Blog for CA674 Cloud Architecture project

Team Members:

* Javedali Shaikh
* Sanjay Singh
* Kevin Shortall

Please note that this github blog was created on 15th November 2017. Up until this date, progress was tracked in a Google Sheet location in Google Drive, at the link shown below. The first entries shown in this blog come from that Google Sheet.

**Google Sheet location:**

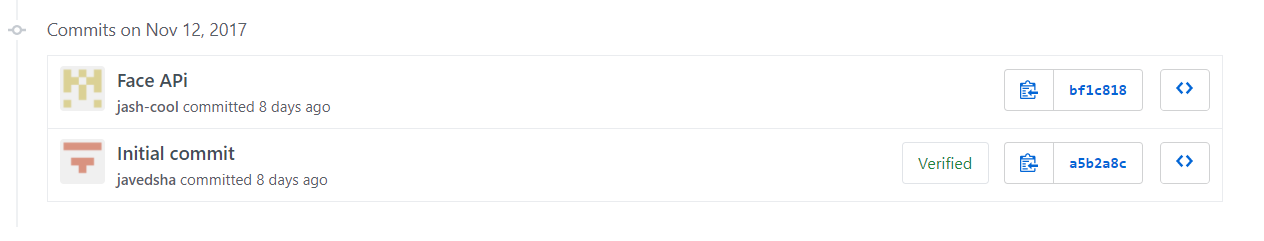
<https://docs.google.com/spreadsheets/d/1fGVU8-_ZSk_-E7G1xKPsJQ1M4j3zhTc23c0DhKyYrhk/edit#gid=0>

**Original Google Sheet entries:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Student(s)** | **Topic** | **Detail** |
| 27/9/2017 | All | Group formation | Initial group formed of three students. |
| 29/9/2017 | Kevin | Project Topic | Initial project topic regarding the storage of photos on various cloud servers sent to the group via email. |
| 1/10/2017 | Sanjay | Project Topic | Additional project topic suggestions regarding various versions of SQL accessing docker locations on different cloud platforms. |
| 6/10/2017 | Kevin | Project Group | Project group of 3 students (Sanjay, Javed, Kevin) confirmed to lecturer. |
| 9/10/2017 | All | Meeting | Group meeting in Business School of DCU to brainstorm project ideas. Discussed possibilities of using DockerHub for code storage locations and accessing it from AWS and Azure. Also examining different storage types, such as table, file, blob, etc. |
| 11/10/2017 | All | Project Topic | Email sent by Kevin to lecturer, after discussion with the whole group, regarding some initial potential project topics, as well as seeking clarification on the scope required. Response received on the same day which ruled out some of the smaller-scoped project ideas. |
| 17/10/2017 | Javed | Project Topic | Suggestion to use face recognition application, fetching images from storage on the cloud and using machine learning services. Other suggestions of IoT Hub (connect and monitor IoT devices) and Develop Micro-services and orchestrate containers. |
| 18/10/2017 | Sanjay | Project | Whatsapp group created for the project. |
| 18/10/2017 | All | Skype Call | Discussed various project options. Decided upon running face recognition app on a hosted website using three different sources - Azure, AWS and OpenShift. |
| 19/10/2017 | Kevin | Project Topic | Confirmation that AWS is more suitable than Google due to Google has a face detection API, but not a face recognition API. |
| 20/10/2017 | Kevin | Project Proposal | Proposal file uploaded to Loop for review by the lecturer. |
| 28/10/2017 | Sanjay | Project Report | Draft report suggestion circulated to the group containing suggested headings and content. |
| 29/10/2017 | Javed | Website format | Request that ability of hosting asp.net websites is required of all three cloud services. Confirmed by all students. |
| 30/10/2017 | All | Skype Call | Confirmation that all three infrastructures can support asp.net websites. |
| 30/10/2017 | Javed | Skype Call | Website code sourced. Code to be refined and submitted to github as the central repository. |
| 08/11/2017 | Kevin | Blog | Google Spreadsheet created and shared to be used as a Blog location for the team. |
| 12/11/2017 | Javed | Web code | Initial website code for face recognition api shared with the group via Github. |
| 13/11/2017 | All | Project Meeting | Demonstration of web code developed so far and it's interaction with Microsoft Azure. Clarification of the next steps to incorporate that code into AWS and Open Shift cloud platforms. Layout of the remainder of the project approximately as follows: 1. Hosting  2. Configuring  3. Storage  4. Scaling |
| 13/11/2017 | Kevin | Visual Studio | Visual Studio 2017 Community Edition installed. When trying to open the FaceAPI\_MVC.sln file, the following feature was required to be installed in Visual Studio: ..\FaceAPI\_MVC.Web\FaceAPI\_MVC.Web.csproj |
| 15/11/2017 | Kevin/Sanjay | Meeting | Meeting in Business School in DCU. Review of recent progress with regards to hosting website on AWS and OpenShift. Demonstration of VM Machine used by Sanjay to access Cloud areas. |
| 15/11/2017 | Sanjay | Minishift | MiniShift setup on Windows -  Installed Virtual Box 5.1.3  minishift-1.6.0-windows-amd64  Just tried how to start and stop Virtual Box using cmd command and setup to access Openshift using local IP - <https://192.168.99.100:8443/console/> |

# 12 November 2017 - Javed

Face API, an ASP.NET based website created and commited to GITHUB. This was our first github commit. Microsoft Azure face API service is used in the backend (Software as a Service)



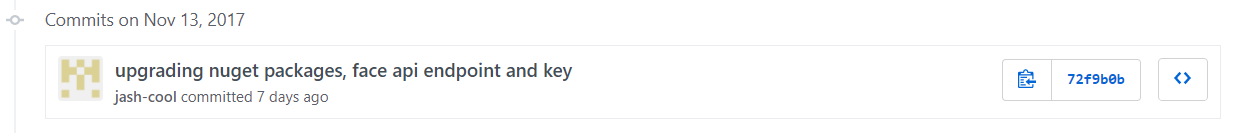
The website has one major functionality added at this moment – detecting faces from an image uploaded through user’s local hard drive. The website code was taken from [here](https://code.msdn.microsoft.com/Face-API-Using-ASPNet-MVC-40259a76) and modified to suite our requirements.

# 13 November 2017 - Javed

Created new Face API account in Microsoft azure to get our free trial Face API endpoint.

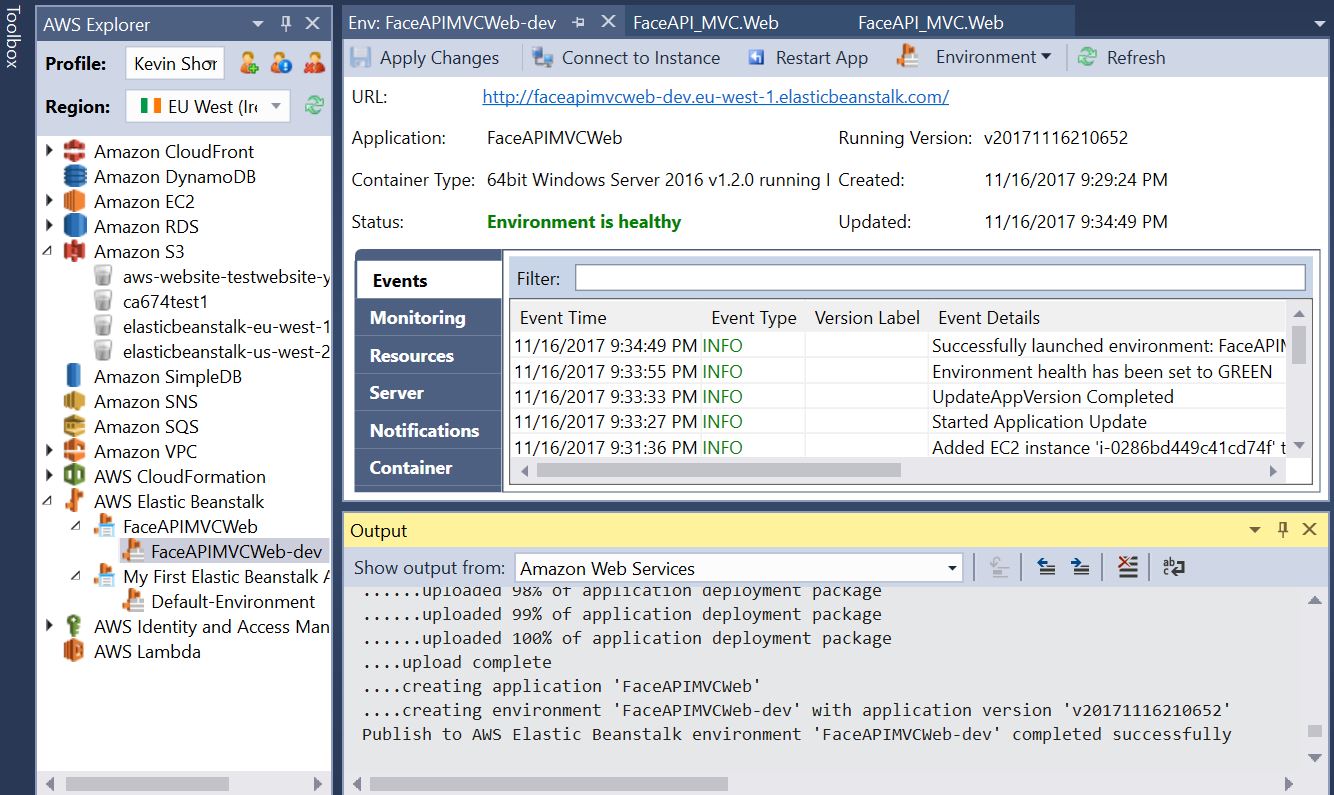
Updated the Face API website with the new face API endpoint and also updated the code to point to right location of Face API endpoint (<https://westcentralus.api.cognitive.microsoft.com/face/v1.0>)

Lastly, code was updated to refer latest nuget packages (binaries) for Face API.



