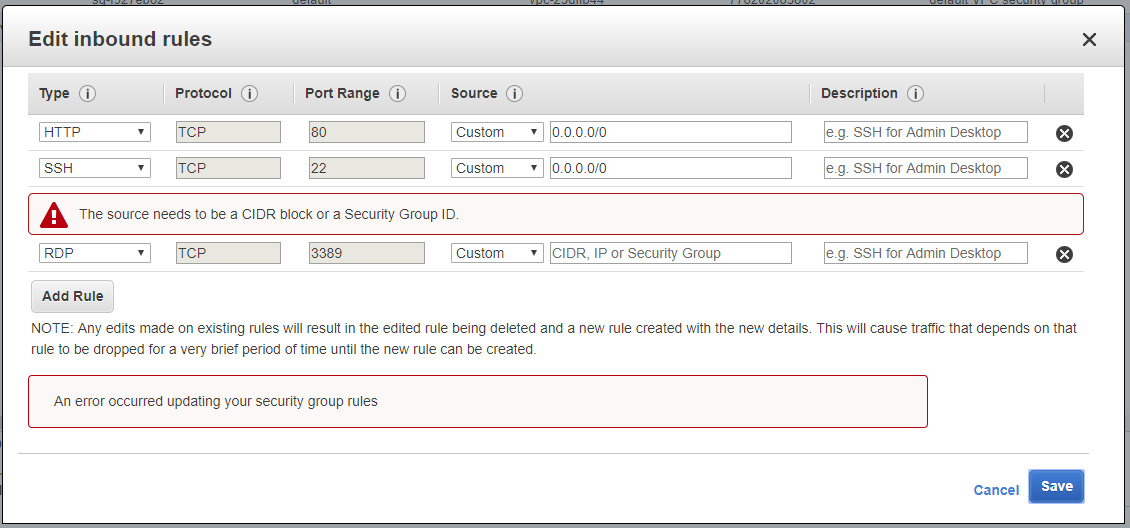
Updated user access security on my laptop to see if that helped. It did not – still uploading problem.





Access denied initially. Had to go back into the Security Group, edit the inbound rules to add an RDP rule. Access to Remote Desktop was then secured.



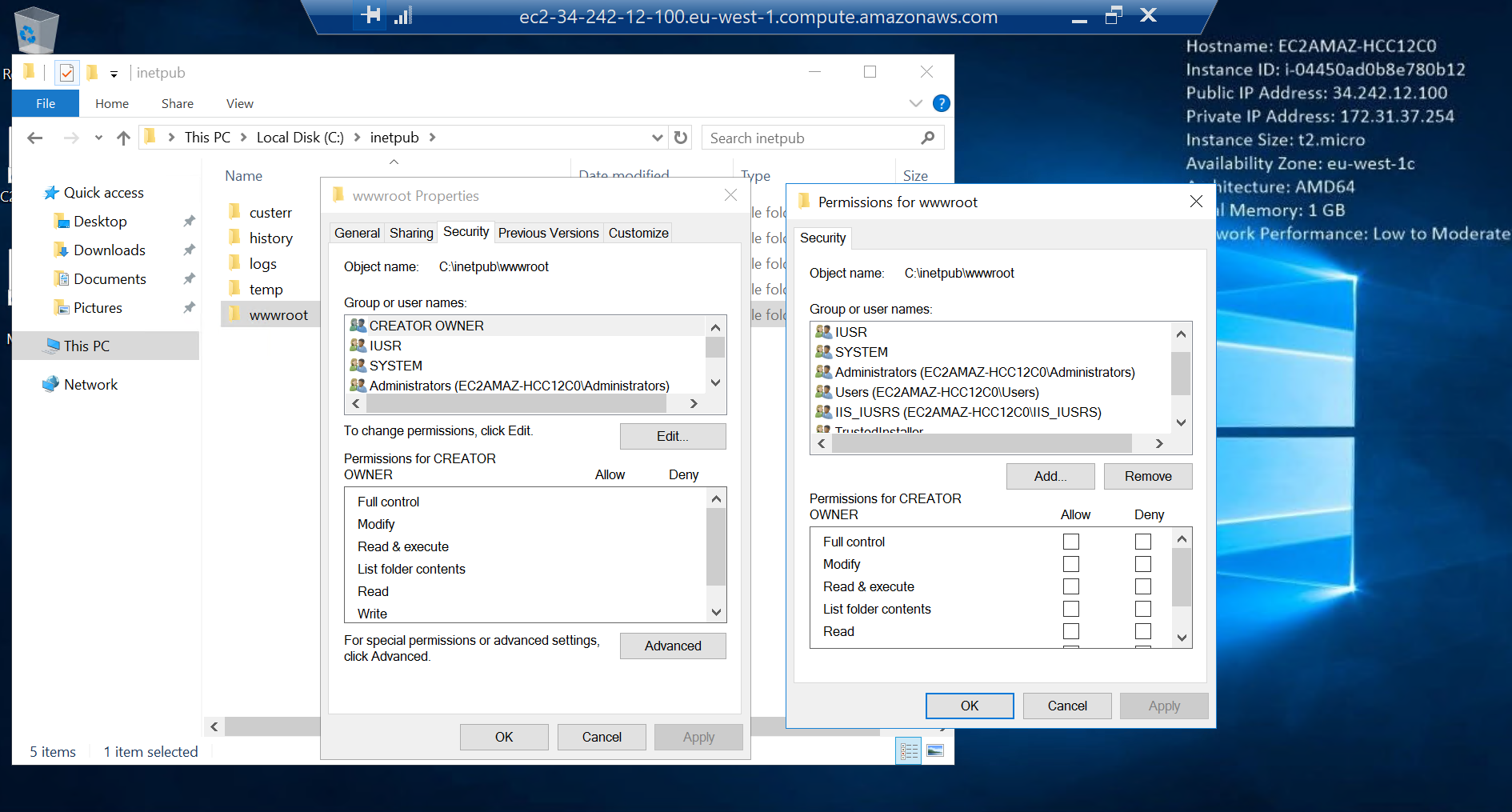


Found C:\inetpub\wwwroot folder in Remote Desktop. Changed access security settings of the following users to include ‘Write’ and ‘Modify’ access:

