**MS Azure Cloud (5th Nov 2019)**

**Challenges Facing on Premises:**

Infrastructure costing

Compute, storage and network

Scaling

Scale up – increase the size of existing instance.

Scale out – increase the number of instances.

Backup

Explicit backup policies need to be implemented.

Security

For application and network need to be configured explicitly.

Monitoring

Need to configure monitoring services.

Compliance

**Cloud: MS, Amazon, Google etc.**

In cloud no need to buy infrastructure – No Infra Cost

Scaling of application instances happens quickly.

Backup – in-built backup policies implemented.

Security – all levels security is enabled.

Monitoring is by default enabled.

Compliance – All kinds’ application can be deployed.

**Pay As You Go Model**

New frameworks and technologies are adopted.

**Types of Clouds:**

Private Cloud [HP, Oracle]

Cloud infrastructure for a single customer.

Cloud vendor provide service only for a single customer.

Public Cloud

Anybody can register and create an account.

You can buy the subscription and start creating services.

Multiple users share the same infrastructure.

Cheaper compared to private cloud.

Hybrid Cloud

Combination of public and private cloud.

Transaction data can be on private cloud and older data can be on public cloud.

**Cloud Models:**

Cloud services include DB Services, compute services, storage services, server less services, messaging services, Identity services, AI services and analytics services and more.

**There are three cloud service models**:

**IaaS** – Infrastructure as a service

Cloud vendor offers infra services like compute (CPU, RAM), storage, network etc.

You can set up your own app infra using the IaaS services.

This services are used by Admins (IT pros)

**PaaS** – Platform as a service

This is used by developers.

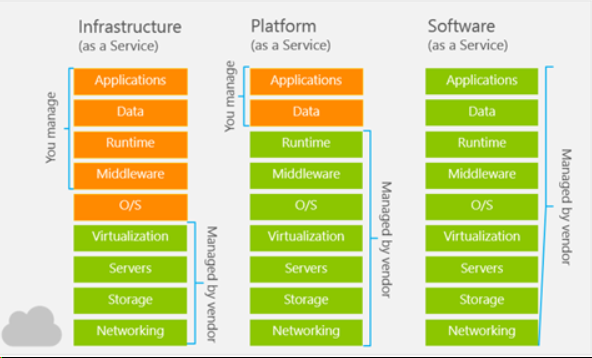
Developers will get a preconfigured platform where server, database and security is already configured.

**SaaS** – Software as a service

It is used by end users.

Software is also provided by the cloud vendor.

End user need to login and start using the application.



**Azure Cloud:**

Provides IaaS and PaaS services.

Azure is having Data Centers in more than 50 regions.

Azure subscription

Free subscription

Free trail – 13000 Rs / $200 for a month.

MSDN subscription

Azure PaaS

Pay as Yo Go – monthly bill.

Corporate accounts.

**Connecting Azure services**:

Web Portal (<https://portal.azure.com>)

Command line options

PowerShell

PowerShell for azure (only for windows) – AzureRm Module

PowerShell Core (cross platform) – Az module

Azure CLI – Cross platform

Azure SDK for programming languages (Azure .NET SDK)

ARM templates (JSON file – declarative model)

REST API

**Azure Resource Manager Model (ARM)**

Resource group

Grouping resources

Group wise deployment, group wise deletion.

Assign permission for group of resources (RBAC-roll based access control).

Group wise billing.

**Azure PowerShell commands**

To connect azure via PowerShell

>> Connect-AzAccount

To get list of resources via PowerShell

>> Get-AzResourceGroup

To create resource group via PowerShell

>> New-AzResourceGroup –Name “HexGroup” –Location “Southeast Asia”

To remove resource group via PowerShell

>> Remove-AzResourceGroup –Name “HexaGroup”

**Locally azure CLI commands**

To login from locally azure CLI

>> az login

To get list of resources via locally azure CLI

>> az group list -o table => for tabular format result

>> az group list -o tsv => for comma seprated result

To create group via locally azure CLI

>> az group create -n HexaGroup -l “Southeast Asia”

To get help via locally azure CLI

>> az group --help

To delete group via locally azure CLI

>> az group delete -n HexaGroup

**ARM Template (To create Resource)**

JSON file that contains the list of resources need to be deployed.

Sections:

Parameters

Any dynamic value accepted from user at the time of deployment.

Variables

Used to store intermediate and reusable values.

Resources

The azure resources such as VM, VNET, Storage ACC, DB etc.

Outputs

Output need to be printed in the screen after template execution.