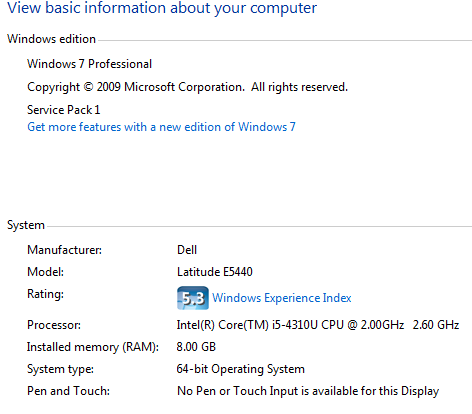
# 16/11/2017 - Kevin

Web app successfully deployed to AWS via the AWS Toolkit in Visual Studio 2017. The URL successfully brings the user to a browser window which shows the web application interface.

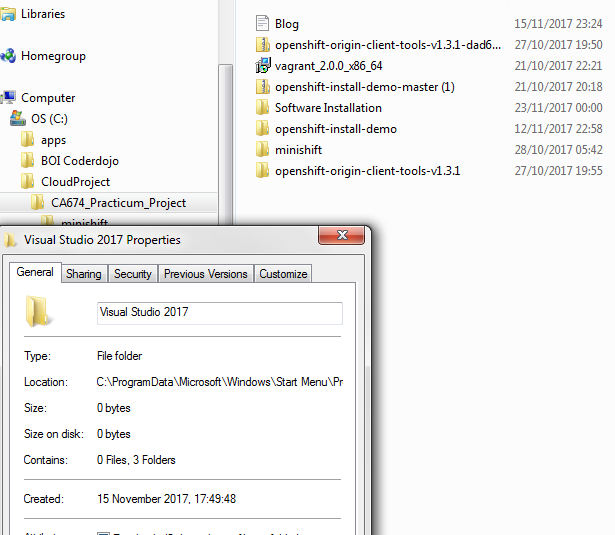


# 16/11/2017 – Sanjay

Before downloading Visual studio community version 2017, analyse laptop spec to check if the laptop current spec support…e.g. Operating System Window 7, RAM- 8.00GB, System type 64bit and installed Visual studio community version 2017.



When trying to open FaceAPI\_MVC.sln file, getting some issue so installed .Net Core 2.0.0 SDK. Checked FacApi site was successfully launched in local machine.



# 17/11/2017 – Kevin

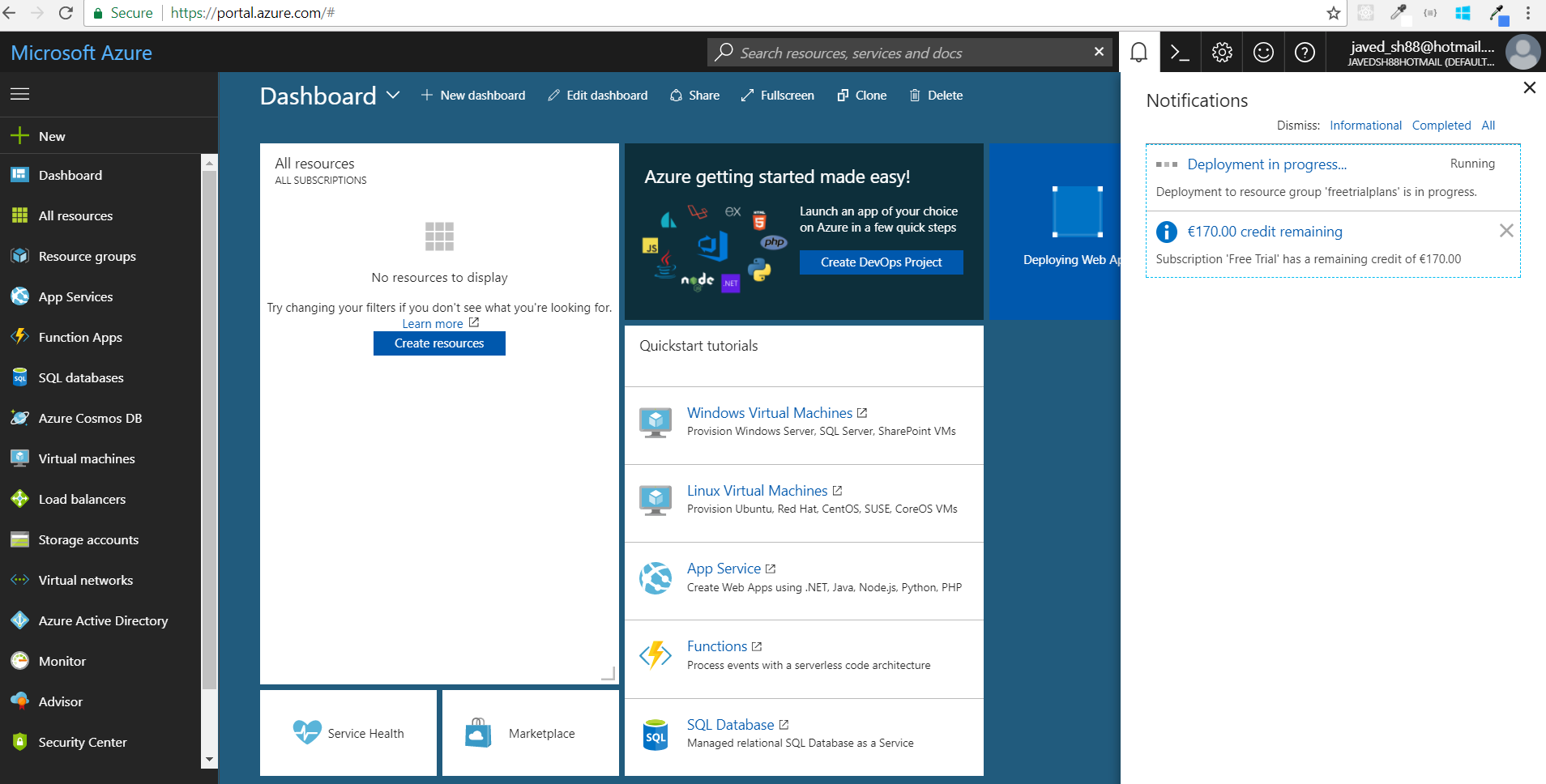
Difficulty found when trying to upload an image from a local source to the Face API webpage. I have accessed the url from different devices, so the page is live, but there is a disconnect with retrieving the images. I am still working on a solution to this.

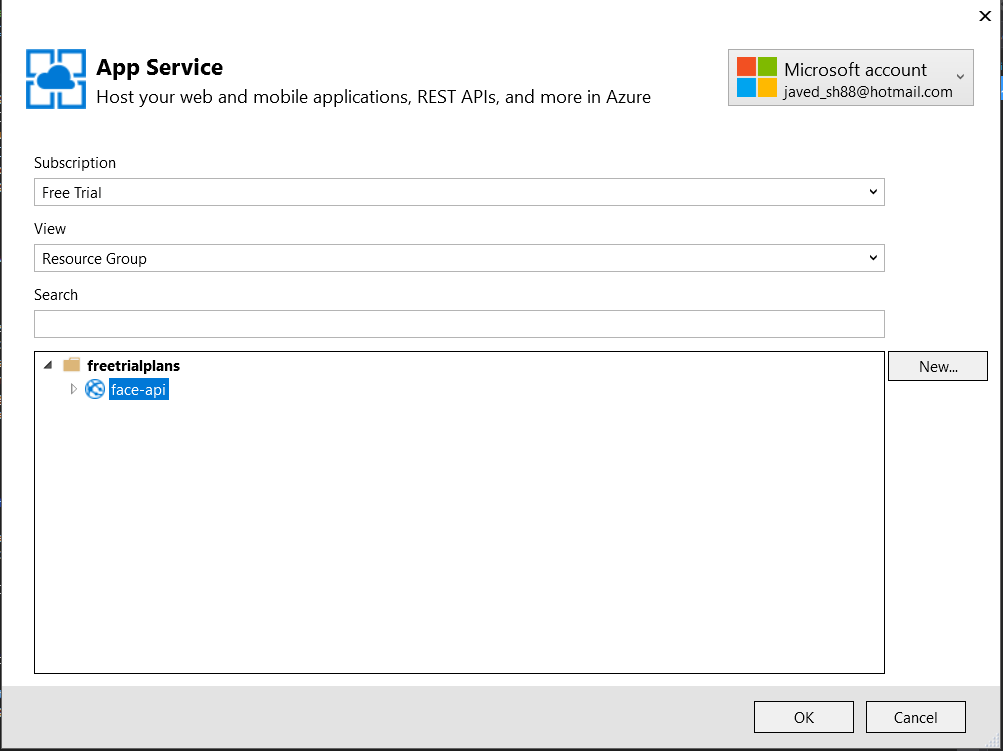
# 18 November 2017 – Javed

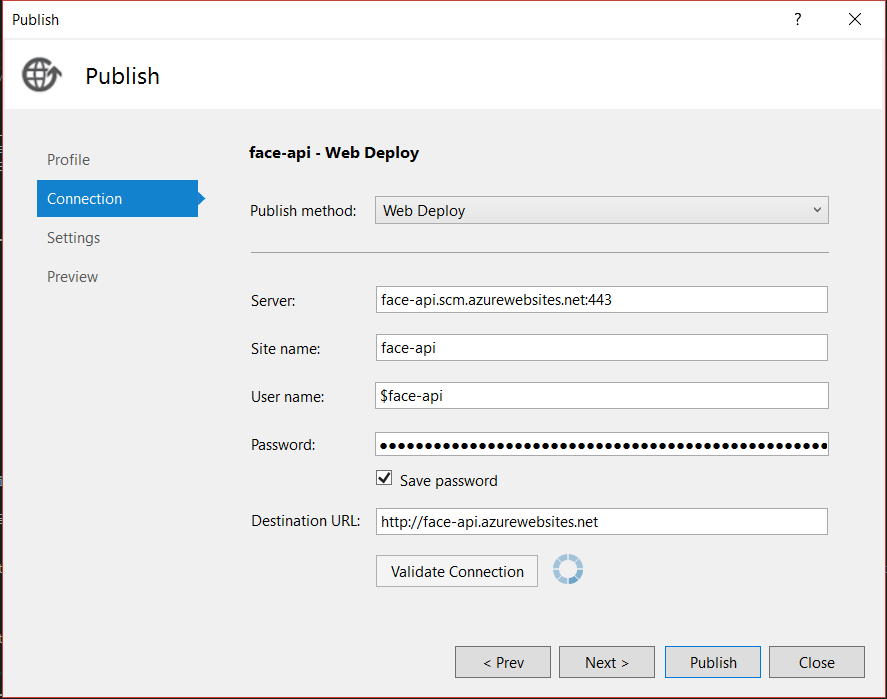
Created new Microsoft Azure Subscription (Free trial). It is valid for 1 year and has $200 credit.