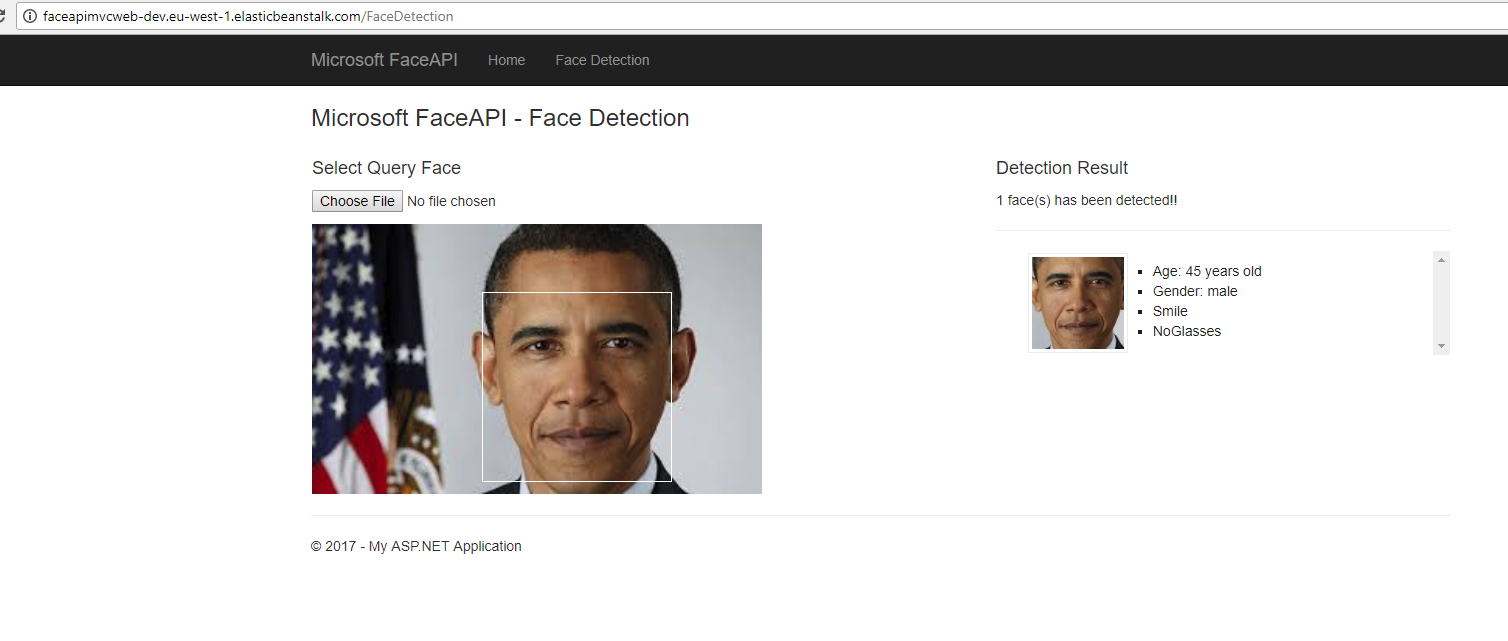
* Users (EC2AMAZ-HCC12C0\users)
* IIS\_IUSRS (EC2AMAZ-HCC12C0\IIS\_IUSRS)
* IUSR

Gave the ‘Creator Owner’ user ‘Full Control’ and ‘Modify’ rights.

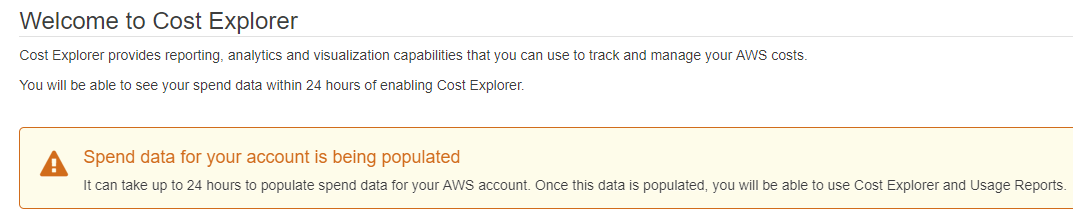
Update 02/12: It is the ‘USERS (EC2AMAZ-HCC12C0\users)’ that needs to have ‘modify’ ticked. After that, the url works without needing to relaunch the app from Visual Studio or from AWS.



Refreshed the app in Visual Studio. Clicked URL in Visual Studio, and app successfully uploaded image and identified the face within it.

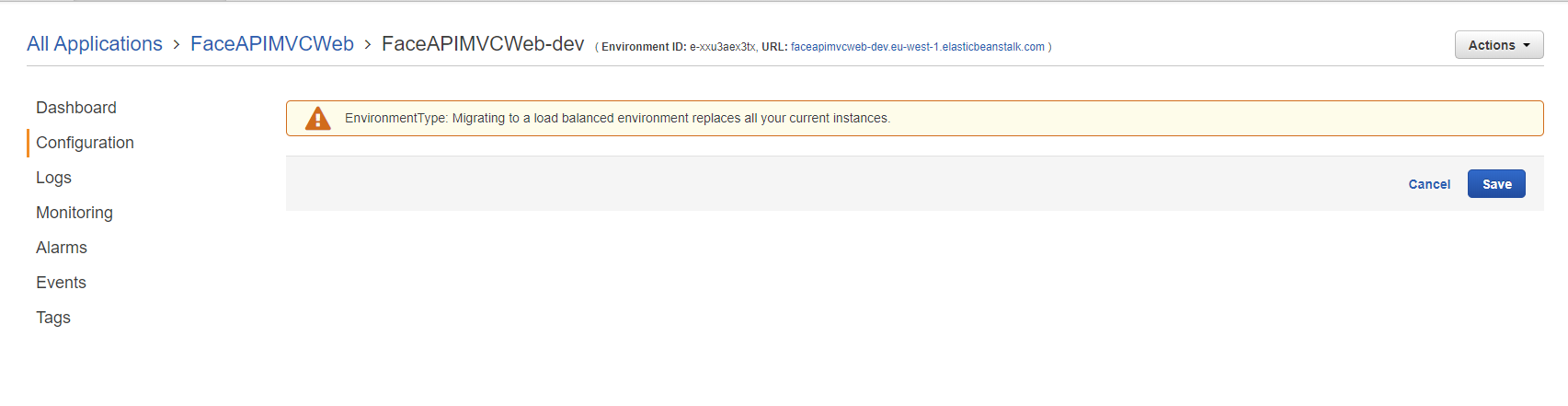


To Do:

* Make URL public so it can be shared. Does the code need to be ‘Upload and Deploy’ for this to happen? No, site is already public. Have tested it on my phone. Does it need to be stored in S3, and made public from there? Not necessary – already live. Does it still work ok?
* Make it autoscalable. In Elastic Beanstalk, Environment type from Single Instance to Load Balancing, Autoscaling. Does it still work ok?
* Move source FaceAPI files to S3 storage. They’re currently on the laptop desktop. Does the app still work ok?
  + http://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html
* Investigate how to access the S3 file storage in AWS via REST API. This will be built into the next version of the code. Not necessary due to now using Docker Hub as a storage location instead.
* Monitoring – check what AWS can offer. See if there is an AWS equivalent of Azure’s ‘Application Insights’. Azure Operational Insights = Amazon CloudTrail – can perform analytics. Azure Application Insights = Amazon CloudWatch – for monitoring of activity, logs, application performance issues, etc. Also, Billing Dashboard and the Cost Explorer.
* 
* Configuration – API Keys/storage to be changed.
* Investigate backup options available in AWS.
  + <https://aws.amazon.com/backup-recovery/>
  + <https://aws.amazon.com/glacier/> - deep/cold storage of data. For archiving and retrieval
  + <https://aws.amazon.com/s3/> - regular storage.
  + <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSSnapshots.html> - takes a point-in-time snapshot
  + <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html> - Amazon Machine Images. “An Amazon Machine Image (AMI) provides the information required to launch an instance, which is a virtual server in the cloud. You specify an AMI when you launch an instance, and you can launch as many instances from the AMI as you need. You can also launch instances from as many different AMIs as you need.”

