**Introduction to Azure**

Azure is the cloud computing platform offered by Microsoft. Azure offers SaaS, PaaS, and IaaS and includes over 600 different services.

**Azure Services**

Click [here](https://azure.microsoft.com/en-us/services/) to view all services offered by Azure.

1. Azure Application Services
   * Azure AI
   * Azure Analytics
   * Azure Internet of Things(IoT)
2. Azure Data Services
   * Azure Storage
3. Azure Development Services
   * Azure DevOps
4. Azure Compute Services
   * Azure Virtual Machines
   * Azure Container Service
5. Azure Network Services
   * Azure Content Delivery Network (CDN)

**Azure Price Model**

The benefit of Azure, as well as most cloud-computing services, is that it features a pay-as-you-go subscription model, meaning you only pay for what resources/services you use.

The exact price range varies from service to service. Here is an example of how much money hosting an SQL database would cost.

Notice that the more data is reserved, the higher prices goes. Customers also get a discount if they commit to a multi-year plan. This is a good option if you know you're going to need the database for a certain amount of time.



**Azure Regions**

Azure's datacenters are split up into regions in order to achieve low latency and high availability. Each region is comprised of multiple data centers that are all connected via a regional network. An Azure Geography is a discrete market that is comprised of one or more regions.

Azure has many regions worldwide including East US, Canada Central, and Japan East. It is the developer's job to choose a region that is close to the users, so they can have a low-latency experience. This [page](https://www.azurespeed.com/Azure/Latency) can be used to test latency.

It is important to note that not all services are available in every region. This [page](https://azure.microsoft.com/en-us/global-infrastructure/services/) can be used to search for services and see which are available in what regions.

**Azure Resources**

An Azure resource is an object used to represent each service in Azure. They can be expressed as a JSON object and contain all the relevant information about a service such as type, name, and location.

A resource group is a logical grouping of resources. There is no strict requirement on how resources should be grouped but a few options are grouping resources by their service type (SQL, Web, VM, etc.) and grouping resources by their stage in the development cycle (development, production, etc.).

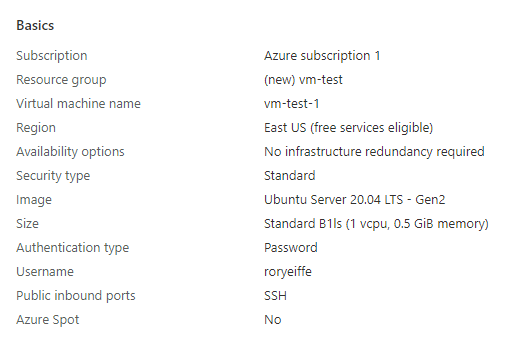
If we want to create a new resource in Azure, we need a resource group to put it in, so we always need to have at least one resource group created before we create a resource.

**Azure Virtual Machines**

**How to create a Virtual Machine**

1. Navigate to the [Azure Portal](https://portal.azure.com/#home) and sign in.
2. Click on "Virtual Machines".
3. On the top left of the Virtual Machines interface, click on the "Create" button. Select "Virtual Machine" from the drop-down menu. You should see a form with different fields relating to your VM.
4. Select your subscription and resource group (create a new one if needed).
5. Your Virtual Machine will also require a name and a region. The name must be unique and the region should be whatever is closest to you or your users.
6. Select your image from the drop-down. This module uses Ubuntu Server 20.04 as an example.
7. Pick a size from the drop-down menu. Click on "See all sizes" to view all available size options. Notice the different columns (CPUs, RAM, Data disks, etc.) and how they relate to the price per month.
8. Choose an authentication type and set up your credentials.
9. For public inbound ports, either select "None" (you can edit network configuration later) or select "Allow select ports" and choose any of the preset ports. This tutorial uses SSH(22).
10. Click on Review + Create.

This is what the settings could look like:



1. Click on "Create" and wait for the VM to be created and deployed. Once this process is done, you are ready to connect to your virtual machine!

**How to connect to a Virtual Machine**

1. Navigate to the [Azure portal](https://portal.azure.com/#home) and log in.
2. Click on "Virtual Machines".
3. Click on your designated virtual machine from from the list of VMs.
4. Click on "Connect" on the top of your screen and the propose method of connecting from the dropdown. This example uses SSH as an example.
5. On the connect page, follow the given instructions. The rest of the module assumes you are using SSH to connect.
6. To connect to a VM using SSH, open up your command prompt and navigate to a folder where you would like to store your SSH key.
7. Create an SSH key pair using ssh-keygen -m PEM -t rsa -b 4096. Your command prompt will ask for a filename. Enter one or leave it empty for the default. Enter a passphrase for the file or leave it empty to not include a passphrase.
8. To connect to the VM hosted on Azure, type ssh -i <path\_to\_ssh\_key> <user\_name>@<ip\_address> into your command prompt.

* Replace <path\_to\_ssh\_key> with the path to the generate SSH key pair from step 7.
* Replace <user\_name> with your azure username and <ip\_address> with the ip addess of your VM. These can both be found on Azure under the SSH section of connect.

1. After entering the command from step 8, enter the password that you set on Azure. You should now see your Virtual Machine. If so, you have successfully set up and connected to your virtual machine!

**Deploying a Web Server to VM**

This example will show you how to add a simple web server to your virtual machine. If you haven't yet, follow the previous steps to get your virtual machine running and connect to it from your host computer.

1. Once connected to your VM, type in sudo apt update && sudo apt -y install apache2 to update your VM and install Apache.
2. The Apache server should now be running. To check the status of our server, type sudo systemctl status apache2.
3. Finally, we can update the Apache web page to be any HTML file we want, by typing the following into the command line:

echo '<!doctype html><html><body><h1>Hello World!</h1></body></html>' | sudo tee /var/www/html/index.html

Feel free to change the HTML code.