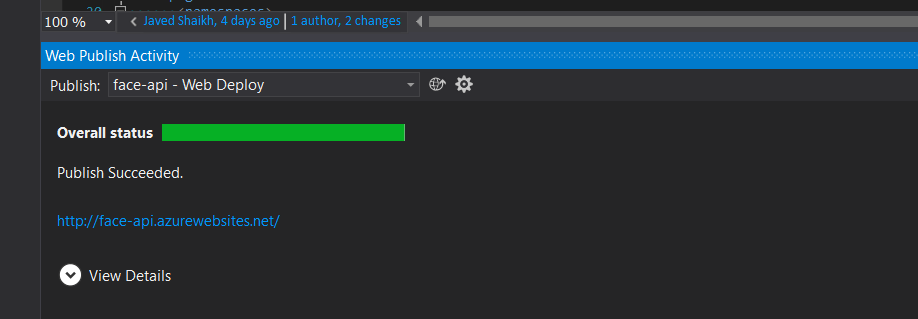
Deployed the Face API website to Microsoft Azure (Web Apps). It is hosted as a website in Azure using S3 (standard tier). Machine configurations are: 1 Core, 1.75 GB RAM and 50GB internal storage. Cost of the machine is $40/month (roughly depend on the numbers of hours). So balance money is $160.

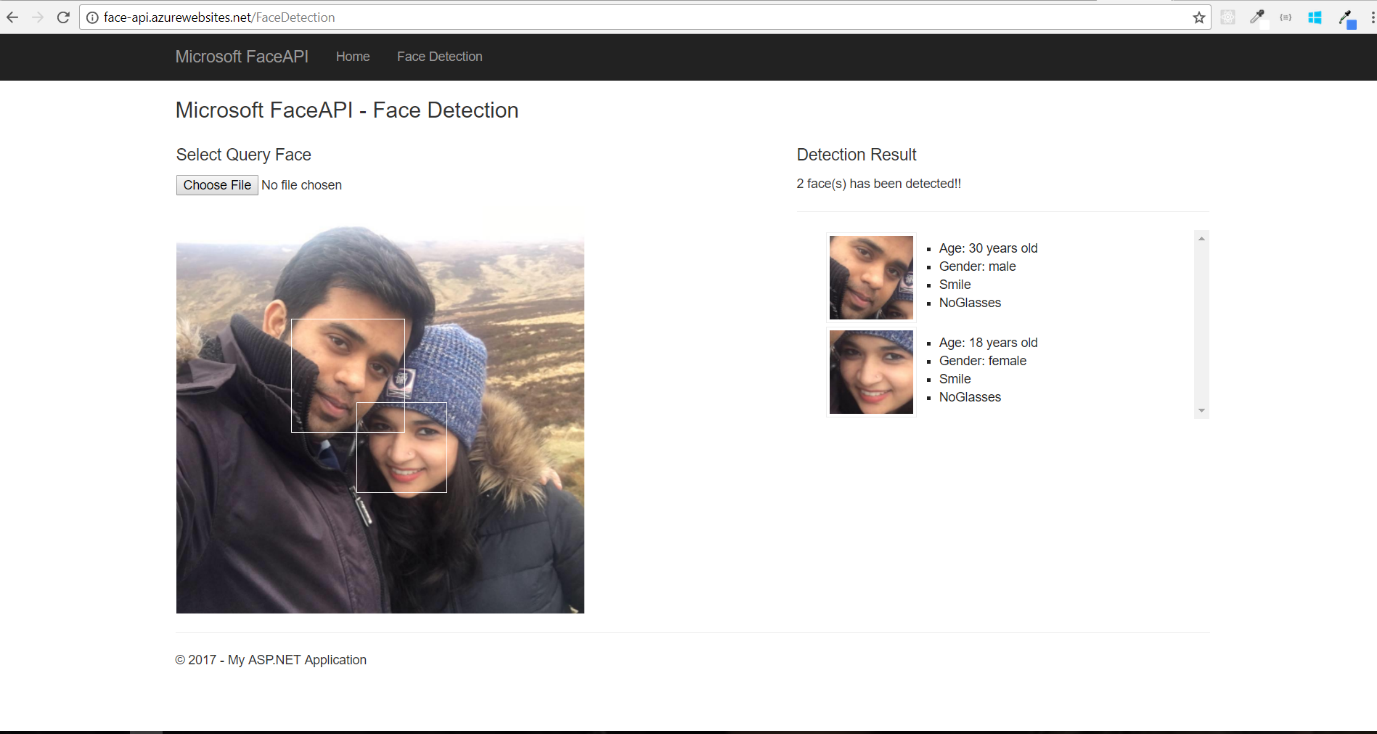
Below are the main screen shot for the process of deploying the website to Azure. Tools used are Visual Studio 2015 and Cloud Explorer.

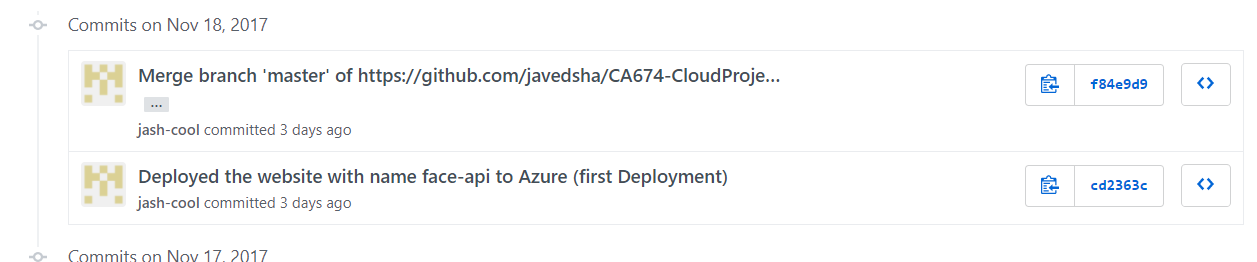












# 19 November 2017 – Javed

Created plan on ‘How to use Storage aspects of Cloud’. Discussed with Sanjay and Kevin and.

There are two options, which we discussed:

1. Creating a storage contract (IStorage.cs) and implementing three different solutions for Azure, AWS and Open Shift.
2. Investigate if all three cloud storages has REST API to access the storage (READ/WRITE). If yes, then we can develop one single component which can interact with any storage layer as long as it follows REST protocols which is quite standard. We are planning to store the different REST endpoints in web.config so that we can change it in each cloud provider.

Looks like option #2 is the best approach. Everyone will investigate and after that we will make a decision.

# 19 November 2017 – Sanjay

Setup account in Open Shift and checked price plan. There are two types of plan in Open Shift-

Stater – Free for individual learning and experimenting amd limited to 1 Project with the following features.

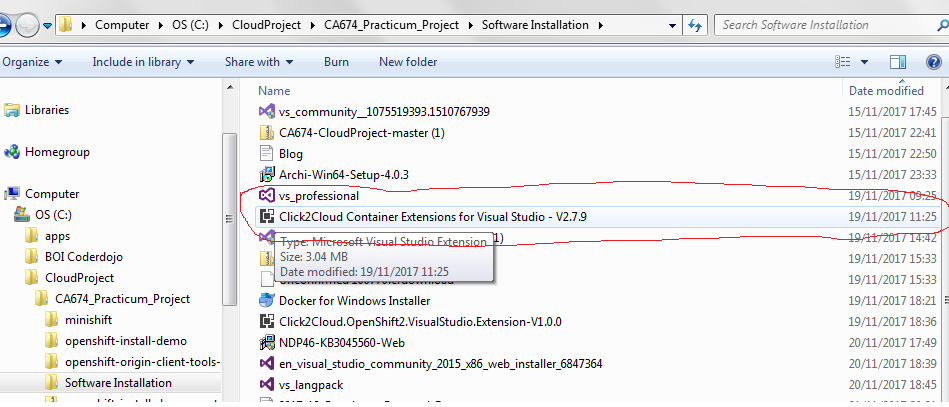
* 1 GB Memory included with 1GB terminating resources quota applies for pods with an active deadline for Build, Config and deployment.
* 1 GB storage included.

Pro – For Professional projects and hosting which supports up to 10 Projects with the following features.

* 2 GB memory included and up to 48GB memory available
* 2 GB terminating memory included and up to 20GB terminating memory available
* Upto 100GB Storage
* Other features like Always on, unlimited usage, invite Collaborators to Projects, Supports Custom Domain and Scheduled jobs.

Open Shift is a community-driven Open Source software support which supports wide verity of languages runtime and data layers including Java EE6, Ruby, PHP, Perl , MySQL.

In Open Shift to launch asp.net application, additional third party software required. Downloaded Visual Studio add-on fromClick2Cloud (<http://click2cloud.net/>)



# 20 November 2017 – All

Had our sync after the lecture. Start Time: 6:00 PM. End Time: 7:40PM.

Main points for discussion:

1. Normal Status update
2. Looking at the issues we are facing in hosting the Face API website in AWS. In AWS, when we are trying to upload an image from local hard drive, we are getting error. After debugging we found that, the code is failing to create a folder in website directory. Though this works fine in Azure. Possible solution discussed are to give permission to ‘wwwroot’ folder in AWS server machine or updating the code to store uploaded images in a different folder.
3. Looking at the issues faces in Open Shift. Open shift has different mechanism to host ASP.NET 4.5 and ASP.NET Core based websites. Turns out that former is more complex. We were trying to set up proper tools on Sanjay’s machine – Visual Studio 2015 Community version and Click2Cloud extension for VS 2015. Sanjay originally installed VS 2017, but the Click2Cloud extension is not supported. So, he uninstalled 2017 version and then re-installed VS 2015. During this process, looks like something has went wrong with Visual Studio and .NET Framework. He is trying to do a clean install again.