**26/11/2017: Config changes and re-testing (done through AWS Elastic Beanstalk interface)**

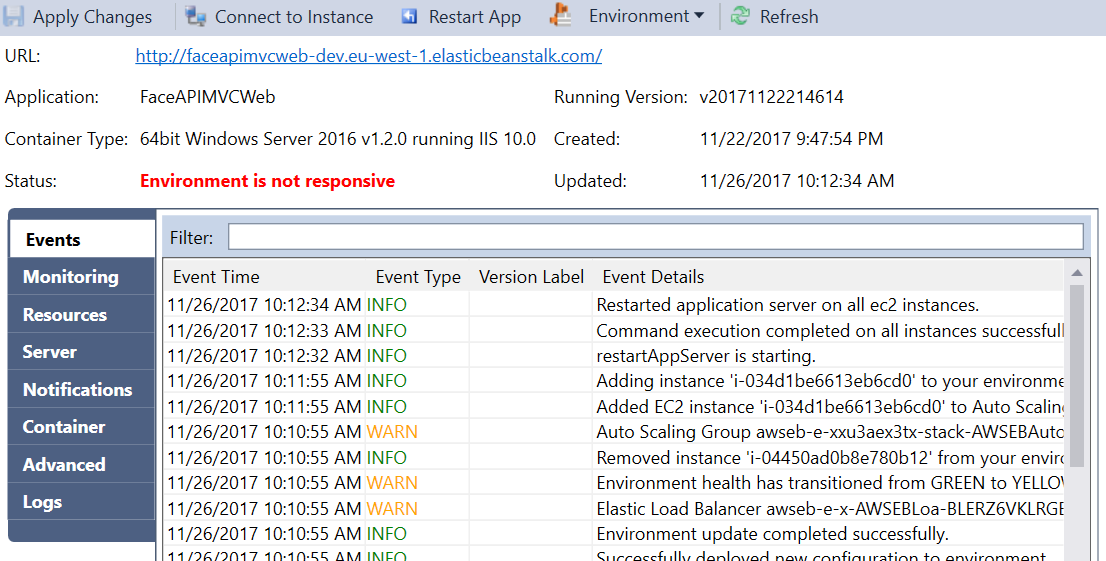
1. Scaling: Changed from Single Instance to Load Balancing, Auto Scaling.

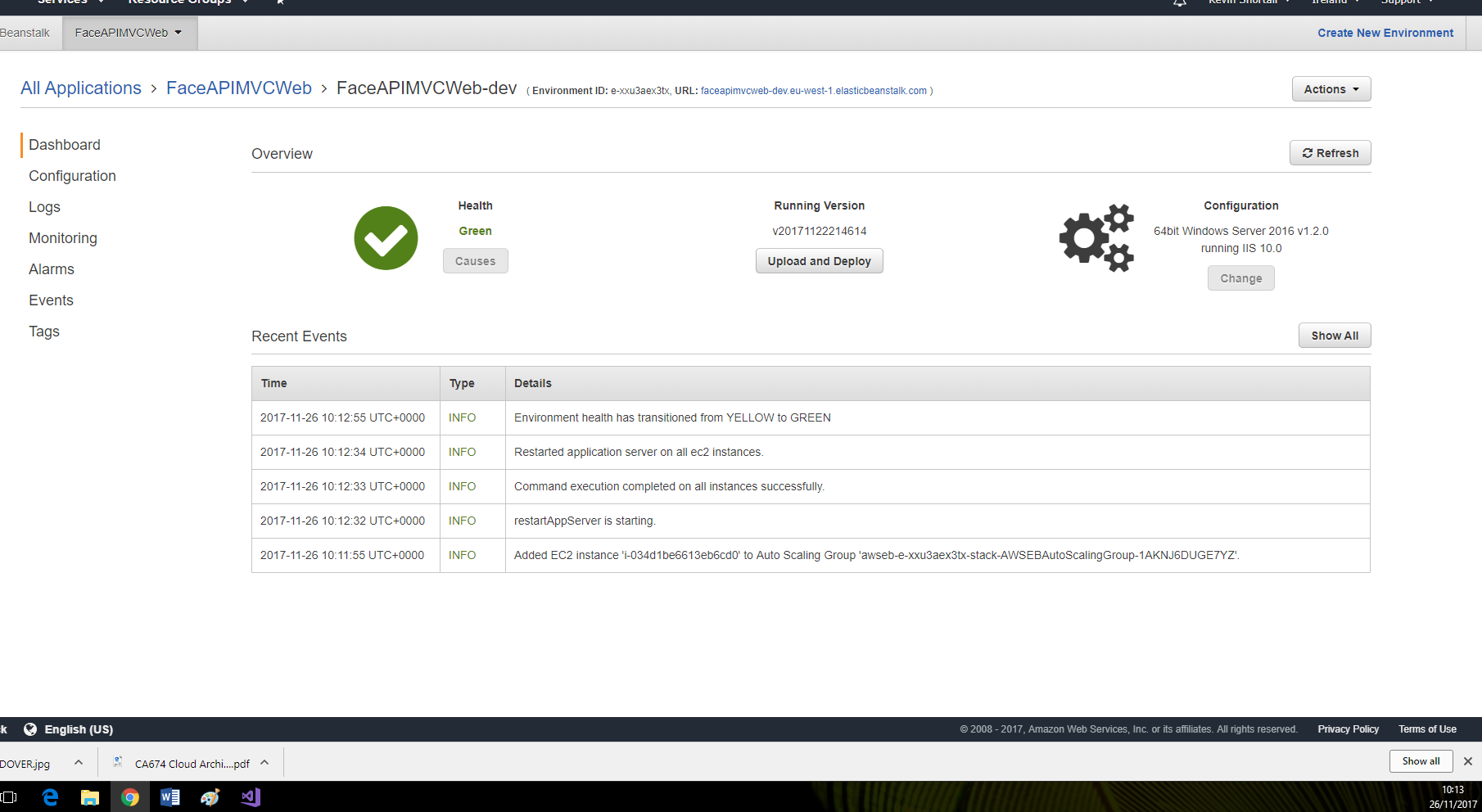


Hit Save. Takes approx. 5 minutes to update the Health Status.

Initially got ‘Yellow’ status in the GUI, and Visual Studio reported ‘Environment is not responsive’. This cleared to healthy/Green status after I hit ‘Restart App’ button in Visual Studio.

Configuring to Autoscale – what is the impact to the cost to the user? Still covered by free usage? Or depends how much you use it?





1. Instances. Left it at t2.micro.

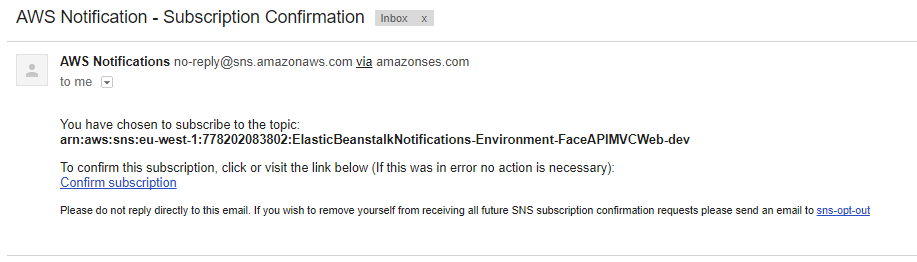
From the Amazon website: “the t2 family of instances provides a moderate amount of power with the ability to burst for short periods of time.”

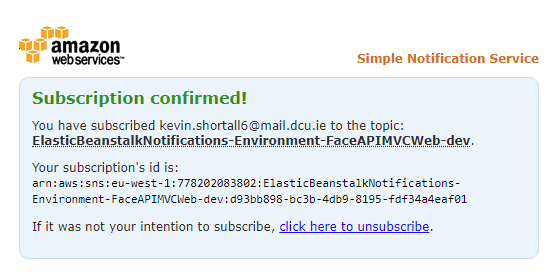
Instances information: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-types.html>

Amazon has a variety of instances, from the entry-level to the advanced computing requirements. As this account was created on the AWS Free Tier, usage of t2.micro instances is allowed for free up to 750 hours per month. Therefore we stuck with this instance. It will also require all the computational requirements we need for this project.



1. Notifications. Email address entered to receive alerts/information on web app changes.





1. Software Configuration:

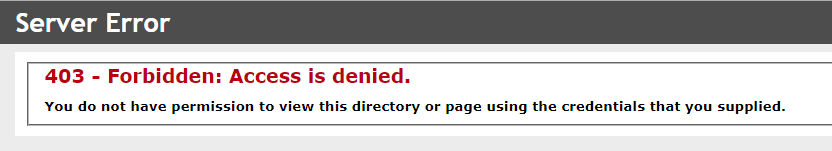
Container Options: Not changed. Left as 4.0 Target .NET runtime, and False for Enable 32-bit applications.