Upload created JSON file from Azure portal



Then execute below commands on PowerShell

>> cd /home/username

>> cd /home/amol

Then To create/Deployment resource group

>> New-AzResourceGroupDeployment -Name “MyDeploy” -ResourceGroupName “HexawareGroup” -TemplateFile “./azuredeploy.json”

>> New-AzResourceGroupDeployment -Name “MyDeploy” -ResourceGroupName “HexawareGroup” -TemplateFile “./azuredeploy.json” -TemplateParameterFile “./azuredeploy.parameters.json”

**Storage Account:**

PaaS service for storing unstructured data/files in Azure.

One storage account can store **maximum 500 TB** data.

Maximum of **200 storage accounts** per subscription.

Every storage account as a unique name that is globally unique.

<https://mystorageacc.blob.core.windows.net/>

A Storage account provides four types of storage options:

**Blob** (Containers)

Unstructured blob files such as audio, video, image, text, documents files etc.

**File Shares**

Unstructured blob files such as audio, video, image, text, documents files etc.

It is possible to map a file share as a network drive in your machine.

It uses SMB 3.0 protocol

**Queues**

It is used for message based communication between applications.

One message max size is 64 kb.

Message max TTL is 7 days.

**Table** Storage

Key-Value pair unstructured data storage (NoSQL)

**Replication**:

It always takes minimum 2 copies of storage account data.

SLA of 99.99 on availability.

Methods:

LRS – Locally redundant storage (same data center)

ZRS – Zone redundant storage (only large data center)

GRS – Geo redundant storage (one country to another country)

RA-GRS – Read Access – Geo redundant storage (one country to another country access primary copy and read access to another country data center)

**Kind**:

Storage V1

All four services are available

Storage media as standard/premium

No option for selection Hot/Cool

Storage V2

Storage V1 + Blob

All service types are supported

Hot and cool is available

Standard and premium is available

Blob

Only blob (containers) type is allowed

Hot and cool is available

No premium support

**Access Tier**:

Hot

Cool

**Storage Media**:

Standard (HDD)

Premium (SSD)

**Security in Storage Account**:

Access Keys

Key1 and Key2 is available

Both the keys have admin privileges.

SAS Tokens

Shared Access Signature

Granular permission over the storage account.

**Blobs:**

Page blobs - disk file un-streamable

Block blobs -

Append blobs - streaming type Ex.

**Queues Service:**

Asynchronous message based communication.

Message max size is 64 kb.

One message max TTL is 7 days.

**Tables Storage:**

Unstructured Key-Value storage.

Data is stored as Entities.

Every entity has a row key and partition key.

An entity can have max of 256 keys.

For every entity there will be 3 built in keys – Row key, Partition key, Timestamp key etc.

**Row** key and **Partition** key is used to uniquely identify a record.

**File Shares:**

Is used to create Network file shares that can be mapped (disk drive) to your machine.

It is used as an extended storage for your machine.

It used the Blob storage concept behind.

It used the SMB 3.0 protocol that provides network mapping facility.

**Azure App Service:**

Is a compute service.

It Is a **PaaS** service used to deploy Web, API and Mobile Applications.

App service **Web app**

Is used to deploy web applications such **as MVC, Web Forms** etc.

App service **API app**

Used to deploy **RESTful services** and **web services**.

App service **Mobile app**

Mobile app backed service is deployed.

Push notifications can be sent to mobile frontend.

URL: <https://[appservicename].azurewebsites.net>

Custom domain mapping.

Every app service provides SLA of 99.9% on availability.

App service plan

Defines the SKU (Capacity) + Location

SKU (Plans)

Free plan

No SLA available

Used for testing purpose.

No custom domain mapping.

Shared infrastructure.

Shared plan

No SLA available

Custom domain mapping supported.

Shared infrastructure.

A minimal charge monthly.

Basic (B1, B2)

SLA available.

Dedicated machine.

Custom domain mapping available.

Manual scaling possible.

Not recommended for production (uses a low machine)

Standard

Recommended for production.

All feature of basic included.

Automatic scaling support.

Staging slot support (5 slots).

Daily 10 times backup.

Premium

More backup timing (50 times in a day).

Maximum 20 additional slots.

VM is created shared network.

ASE (App Service Environment) (Isolated)

VM is created in an isolated network.

**Deployment:**

Deploy from VS/VS Code

Deployment center

FTP

Github

DropBox

OneDrive

**Scale web Apps:**

Scale out – Horizontal scaling

Increase the number of instances of the application.

For performance improvement.

Manual scaling

Automatic scaling – scale based on some metric.

Scale Up – Vertical scaling

Increase size is increased than the number.

Changing the App service plan.

To increase the features.

**Deployment Slots:**

Slots are created for testing and staging environment.

By default a production slot only available.