# 21/11/2017 – Kevin

Accessed the Remote Desktop for the running instance in AWS after difficulties with Key Pairs. Accessed the C:\inetpub\wwwroot folder. Updated the permissions to allow full access. But this did not change the result.

Attempted to open the AWS Console in the Remote Desktop browser, but was told that there was a Java Script error. There was no option to update the Java Script.

Attempted to open AWS through the shortcuts pre-loaded on the Remote Desktop desktop. These did not allow login either.

Changed the code in the config file of the app to point to a different folder. No improvement.

Went through the process of creating and deploying a new app from the beginning again through Visual Studio. The result was an inaccessible web page with the error message of ‘403 – Forbidden: Access is denied”.

Went through the process of creating and deploying a new app from the beginning through the AWS Console in my laptop’s browser. No improvement.

# 22/11/2017 – Kevin

Web App successfully hosted via AWS after starting process again from the very beginning. The access rights of users in the Remote Desktop environment in the C:\inetpub\wwwroot folder appears to have been the problem, as previously suspected. App now successfully uploads images and identifies if faces are present in those images. Brief video demo uploaded to Github blog location.

# 22/11/2017 – All

Skype Call to discuss problems, next steps and ideas.

Storage – all three cloud providers to be examined for storage locations, where we can each store 10 image files. Using REST API, the Web App will then call those images and identify faces within them.

Monitoring - Agreed that all three cloud providers be examined for their monitoring metrics and capabilities. There will be differences across the platforms, so agreed that Azure be the reference platform and AWS/Open Shift to be compared to it.

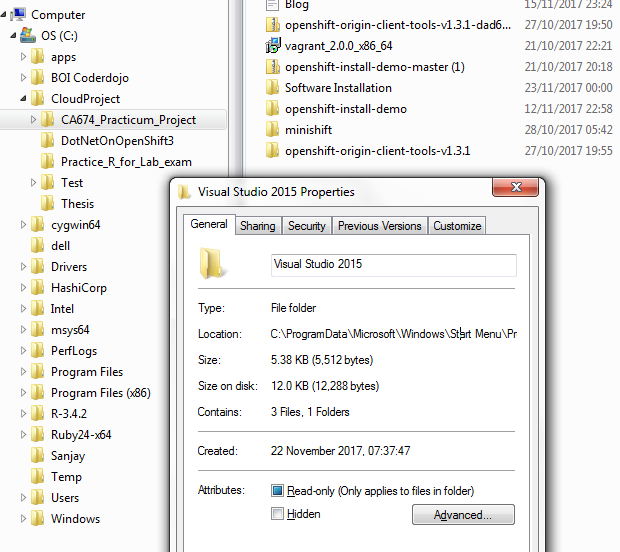
Configurations – exploring of configuration changes in all three cloud providers. This will involve changing API Keys with an aim to have one app which will be applicable across any of the three cloud providers, without any cloud-specific additional coding required.

Backup – all three cloud providers to be investigated for their back-up options.

Kevin to create a new folder in Github location to contain the working code used in AWS (code will be the same as the original code).

# 22/11/2017 – Sanjay

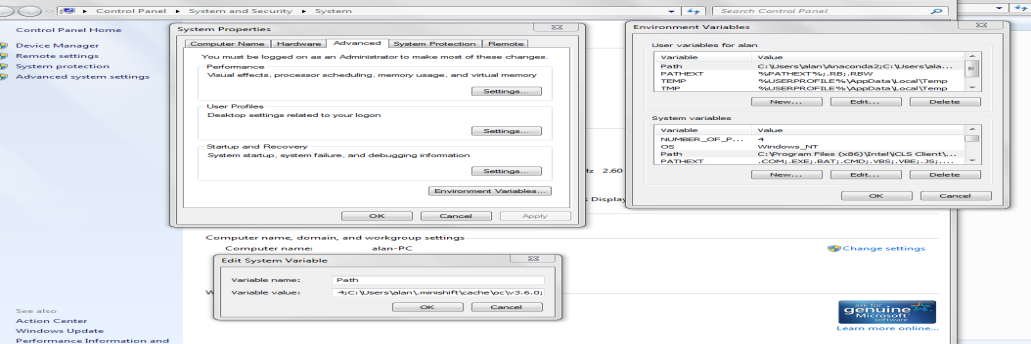
After analysing Open Shift add-on which was downloaded for Visual Studio 2017, discovered that Click2Cloud add-on does not support and no template was available for asp.net 4.5. Hence downloaded Visual Studio 2015.



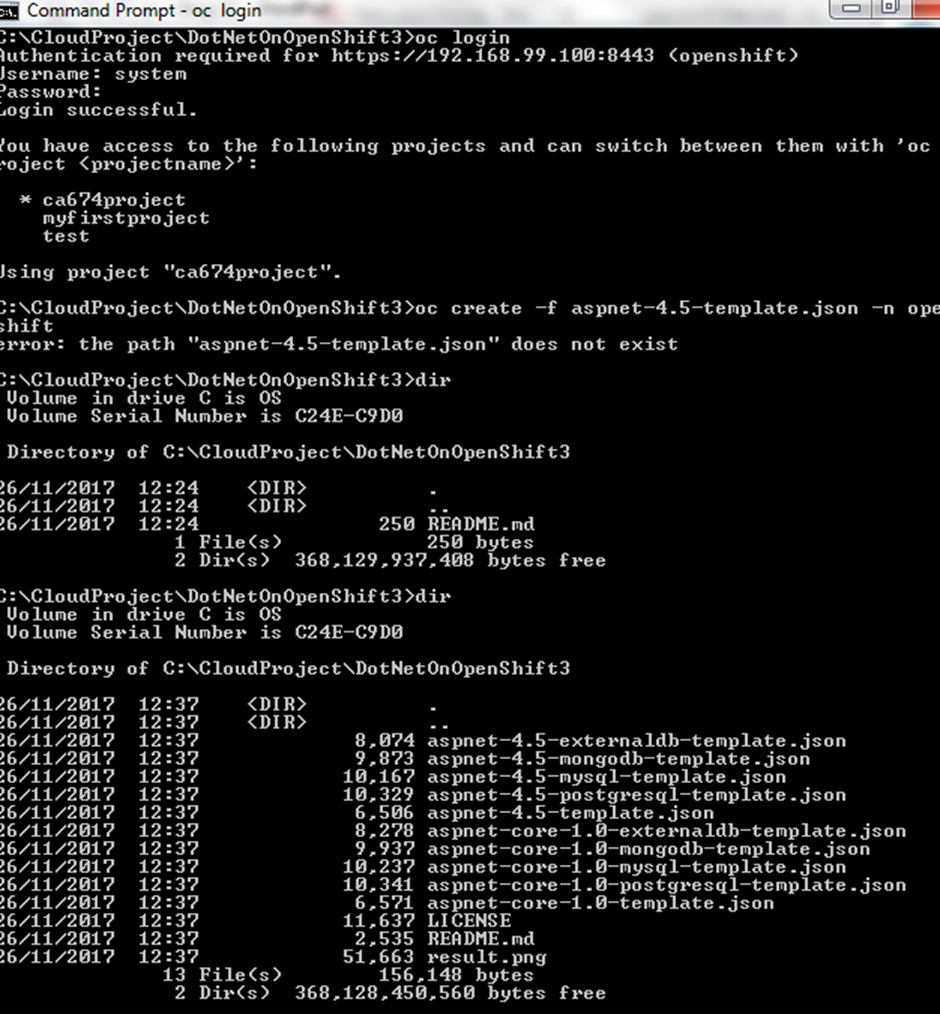
Using Visual Studio Community edition 2015, first tried to launch FacApi website in local laptop and follow step by step process to download Click2Cloud add-on and additional software.

Pre-Requisite Softwares

1. Install Microsoftware Visual Studio 2015 – Already downloaded and launched FacApi
2. Git for Windows
3. Open Shift Command Line Interface (CLI)
4. MicroSoft .Net Core SDK Preview 2
5. Setup environment path to access Open Shift from Command line & Open Shift Command Line Interface (CLI)



Open Shit Command line login screenshot



# 24/11/2017 – Kevin

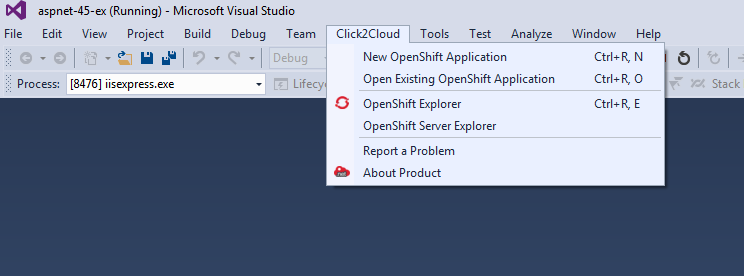
Investigating hosting the website files in an AWS S3 bucket and launching the website from there, rather than from my laptop location. Accessing the application files in S3 through a REST API appears to require an SDK (Software Development Kit). I think Visual Studio can act as an SDK for AWS but I need to confirm this.

Suggested report format to the team.

The AWS equivalent of Azure Application Insights is Amazon CloudWatch. This can be used for monitoring of application performance issues. The AWS equivalent of Azure Operational Insights is Amazon CloudTrail. This can be used to perform data analytics of your application.