AWS X-Ray enabled. Free tier usage.

<https://aws.amazon.com/xray/>

Log Options: Switched on Enable log file rotation to Amazon s3. If checked, service logs are published to S3.

Took a long time to refresh. Error showing in Visual Studio, even though status is given as healthy. Clicked link in AWS and got ‘403 – Forbidden: Access is denied’.



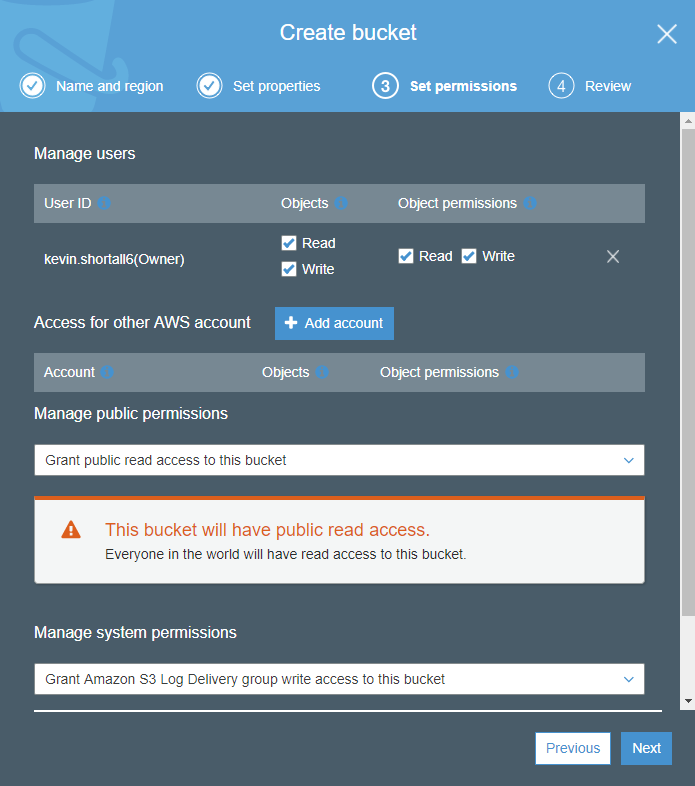
Restarted App in Visual Studio. No improvements.

Rolled back Software Configuration changes above. Removed X-Ray and Logging to S3. Restarted App. No improvements.

Removed email address from notifications. No improvements.

Changed from Load Balancing, Auto Scaling to Single Instance.

When creating an S3 bucket, AWS is security-conscious and prompts you to keep things non-public.



* FaceApiImages S3 bucket:

<https://s3.console.aws.amazon.com/s3/buckets/faceapiimages/?region=eu-west-1&tab=overview>

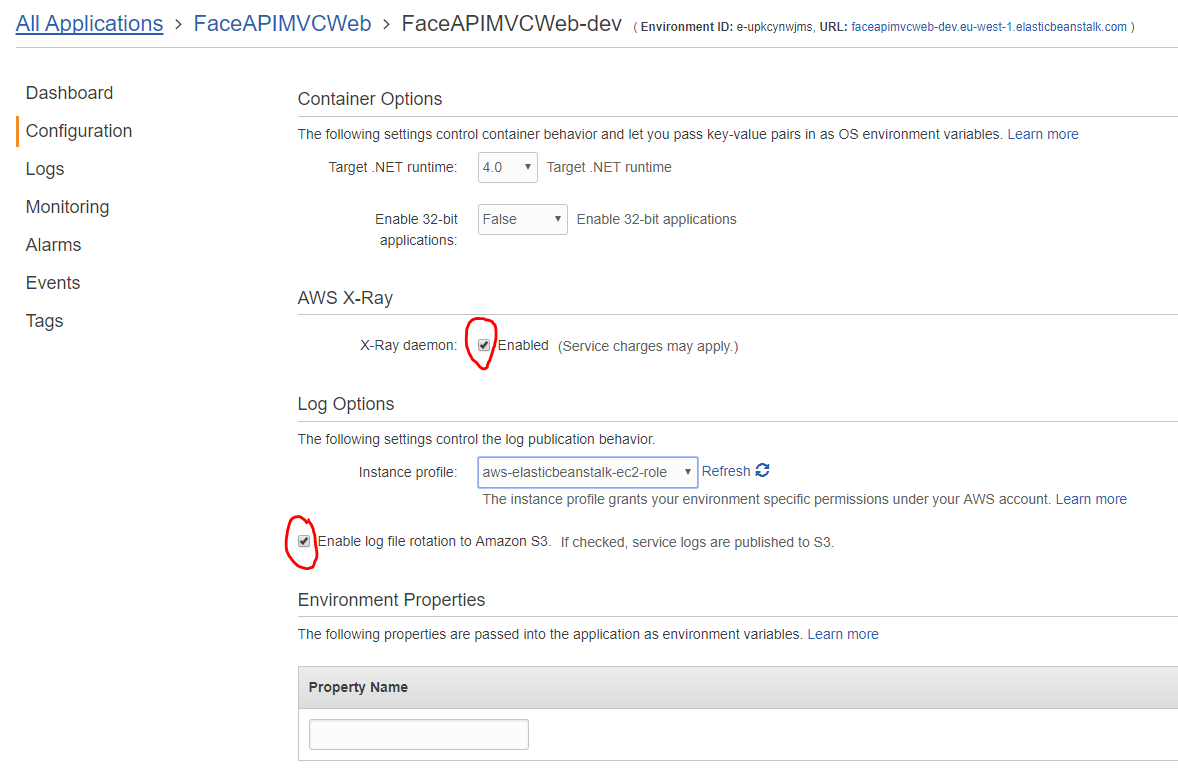
* Face API loaded from a stored location on Google Drive – is that inter-operability??

## 26/11/2017: 2nd attempt at changing the configuration settings:

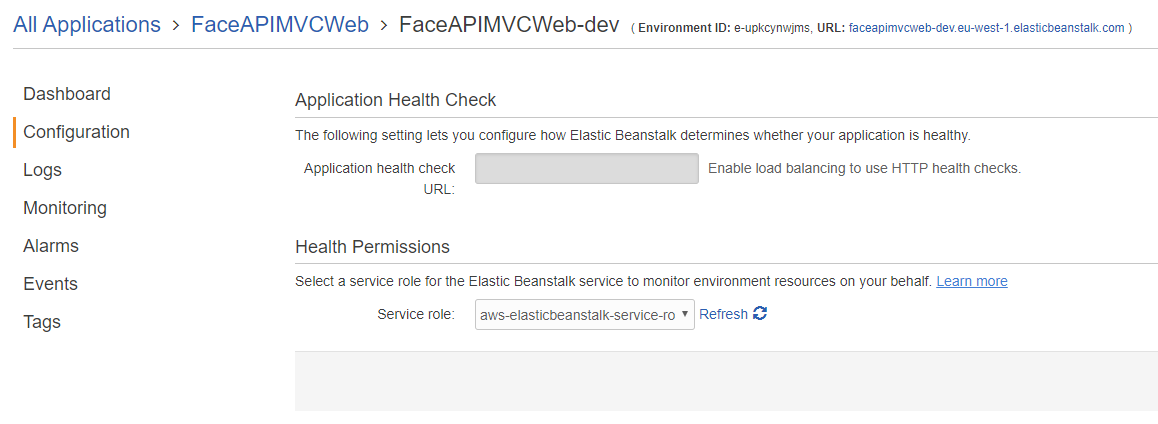
1. Scaling: leaving that change until last. Gives warnings. Will need to change the max #instances – currently max/min both at 1 by default for single instance.

<http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features-managing-env-types.html>

1. Instances: left it as t2.micro.
2. Notifications: Entered my dcu email address and confirmed by clicking the link in the received email.
3. Software Configuration: Enabled X-Ray daemon and logs published to S3. App updated and still working fine.