We can create some additional slots in Standard (5 slots) and Premium (20 slots) plans.

**Cosmos DB Service:**

Database **PaaS** service.

A multi-model, planet scale database that supports multiple database engines.

**Document Types:**

Document DB (SQL API)

MongoDB API

**Key-Value Pair:**

Azure Table API

**Column-Family Type:**

Cassandra

**Graph:**

Gremlin

It uses only SSD disk.

**SLA – 99.99999 %**

Availability

Durability

Consistency

Latency

Cosmos DB Account

Select DB model

Database

Collection/Containers

Documents/Data item

**Throughput:**

Performance unit for a database and collection.

RU – Request Unit.

1 RU = one read of 1 KB data

**Multi-homing API:**

**Failover:** Automatic failover

Manual failover

**Consistency Levels:**

**AKS: [Azure Kubernetes Service]**

It is cluster service

Monitoring

Securing

Backup

Custom domain mapping

DR

Upgrade

**Orchestrators**:

Which do monitoring and management of cluster

Automatic patching

Automatic upgrade

Disaster recovery

Automatic scaling

Securing applications

User can give instruction to the cluster orchestrator.

Orchestrator receives instruction from user and executes within cluster.

Orchestrator (Compute Clustered services) in Azure:

**AKS** Azure Kubernetes Service

Master – worker node architecture.

Master machine is running the application

Master does not runs the client application

Client apps are running in worker machines.

Master machine contains system services such as cluster manager, storage service, network service etc.

AKS cluster can run only containerized services. (It is container cluster)

We can deploy micro services and monolithic apps.

**SF** Service Fabric

It is a micro services cluster.

Application can be containerized or non-containerized.

It can deploy apps that is developed using **Service Fabric SDK**.

In SF cluster there is no Master, all machines are worker nodes.

Orchestrator name is Service Fabric.

**Container Clusters:**

**Kubernetes** Cluster – Orchestrator name is Kubernetes (K8S).

Docker **Swarm** Cluster – introduced by google

DC/OS Cluster – orchestrator name is DC/OS

Kubernetes is orchestrator running in Cluster.

Kubernetes CLI (kubectl.exe)

**Managed Cluster (MC)**

There are multiple services are created like

Nsg – network security group

Routetable – route table

Vmss - Virtual machine scale set

VM – virtual machine

Load balancer

**POD**

**ReplicaSet**

It used to maintaining the number of instances.

**Service – load balancer**

Communicator between client and POR/app

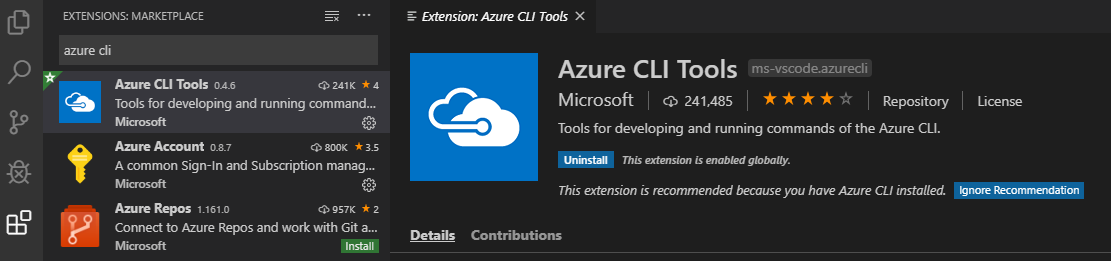
**Label selector** is used to map the service and node/container

**To Create Cluster:**

Step 1 Need to install azure cli in local system

azure-cli

step 2 Open VS code and add extension



**Open the Api or Web application in VS code**

**Create DockerFile and .dockerignore file**

**Create Image/container form vs code run the below command**

>> docker build -t catalogapi .

Go to hub.docker.com and create an account

**Login to docker hub using command promp:**

**>>** docker login

To change the name

>> docker tag server:latest myname/server:latest

**To push docker image on docker hub:**

**>>** docker push amolpatole/catalogapi:latest

**Kubernetes CLI (kubectl.exe) to install**

**>> az aks install-cli**

**>> kubectl version**

**>> kubectl get nodes**

**>> kubectl get svc**

**To create YAML need to install extension YAML**

**To deploy catalog api using yaml**

* **Open cmd from yaml directory and run below command**

**>> kubectl apply -f ./catalogapi-deploy.yaml**

**To delete deployment**

**>> kubectl delete deploye catalogapi-deploy.yaml**

**To view the list of deployments**

**>>** **kubectl get deploy**

**To view the replica sets**

**>> kubectl get rs**

**To get the list of PODS**

**>>** **kubectl get pods**

**To describe the pod information**

**>> kubectl describe pod <pod-name>**

**To view the pod console**

**>> kubectl log <pod-name>**

**Kubernetes Service Types:**

Load Balancer

Service with public IP and private IP

To access service by publicly

Cluster IP

Service without Public IP

**To deploy load balancer service**

>> kubectl apply -f “catalogapi-service.yaml”

**To get created service file**

>> kubectl get svc

az aks browse --n <cluster-name> -g <group-name>

**API Gateway Management:**

Also called APIM.