# 26/11/2017 – Sanjay

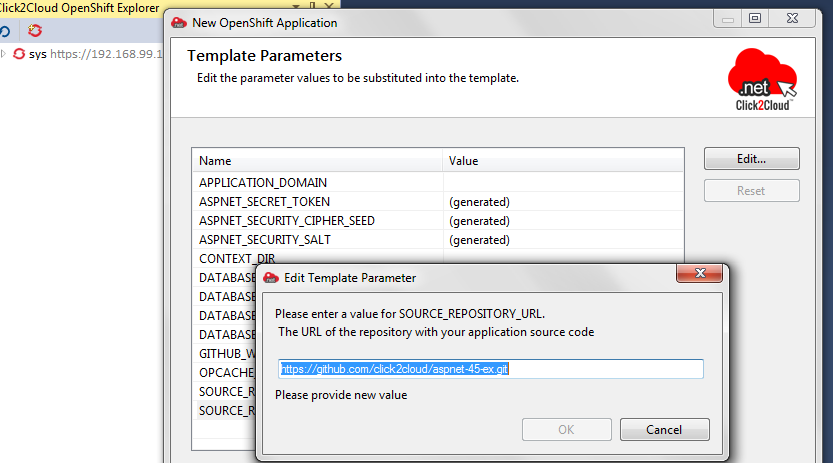
After downloading all Pre-requisite software, created first asp application using Custom template and followed below steps

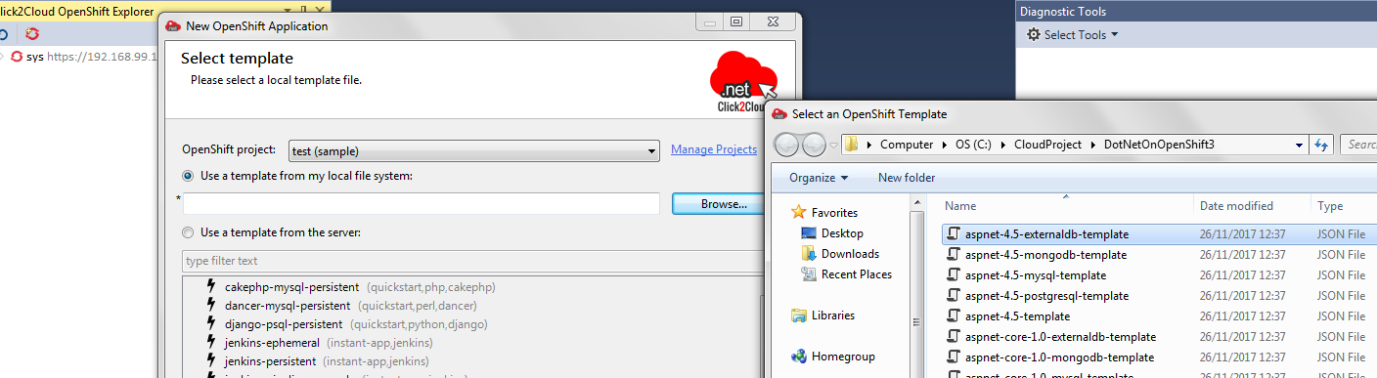


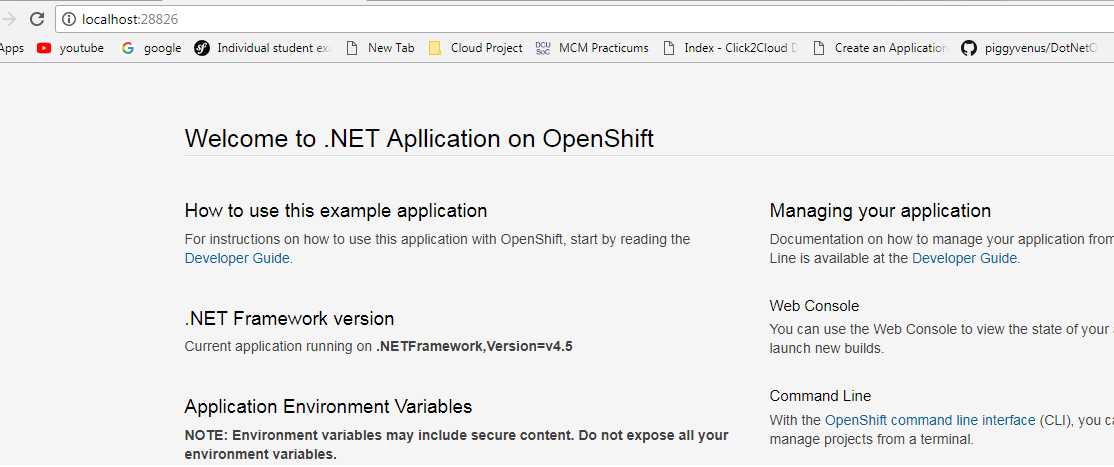
Created project in connected server by selecting “New OpenShift Application” and provided new project and other details and click Finish button.

Select template using template wizard.

Open Shift Command Line Interface (CLI) and downloaded Click2Cloud add-on for asp.net 4.5 templates and select Use a template from my local file system. Then click on Define button using Template details dialog box from Open Shift resources and click next or finish to create with default template parameters and labels. Below are the screenshots