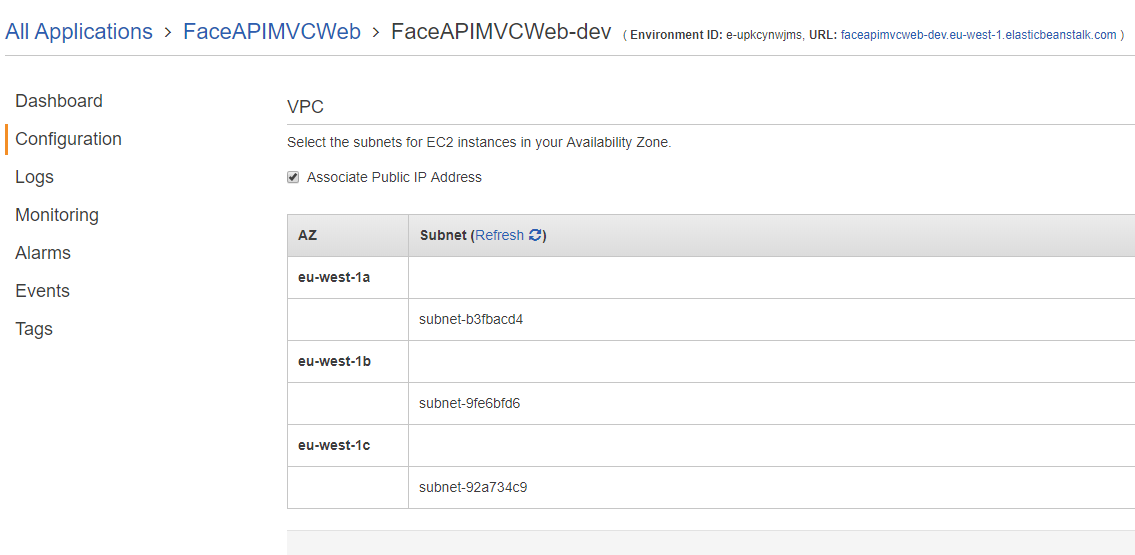


1. Updates and Deployments: No changes necessary and none that made sense to trial.
2. Health: No changes required or possible. Health check url was greyed out and there was only one Service role option. No changes made.



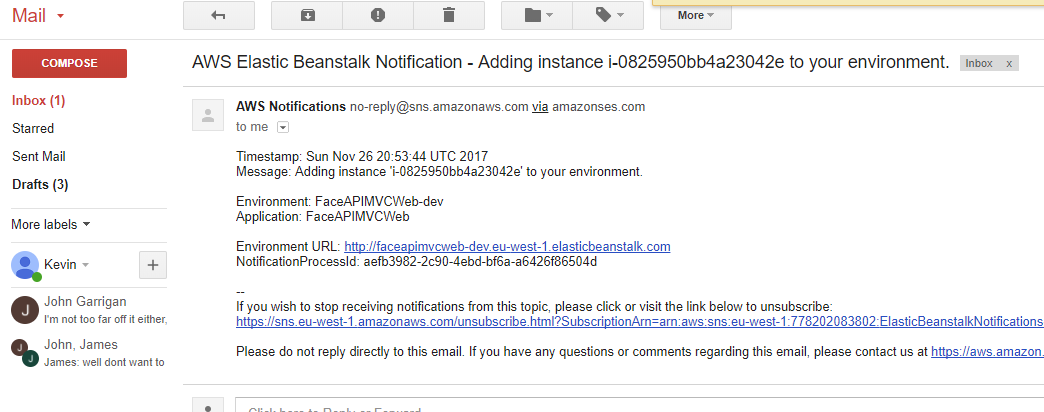
<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/concepts-roles.html?icmpid=docs_elasticbeanstalk_console>

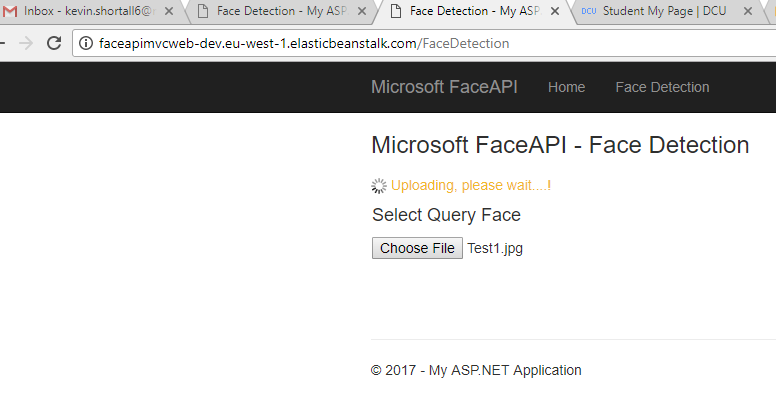
1. Network Tier: Ticked box for ‘Associate Public IP Address’ as it was recommended: “When one subnet is used for both EC2 and ELB it is suggested that you check the "Associate Public IP Address" checkbox unless you have set up a NAT instance to route traffic from the Internet to your ELB subnet.”



URL did not work after update.

Email notification of one added instance to my environment, and another email for one removed instance from my environment. Both links open the app url but hang on the ‘Uploading, please wait…!’ page.





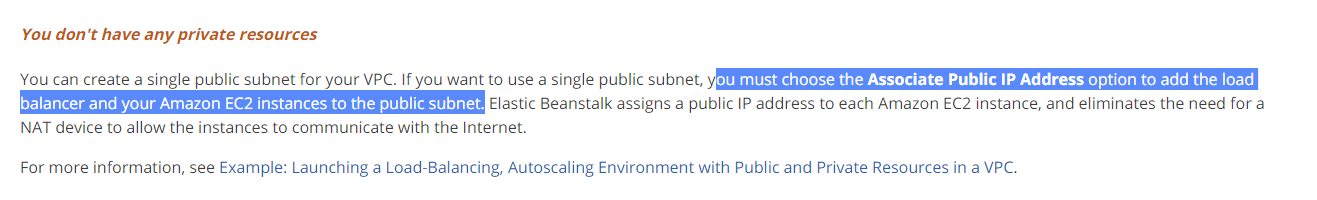
The old instance has been terminated. The new instance replaces it. Therefore need to log into remote desktop of new instance (i.e. connect to it), use same key pair file as previously, and change permissions in the wwwrooot folder as before.