It is a middleware component in between client and API app

API deployed in:

API app

Azure VM

ACI

K8S

Fabric

**Management Features and functionalities**

Security

Enable xml/JSON transformations post deployment

Request limiting (controlling the request from same user)

Mocking (dummy response)

Versioning

Replacing text

Change the request and response header.

**Components:**

**Publisher Portal Policies:**

The configuration applied on request/response is called policy.

Policy is the set of configuration done on APIs

Azure portal itself is used ad publisher portal

**Developer Portal:**

API consumers (client app developers) subscribe for the APIs

Once subscribed they will get a subscription key.

Key must be sent to API gateway along with request in headers

Header **Ocp-Apim-Subscription-Key** is used to send the key.

**Gateway Portal:**

API configuration are done in gateway.

Monitoring is done in gateway.

Gateway receives the request from the client and forward it to API.

Gateway forward the responses sent by the API to the client.

**Product:** Set of API collection

**API:** Set of Operation/function collection

**Operations:** single function

**Policies:**

Backend URL

Mocking

Request limit by key

IP filter

Set header

CORS

Find and replace

**Logic Apps:**

It is a **PaaS** service used for application integration.

You can integrate multiple SaaS solution in a single workflow.

Used to create to workflow.

A workflow contains one trigger and multiple actions.

A logic app uses Connectors to connect to the SaaS solutions.

**Types of triggers:**

Push triggers

Http URL request trigger

Pull triggers

Mail connectors

Recurring triggers

Timer trigger

**ACI – Azure Container Instance**

<https://Aksworkshop.io> for virtual nodes demo

This is a server less computing service.

Azure Container Instances offers the fastest and simplest way to run a container in Azure

Container deployment options in Azure

Web App for Containers

App service plan is created. (Pre provision resources)

One app per service. (Single container deployment)

AKS

Cluster based deployment

Multiple apps within the cluster service.

Orchestrator required – Kubernetes.

Service Fabric

Cluster based deployment.

Only for micro services.

Containerized apps need to deploy as SF project.

ACI – Azure Container Instances

Server less architecture

No need to provision any resource before deployment.

Single container deployment.

**Azure Function App:**

Server less computing solution.

Like a web job

A function starts executing with a trigger.

Timer trigger

Http trigger

Blob trigger

Queue trigger

A function app is a piece of code that performs a single operation.

We need to write the code in C#, F#, JS, PHP, Python and Java etc.

**Bindings:**

Input bindings

Output bindings

Trigger bindings (A type of input binding)

**Hosting Plans:**

Consumption plan

A function can run for a maximum of 10 min (default timeout in 5 min).

Billing is based on number of executions, execution time and memory used.

App service plan

Enable **always on** for well performance at first request.

**Function App and Function:**

x-functions-key

Function is a piece of code that does an atomic task.

Function app contains one or more

Develop function locally using Azure Function CLI

>> npm i -g azure-functions-core-tools

>> func --help

Create a new function app

>> func init

To create function inside app

>> func new

Publish the function app with the setting

>> func azure functionapp publish <functionapp-name> -i

**Azure DevOps:**

Development and operations activities happens in the same platform

Job allocation, - top managers/project managers

Priorities jobs, - top managers/project managers

Code management, - developers

Testing, - testers

Automated build and deployment – deployment engineers. (CI/CD)

SDLC

Requirement analysis

Collect client requirements

Priorities jobs

Assigning task to developers and designers.

Designing phase

ER Diagram

UML diagram

Development

Coding

Unit testing by developers

Testing

Integration testing

System testing

Load testing

UAT etc.

Bug and defect reporting

Modification

Deployment

Git

SCM – source code management

Open source

Distributed version control management

TFVC

SCM tool

MS

Centralized version control

**Git Commands:**

git branch to get list of branches

git merge feature/auth To create branch

git push -u origin –all To push all branches

**Pipelines:**

Build pipeline (CI – Continues integration)

Release pipeline (CD – continues delivery/deployment)

**Event Grid: (**Custom Topics) means real time sending notification like Facebook or WhatsApp showing online when user is logged in on app.

It is a fully managed Event Ingestion system.

Here one to many communication