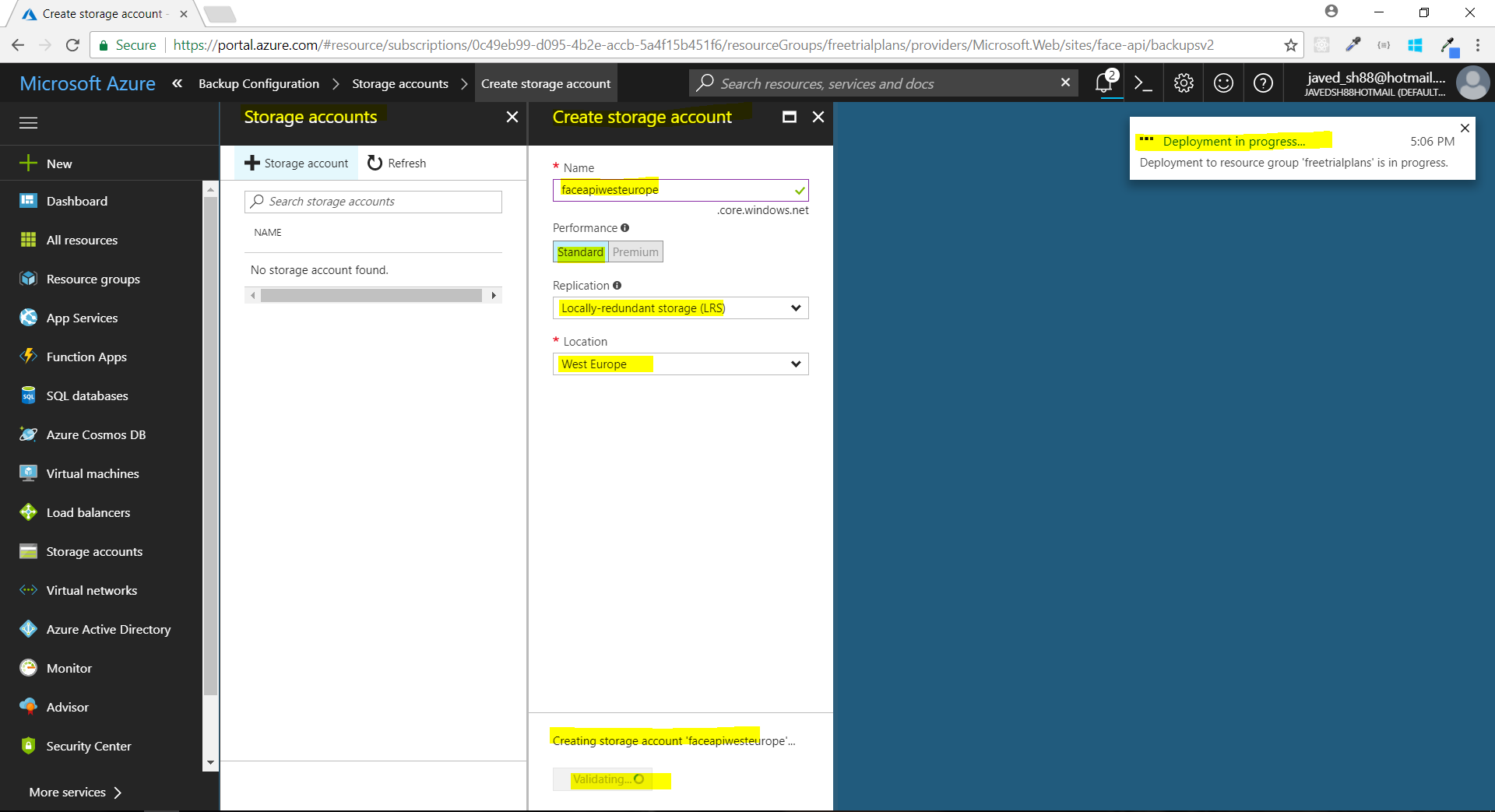


# 26th November 2017 – Javed

**Storage account created in Azure**

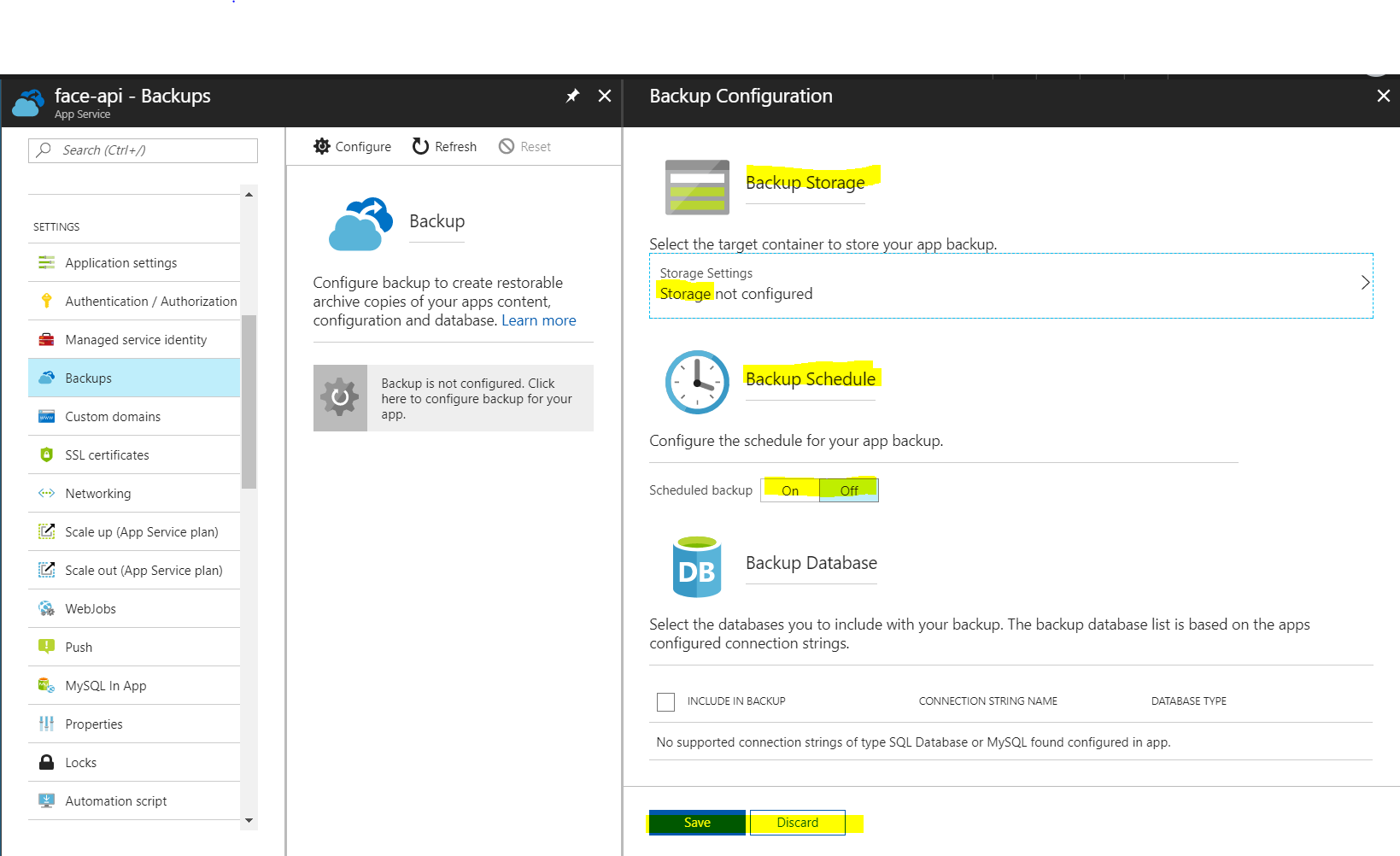
* Name: faceapiweustorage, Endpoint: <https://faceapiweustorage.blob.core.windows.net/>
* Authentication: Public, Subscription: Free Trial, Type: Blob Storage
* Location: West Europe, North Europe, Replication: Read-access-geo-redundant (RA-GRS)
* **Time Taken: 1-2 mins**
* Problems faced: Azure was not creating the storage account for almost half a day. I tried after few hours and it worked fine. Looks like there was a temporary glitch in the Azure Portal.

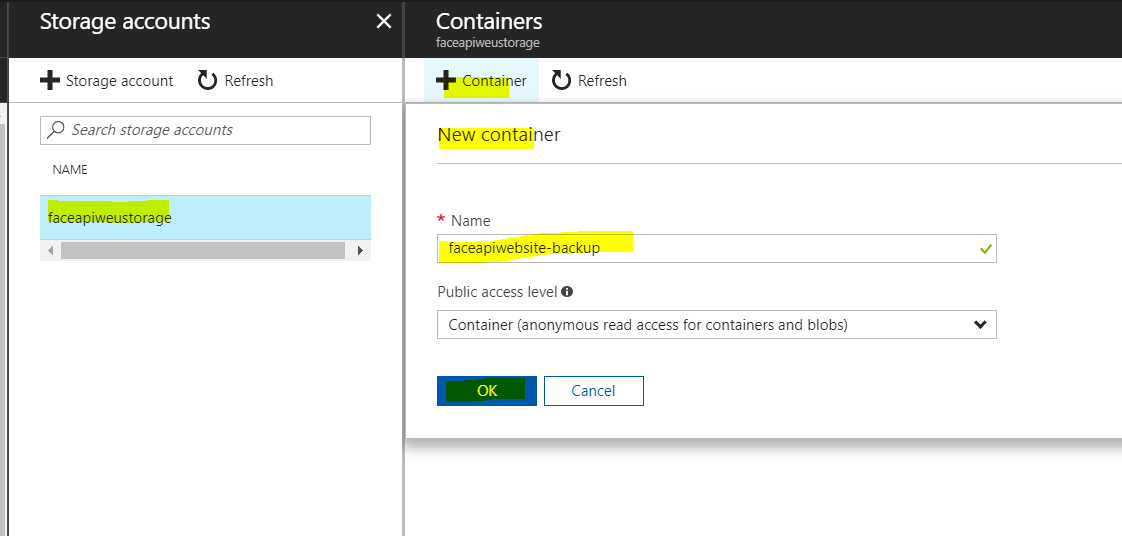
**Screenshot**

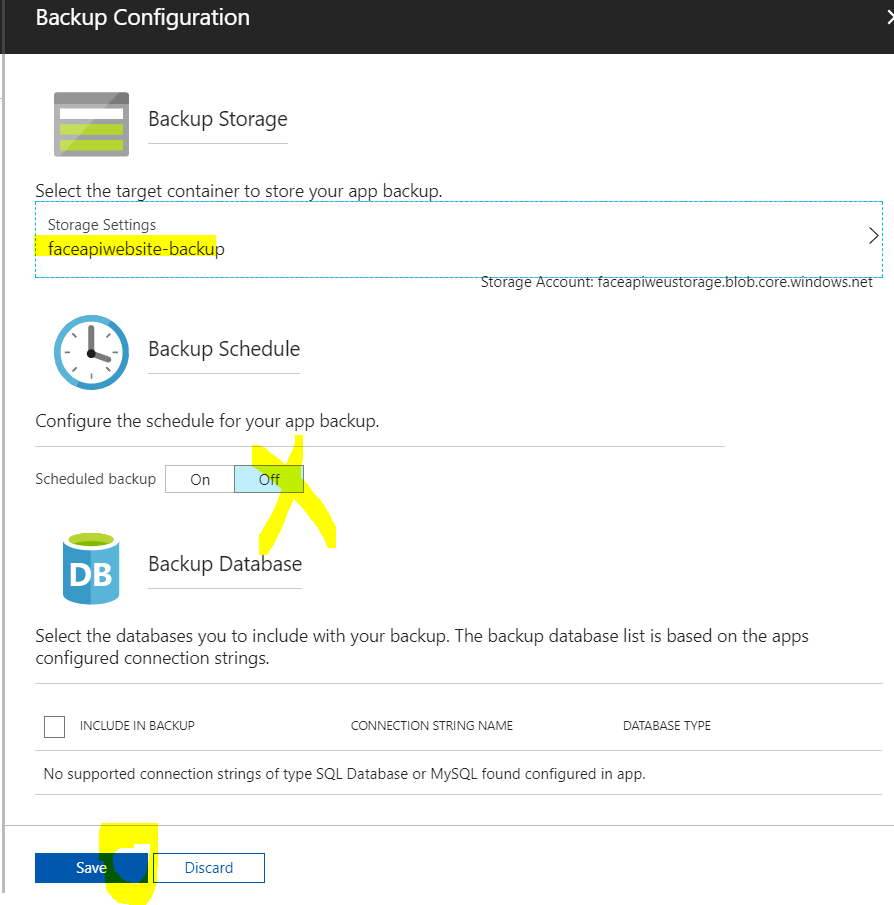


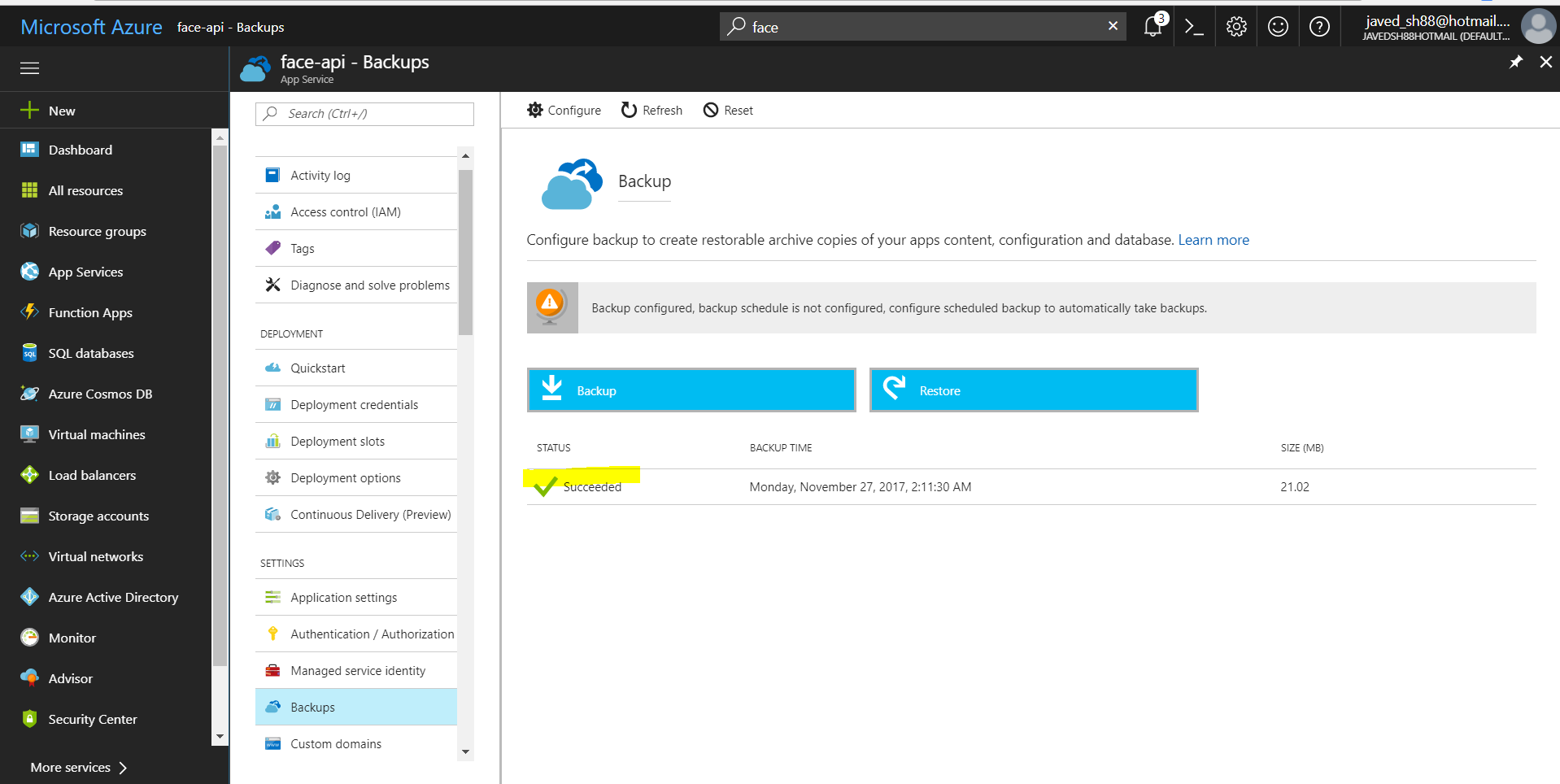
**First Back up created for Azure Face API Website**