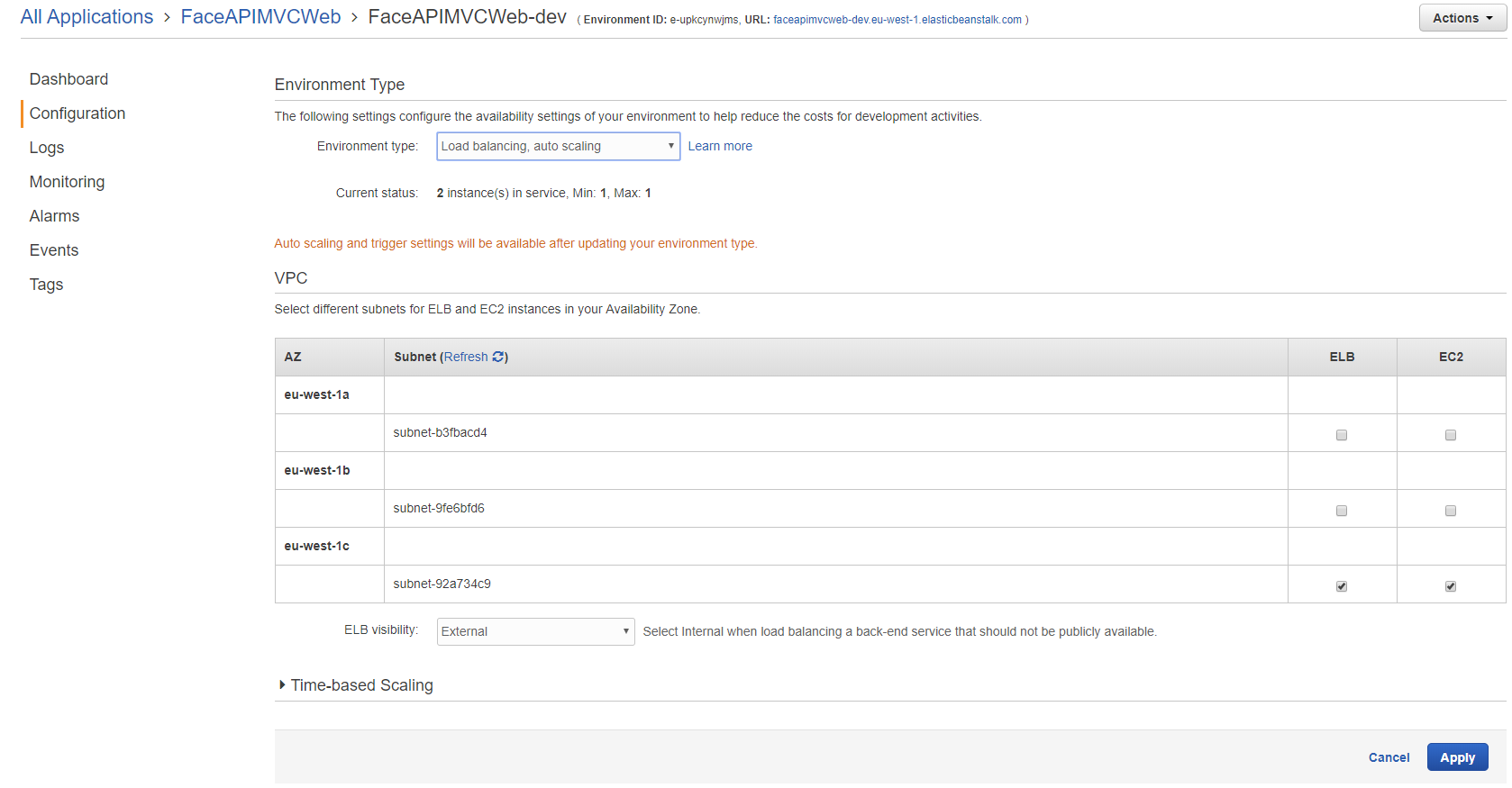
Changed IUSR and IIS\_USRS. Gave full app access immediately. Rolled back the IUSR permissions to just ‘read’ -> app still worked fine. Rolled back the IIS\_IUSR permissions to just ‘read’ -> app still worked fine. Not sure what’s caused this.



SCALING: Set to Load Balancing, auto scaling.

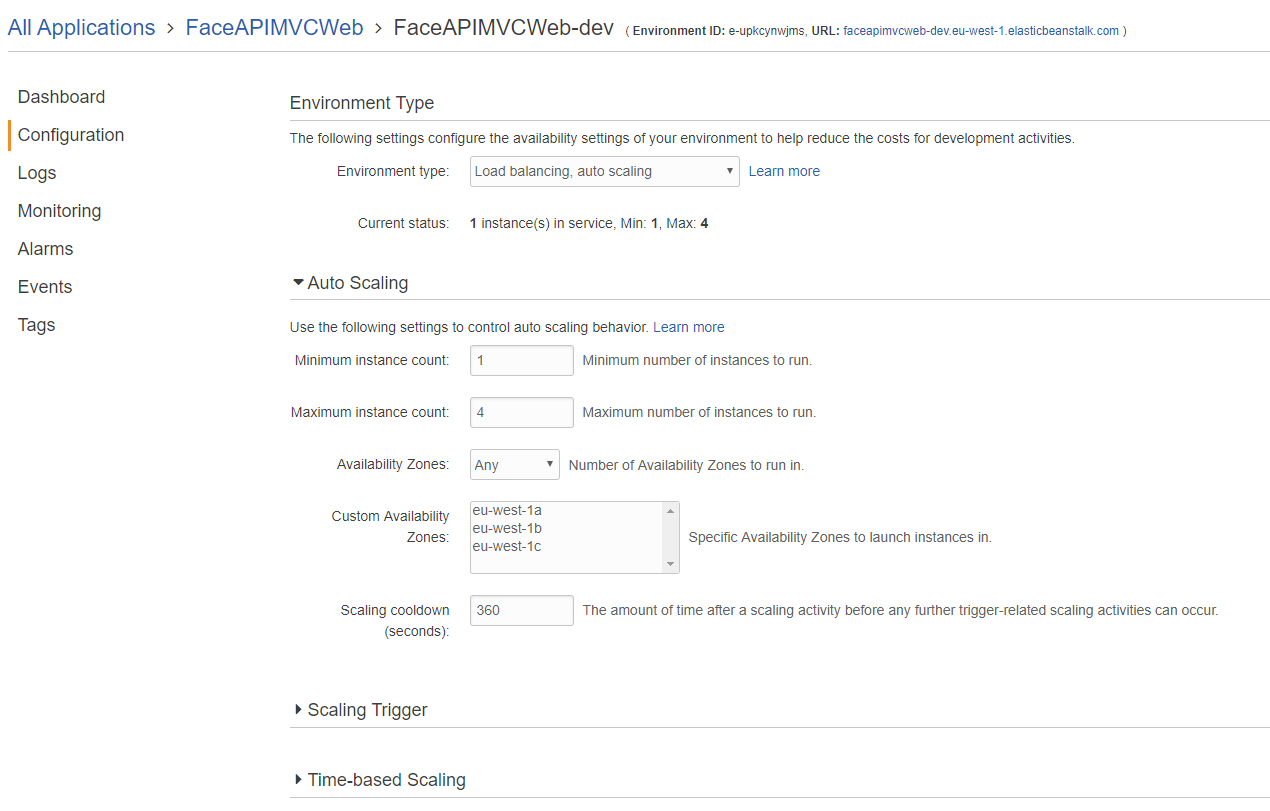
<http://docs.aws.amazon.com/elasticbeanstalk/latest/dg/vpc.html>



