**UWB – CSE 490A - Intro to Cloud Computing**

**Azure WebSite project**

**Objective**

In this project, you will demonstrate basic understanding of building a WebSite using Azure PaaS features including Web Apps, End Point monitoring, Tables, entity framework, deployment slots etc.

**Cloud concepts covered**

1. Platform as a Service
2. Web Sites using server side code
3. Azure Table storage (NoSQL) for semi-structured non-relational data
4. Azure Entity Framework
5. End point monitoring from geo-distributed locations
6. Horizontal Auto-scaling of Web-Site based on CPU usage
7. Slot based deployment for production and staging activities
8. Model-View-Controller based apps (Extra Credit)

**Overview**

In this project you will be building a Web site using ASP.NET WebSite feature in Visual Studio 2015 and running the same in your Azure subscription. One easy way to build this in VS is with WebForms but you are free to choose which ever template you like. If you choose to build using MVC then you will be eligible for extra credit (see below). You can write code using just the source mode or by drag-n-drop with the design mode or both. When the site is launched its home page should have three sections:

1. Student Name label
2. Load data
3. Query

Load Data section will provide a button (with appropriate text on it e.g. “Load Table”). When the user will click on this button your server code will read customer entries from a blob. Blob will be available at <https://sbsa.blob.core.windows.net/websiteproject/sample.txt>. You should copy this file into your own storage blob. After reading these entries from your blob, your code should create an Azure storage table and enter each row from the blob into your table. You are free choose any name for your table. The blob will have multiple entries each representing a unique customer with last name being the partition key (first entry in a row) and first name being the row key (second entry). Partition key and row key combo will be assured to be unique. These two keys will follow variable numbers of key and value pairs with a “=” in between them and will define that customer’s information. Here are few examples:

Bharati Sudeep Email=sudeepb@uw.edu Phone=425.555.1234 Gender=male

Smith Brian Phone=425.555.9687

Gordon Jill Email=jillg@uw.edu Phone=425.555.2345 Gender=female

Wagner Bobby Phone=425.555.5463 Email=bobbyw@hotmail.com

Smith Jeff Phone=425.555.7643 Email=JeffS@gmail.com

As an example, in the first customer entry, Bharati (last name) is the partition key, Sudeep (first name) is the row key and Email and Phone are the attributes associated with that customer.

Once this load operation succeeds, your site should print Success or failure next to the button to let the user know the status of that operation.

User should be allowed to load the table more than once and should be notified if there is any issue in building the table with an appropriate message in the adjacent text box.

In the third section of Query, you will enable the user to enter combination of three values in three text edit boxes – last name, first name and attribute. One of the first or last name will always be given. If only one of first and last name is given and the other one is blank, would mean the query should return all the matching customer entries. Attribute text box should allow a value as well as blank. Blank attribute would signify that the user is querying for all the attributes. Next to these three text boxes you will have a “Query” button. After filling text in the text boxes and on clicking the Query button, your server side code should return back the right set of value pair attributes. For example (<> means blank entry):

Smith, Brian, Phone => should return

“Smith Brian Phone=425.555.9687”

<>, Brian, Phone => should return

“Smith Brian Phone=425.555.9687”

Smith, Brian, Email => should return

“Smith Brian Email=”

Smith, <>, <> => should return

“Smith Brian Phone=425.555.9687”

“Smith Jeff Phone=425.555.7643 Email=JeffS@gmail.com”

Smith, <>, Phone => should return

“Smith Brian Phone=425.555.9687”

“Smith Jeff Phone=425.555.7643”

You should publish your web site from VS by using publish profile from the Azure portal for the Azure web site where you want your project to be published.

You should setup monitoring of end point of your web site from at least two different geo locations.

You should setup horizontal auto-scaling for your website with minimum 1 instance and max 3 instances with 1 instances as long as CPU usage is below 50%. 2nd instance should kick in when the CPU usage goes beyond 50% and 3rd instance should be added when the CPU usage goes above 80%.

You can do most of the testing of your site within the browser and localhost. And once you feel good about your site you can host it on Azure with the publish command of VS.

You can test most of your functionality in Azure with Free tier website but ensure to move to Standard when you test monitoring, slots etc. When you submit your work, your website should be in standard mode.

You should setup at least one slot to show production vs. staging deployment and demonstrate the swapping of the two as part of continuous development practice.

It will be an extra credit work if you choose to build your web site with Model View Controller (MVC). If you do choose to build based on MVC, it won’t be required then to build your site with WebForms or other templates.

**Submission**

1. Assigned on Nov 2nd 2015
2. Due by end of the day Nov 9th 2015
3. Submit a compressed zip folder in canvas containing your full VS project
4. Leave your website running for me to test it out for grading. It will need to be in standard hosting tier. I will check:
   1. Site is working as expected for various queries (may change the file entries and reload)
   2. Publishing has taken place from VS
   3. End point monitoring data is available in the portal
   4. Auto-scaling is setup properly (exercising auto-scaling functionality is not required)
   5. Slot based deployment is working with two successive swaps
   6. Code inspection of correctness, cleanliness and relevant comments

**Extra Credit (15%)**

1. Extra Credit for MVC based web site development will be 15%
2. Same checks will be done as mentioned above for MVC based site as well

**Few Useful Links**

1. ASP.NET WebForms intro - <http://www.asp.net/web-forms/overview>
2. How to create a simple APS.Net MVC Web App from VS and create a new site to deploy it with one button. <https://azure.microsoft.com/en-us/documentation/articles/web-sites-dotnet-get-started/>
3. How to use Azure Storage Tables and entity framework - <https://azure.microsoft.com/en-us/documentation/articles/storage-dotnet-how-to-use-tables/>
4. An article on Azure Tables - <https://msdn.microsoft.com/en-us/magazine/ff796231.aspx>
5. Table creation after a delete operation can take up to 40 seconds - <http://answers.flyppdevportal.com/categories/azure/azuredevelopment.aspx?ID=f79c24cc-81e5-4b9b-8e9c-2a066115936f>
6. Intro MVC app with server and client side code <http://docs.asp.net/en/latest/tutorials/your-first-aspnet-application.html>