1. To test our Apache server, we need to allow access via HTTP. Head back to the Azure portal and navigate to the virtual machine that we have running. On the sidebar, there should be a tab called "Networking". This is where we can update our port rules.
2. On the Networking page, you should see a list of port rules. Right above the list, click on "Add inbound port rule". A screen should pop up that lets you configure the setting for this new port rule. Find the "Service" dropdown and change it to "HTTP". Ensure that the port changes to 80.
3. Finally, navigate to http://<ip\_address>:80 in your web browser. Replace <ip\_address> with the ip address of the virtual machine. You should see the custom HTML code written in step 3. If so, you have successfully deployed a web server to your VM!

**Azure Storage**

Azure Storage is Azure's solution for data storage scenarios. It allows for the storage of a variety of data objects in the cloud. Read more about it [here](https://docs.microsoft.com/en-us/azure/storage/common/storage-introduction).

**Benefits of Azure Storage**

1. Durable and Highly Available - Redundancy of data storage allows data to be durable in the event of an outage and highly available.
2. Secure - Data written to an Azure storage account is encrypted.
3. Scalable - Azure Storage is designed to meet whatever performance needs an application requires.
4. Management - Like most Azure services, management tasks including maintenance and updates are handled by Azure.
5. Accessible - Data is accessible from anywhere in the world via HTTP or HTTPS.

**Azure Storage Services**

* Azure Blobs - a massively scalable object store for text and binary data
* Azure Files - managed file shares
* Azure Queues - messaging store for reliable messaging between application components
* Azure Tables - NoSQL stores for schemaless storage of structured data
* Azure Disks - block-level storage columes for Azure VMs

**Azure Blob Storage**

Blob storage is optimized for storing massive amounts of unstructured data, such as text or binary data.

**Use Cases:**

* Serving images or documents directly to a browser
* Storing files for distributed access
* Streaming video and audio
* Storing data for backup and restore, disaster recovery, and archiving
* Storing data for analysis by an on-premises or Azure-hosted service

**Deploying a static website using Azure Storage**

**Prequisites**

1. Own an Azure account (free version is fine).
2. Have Visual Studio Code installed on your machine. [Link to download](https://code.visualstudio.com/download)

**Azure Steps**

1. Sign in to your Azure account and navigate to the [Azure Portal](https://portal.azure.com/#home)
2. Click on "Create a Resource".
3. Scroll to find, or search for "Storage Account". Click "Create" right underneath.
4. On the storage account page, pick a subscription, resource group, region, and give a name for the storage account. When this information is entered, click on Review + Crate, and then confirm the creation of the storage account. Wait for deployment to finish.
5. Go to the resource by clicking on "Go to resource" when deployment is finished. Alternatively, navigate to your list of recent resources by going back to the [home page](https://portal.azure.com/#home) and clicking on the storage account.
6. Find the capabilities tab and click on "Static website" to configure one.
7. On the static website page, click the toggle bar to enable the static website.
8. Two input fields should appear.
   * Under "Index document name", put "index.html" or a file name to represent the entry point of the website.
   * Under "Error document path", put "404.html" or a file name to represent the error page of the website.
9. Save your changes and 2 endpoints should appear. We'll use these later.

**Visual Studio Code**

1. Head over to Visual Studio Code and make sure you have the Azure Storage extension installed. To do so, on the left side-bar, click on the extensions tab (it looks like 3 connected squares with a 4th one that is detached). Search "Azure Storage" in the bar that comes up. Either install it, or verify that it is already installed.
2. Create a new folder and create 2 HTML files in it, corresponding to the names configured in the Azure portal (index.html and 404.html).
3. Populate the 2 files with some sample code.

index.html

<!DOCTYPE html>

<html>

<body>

<h1>Hello World! From Rory</h1>

</body>

</html>

404.html

<!DOCTYPE html>

<html>

<body>

<h1>404 Error from Rory</h1>

</body>

</html>

1. To put it all together, right-click on the file explorer on the left and select "Deploy to Static Website via Azure Storage".
2. Visual Studio Code should prompt you to sign in and select a storage account. Select the account created in this module. When deployment is complete, a notification should appear on Visual Studio Code with a link to the website. Alternatively, you can navigate back to the Azure portal to grab the link.
3. The website should display what you put in your index.html file. To test out the 404.html, try adding "/anything" to your url. You should see the 404 page. If these work, you have successfully deployed a website to Azure storage!

**Creating an Azure Database for MySQL**

**Prerequisites**

1. Azure Account
2. MySQL work-bench, can be found [here](https://www.mysql.com/products/workbench/)

**Steps**

1. Navigate to the home page of the Azure Portal, [here](https://portal.azure.com/#home).
2. Click on "Create a Resource".
3. Search for "Azure Database for MySQL" in the search bar and select it. Click on "Create" and then select "Create" under "Single Server".
4. Select a subscription, resource group, and location. Enter a server name that is unique. Enter an admin username and password. Make sure to remember these credentials.
5. Click on "Review + Create" and then "Create" after verifying the information.
6. Go to the resource and click on "Connection security" on the left sidebar.
7. Click on "Add current client IP address" and save your configuration. This will let you connect from your own computer.
8. Open up the MySQL workbench, select "Database" from the navbar and then click "Connect to Database".
9. Enter in the server information including the hostname, port number, and username/password.
10. The port number should be 3306 unless you changed it from the default.
11. The hostname can be found on the overview page of the resource (labelled "Server name").
12. The username can be found under "Server admin login name" on the overview page of the resource.
13. Click "okay" and you should be able to see a text box where you can type in queries. If this works, you have successfully connected to your Azure MySQL database!

***Thank you***