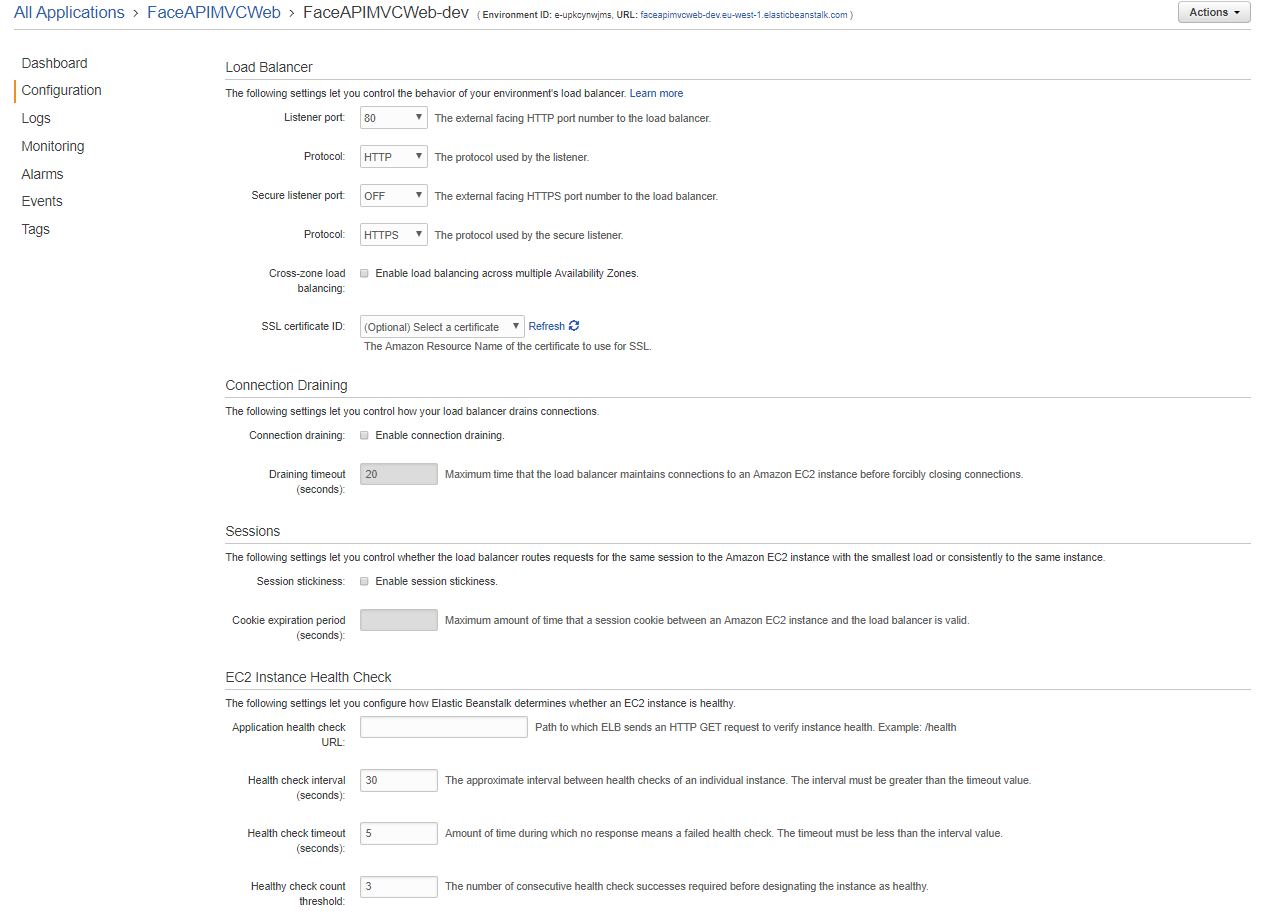


Warning that this changes all existing instances.

When it loaded successfully, the url still worked fully. I checked in the Configuration/Scaling area, and the Auto Scaling information had been implemented – Min=1, Max=4. I think it worked this time (and not last time) because I had the box ticked in the Configuration/VPC section for ‘Associate Public IP Address.



New Configuration/Load Balancing section has also been created. I did not change any settings here, they are all default settings.

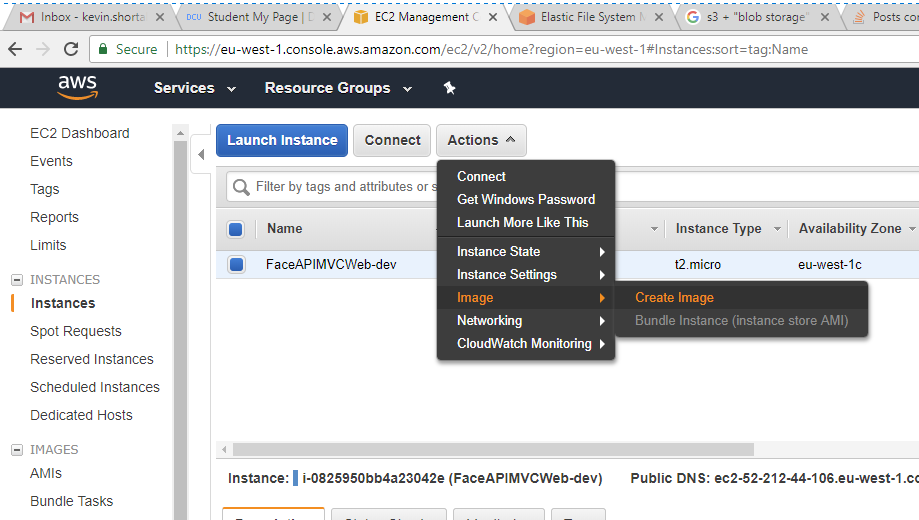


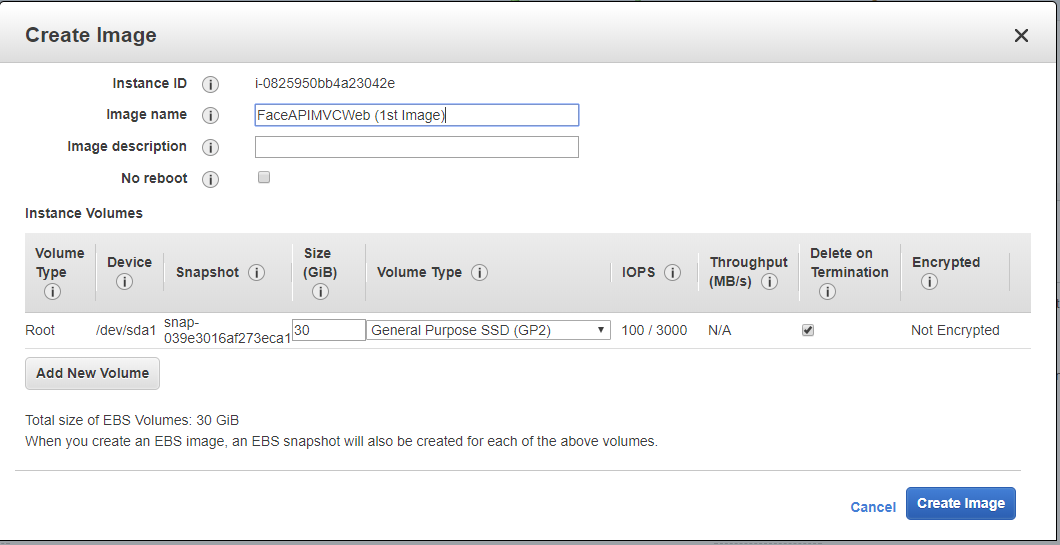
**Microsoft Azure Blob Storage v Amazon Simple Storage Service (S3):**

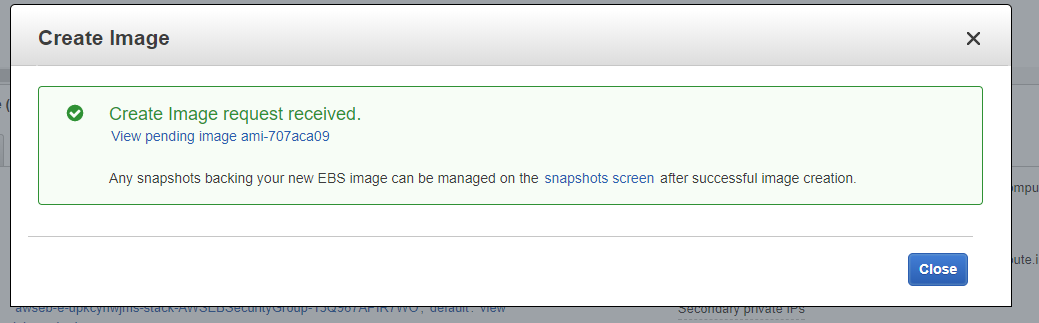
<http://gauravmantri.com/2012/05/13/comparing-windows-azure-blob-storage-and-amazon-simple-storage-service-s3summary/>