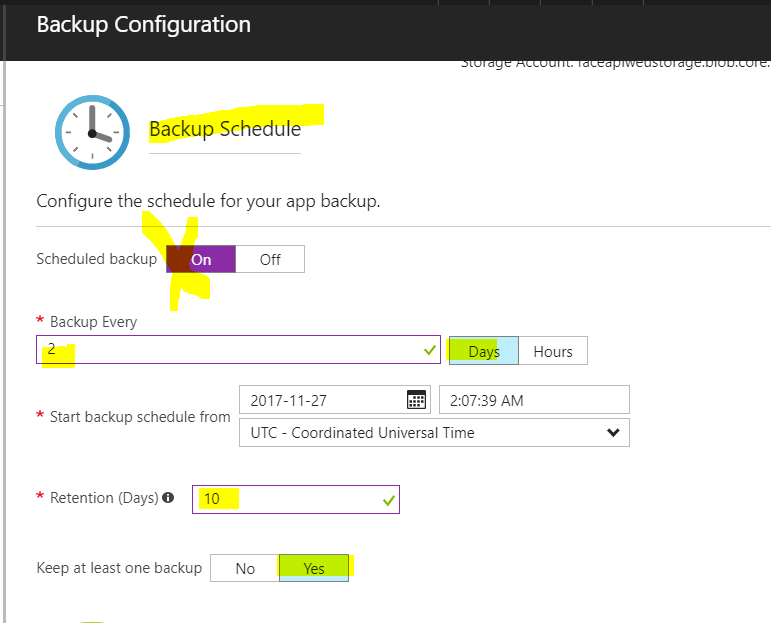
* Back up was created using Azure online portal - <https://portal.azure.com>
* Storage account ‘faceapiweustorage’ was required for the back up.
* Backup is set to be schedule for every 2 days and retention policy of 10 days.
* **Time Taken: 4-5 minutes**

**Screenshots**







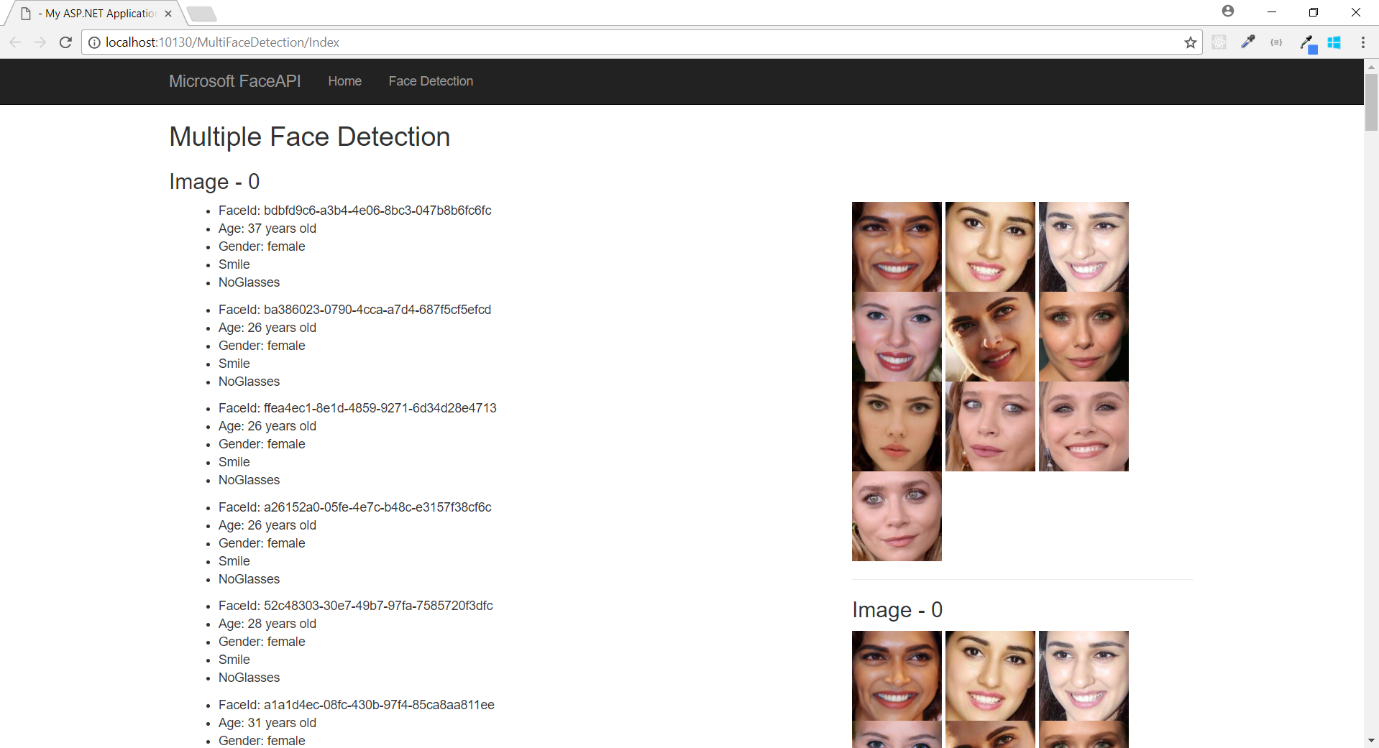


# 27th November 2016 – Javed

**Added new feature – Multi Detection of faces from multiple images.**

* Uses storage (azure), created yesterday.
* Storage contain sample images (10). These sample images are fetched at runtime, and iteratively FACE API is used to detect faces. Results are displayed with the detected faces highlighted.
* This is the first version. The code needs to be polished, specially the look and feel.
* Storage is accessed via REST APIs. Storage name and access key are stored in Web.config file, so that it is easy to change later. Also, advantage of REST API will be when we move to another cloud, less changes will be required (for using different storage e.g. AWS)
* Problem Faced:
  + Accessing storage via REST API (few problems in encoding)
  + Creating cropped files on the cloud server (website)

**Screenshot of website**

****

# 27th November 2017 – Javed

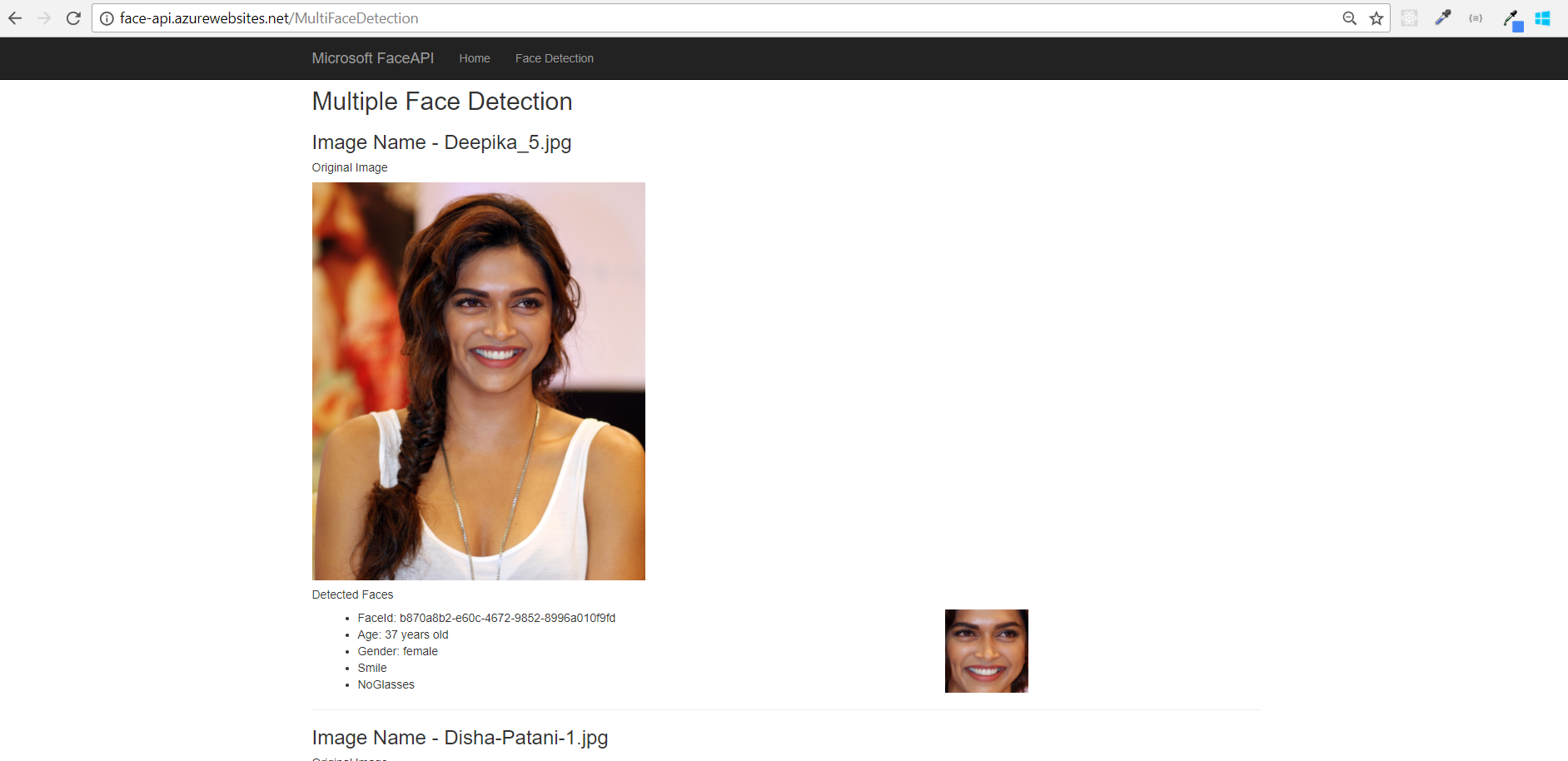
**Fit and Finish for Face API Multi Faces Detection using Storage (Azure and AWS)**