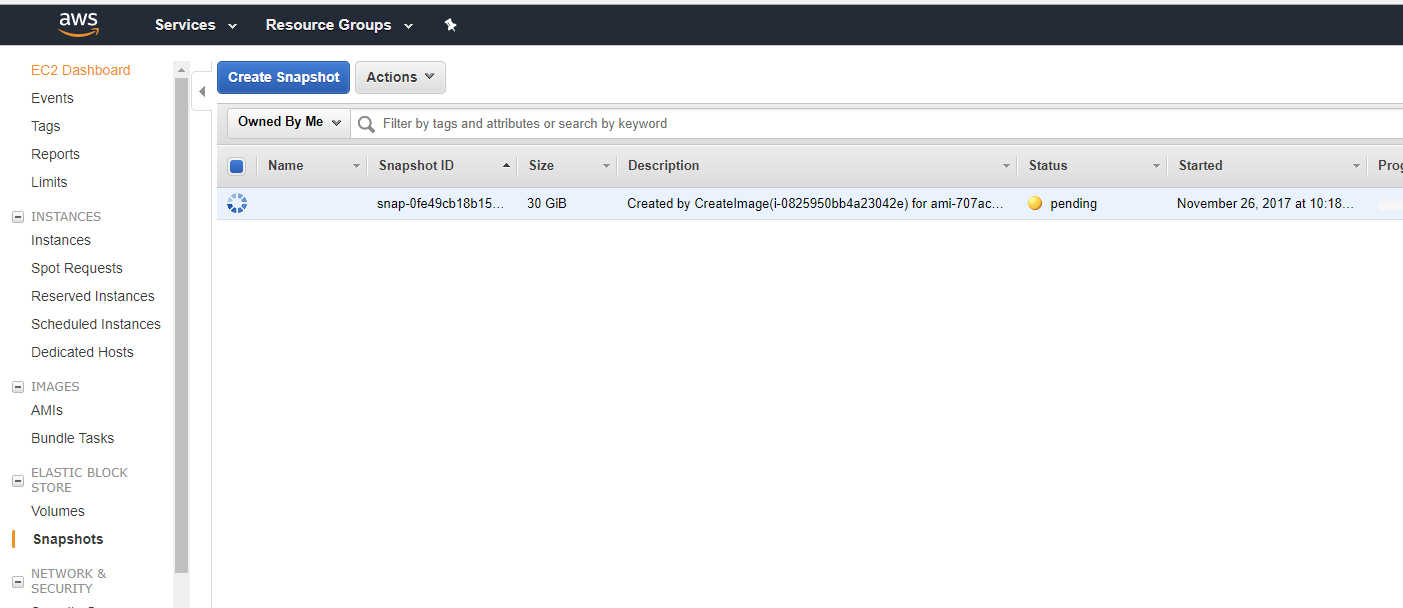
CREATE IMAGE:

EC2 -> Instances -> Actions -> Image -> Create Image









28/11/2017:

Autoscaling code works, bring in all images. Initial autoscaling setting shown below. To be altered and monitored.

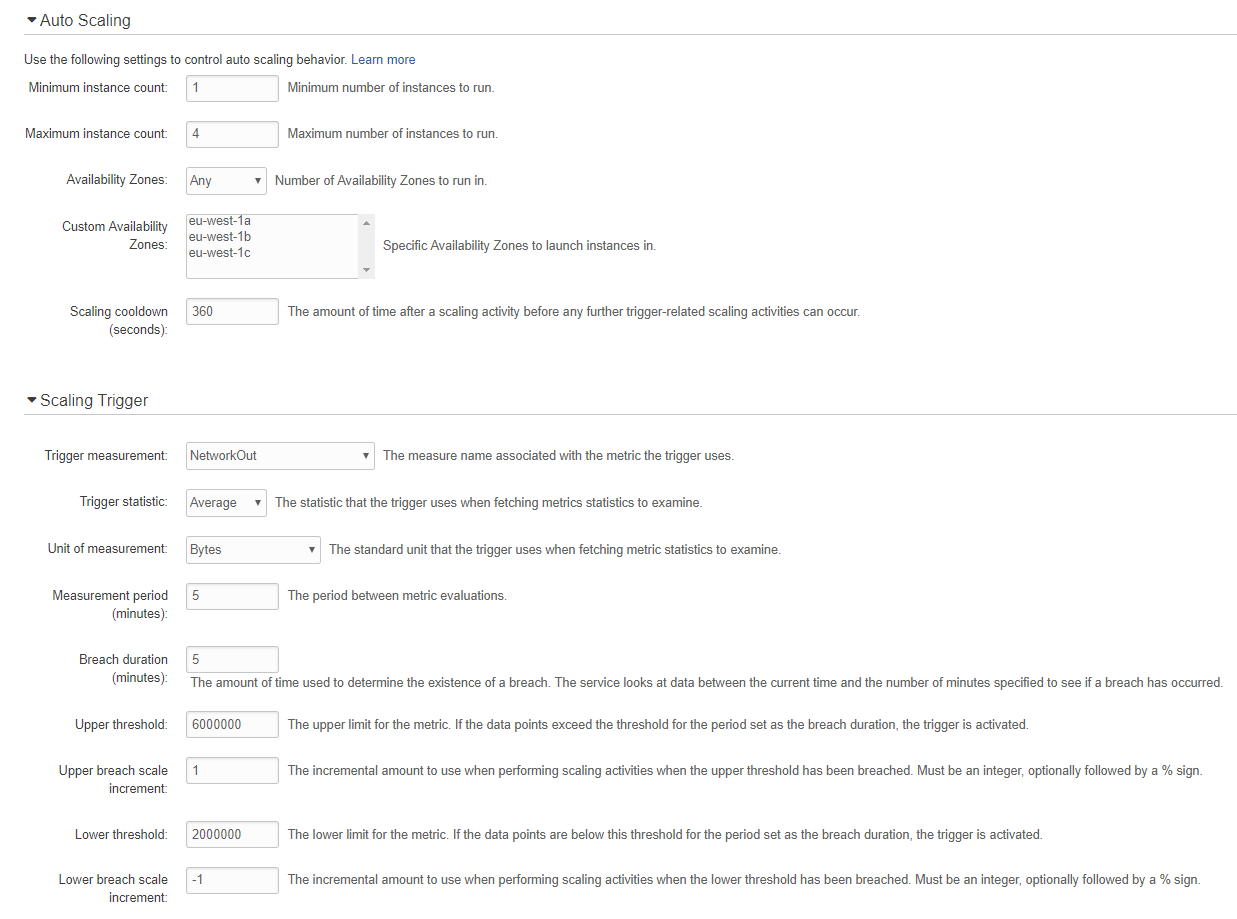
Images showing in S3, put there by Javed.

CloudWatch: Metrics. All > EC2 > By Auto Scaling Group. Select



**To verify Autoscaling works:**

<http://docs.aws.amazon.com/autoscaling/latest/userguide/as-instance-monitoring.html>



**Video approach:**

Begin with Visual Studio open. Explain basic layout and show how the app looks in Visual Studio once you double-click it in Windows Explorer.

Hit F5 to run it locally and demonstrate both basic Face App and MultiFace App.

Deploy the app to Elastic Beanstalk from Visual Studio. Have AWS open already, and show the status changing to updating in both VS and AWS.

Once deployed, show that the url brings you the webpage, from both VS and AWS.

Show CloudWatch and the graphs showing the increased volume after deploying, and demonstrate that Autoscaling was utilised, if possible.

Show where the S3 storage of the photos is, and where in VS the code points to it.

Show within AWS where the Elastic Beanstalk is shown, where the Instances, Key Pairs

02/12: Things that need to be included in the report:

The fact that permissions were automatically not given to view the url. I had to open a remote desktop, load a key-pair (not obvious to first-time users), find a particular folder, and allow ‘write’ permissions for a user, before I could open and work the url. Not user/beginner-friendly.