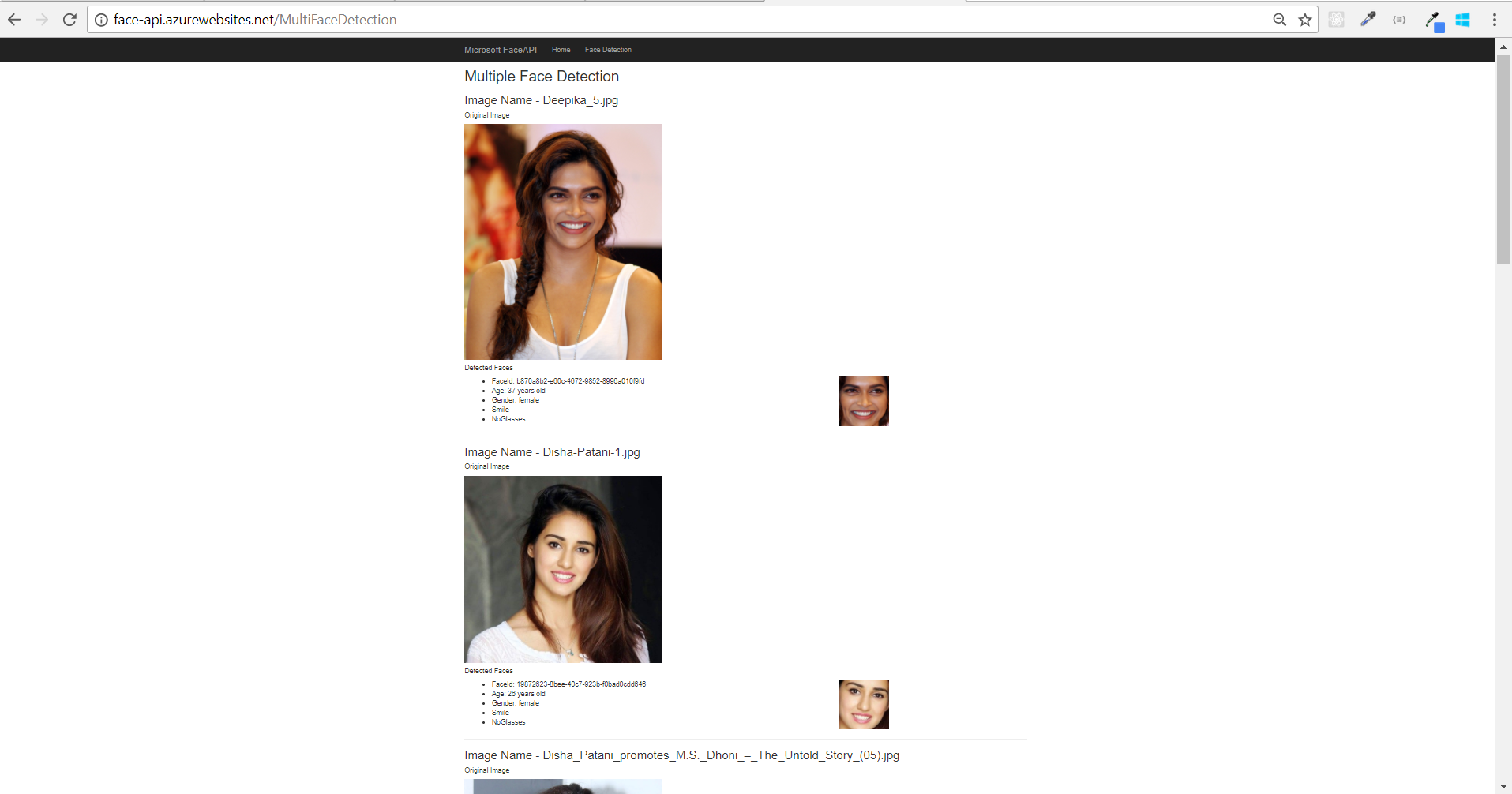
* Update the code to display much better HTML (beautiful)
* Updated git ignore file (.gitignore), to ignore unwanted files and not them committed to GITHUB, like binaries etc.
* Updated web.config to set the custom error mode to OFF, so that when accessing the website (not on local), we can see the actual error.

**Problems Faced:**

* GitHub was down for few moments

**Screenshot**





# 27/11/2017 - Kevin

# Revised Draft Report circulated to the group with suggested layout and headings. To be altered as required by the group. Once agreed upon the format, I’ll upload it to Github.

# 28th November 2017 – Javed

**Migration to AWS S3 storage for AWS website**

* Uploaded same 10 images as for Azure website to AWS S3 storage
* Set the permission to public for easy access
* Updated the AWS website to use the S3 storage via REST API and C#

**Problems Faced**

* None. I just have to update few lines of code and that is it. So, migration to S3 storage was pretty issue and thus we can say high interoperability.

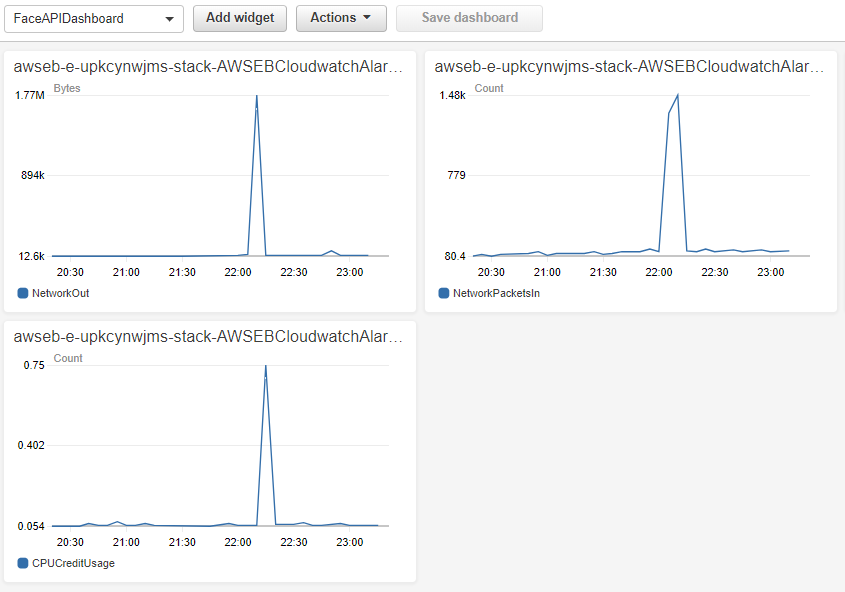
**Moved Azure Face API to proper blob storage (Free Trial Version)**

Earlier the azure subscription used was not the free trial version, which we are using for Azure. It was my company storage. The reason for this was; I was not able to create blob storage in Azure under free trial subscription for almost half a day.

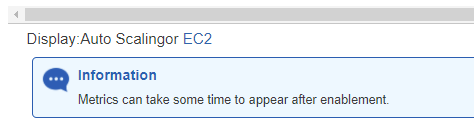
# 28/11/2017 – Kevin

Auto-scaling code run in AWS from Visual Studio and successful first time. Images are successfully retrieved from S3 storage area in AWS and resulting output visible on url.

CloudWatch monitoring launched within AWS to verify Auto-scaling has been achieved. Dashboard created with all Instance-related metrics displaying, such as packets in/out, Credit Usage, etc – snapshot below.



AutoScaling monitoring enabled in AWS, but may take some time for metrics to appear. Will be checked again later.



29/11/2017 – All

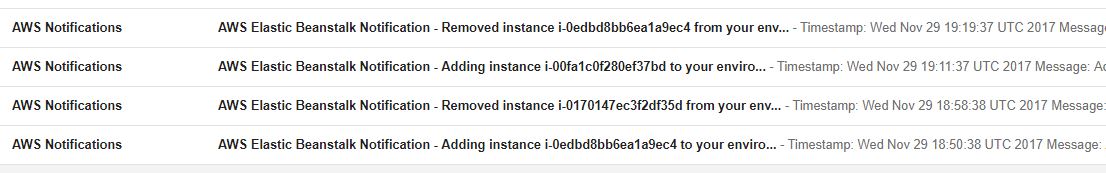
Team meeting held after Week 11 lecture in DCU. Main topics discussed were:

* Continued work on deploying a website to OpenShift and issues found there
* Further research to be conducted by all team members into their respective cloud platforms.
* Report format reviewed by the team. Kevin to make some alterations and upload to Github.
* General outline for the content of our videos – due to the 5-min limit for the whole team, it will be a high-level demonstration of the platform and any working/deployed code. Videos to be completed individually and then amalgamated.
* Presentation slides to follow on from our completed report.
* Trial run of our presentation to be done after Week 12 lecture in DCU.

29/11/2017 – Kevin

Report uploaded to Github for the team to begin working on in their own time.

Auto-scaling in AWS verified as working, evidenced by the increase and decreased number of instances. Due to monitoring alerts setup, the changes in the number of instances were sent to my email account – see below.



CPU Utilization graph in the Elastic Beanstalk Monitoring section verified the CPU loading being experienced as a result of the web app coding being modified.

