Microsoft Azure

COMP6205: Web Development

Dr A. Rezazadeh
November 17
What is a “Cloud”? 

- Cloud: on-demand, scalable, multi-tenant, self-service compute and storage resources
Cloud computing

- Cloud computing provides a modern alternative to the traditional on-premises datacenter.
- A public cloud vendor is completely responsible for hardware purchase and maintenance and provides a wide variety of platform services that you can use.
- Cloud environments provide an online portal experience, making it easy for users to manage compute, storage, network, and application resources.
  - For example, in the Azure portal, a user can create a virtual machine (VM) configuration specifying the following: the VM size (with regard to CPU, RAM, and local disks), the operating system, any pre-deployed software, the network configuration, and the location of the VM.
Cloud Computing Models

On-Premises (Private Cloud):
- Data & Access
- Applications
- Runtime
- Operating System
- Virtual Machine
- Compute
- Networking
- Storage

Infrastructure (as a Service):
- Data & Access
- Applications
- Runtime
- Operating System
- Virtual Machine

Platform (as a Service):
- Data & Access
- Applications
- Runtime
- Operating System
- Virtual Machine

Software (as a Service):
- Data & Access
- Applications
- Runtime
- Operating System
- Virtual Machine
- Compute
- Networking
- Storage

Host
Develop
Consume

COMP6205 - Microsoft Azure
Cloud Terminology

- **Infrastructure as a Service (IaaS):** basic compute and storage resources
  - On-demand servers
  - Amazon EC2, VMWare vCloud

- **Platform as a Service (PaaS):** cloud application infrastructure
  - On-demand application-hosting environment
  - E.g. Google AppEngine, Salesforce.com, Microsoft Azure

- **Software as a Service (SaaS):** cloud applications
  - On-demand applications
  - E.g. Office 365, GMail, Microsoft Office Web Companions
Cloud: Efficiency Versus Control

<table>
<thead>
<tr>
<th>= Managed for You</th>
<th>Standalone Servers</th>
<th>IaaS</th>
<th>PaaS</th>
<th>SaaS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Applications</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Runtimes</td>
<td></td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Database</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Operating System</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Virtualization</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Server</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Storage</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
</tr>
<tr>
<td>Networking</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Efficiency → Control+Cost

COMP6205 - Microsoft Azure
**IaaS**

1. Choose image, then create VM for DBMS and configure DBMS
2. Create the configuration for VM(s) & applications
3. Provision database, then create tables and add data
4. Install application
5. Configure load balancer
6. Manage VMs and DBMS (e.g., deploying new OS images in VMs)

**Diagram**

- **Library**: VM Images
- **Data**: DBMS
  - **Application**: Web Server
    - **Operating System**: VM
- **Load Balancer**: 

**COMP6205 - Microsoft Azure**
PaaS

1) Provision database, then create tables and add data

2) Deploy application

- Data
  - DBMS
  - Operating System
  - VM
- Application
  - Web Server
  - Operating System
  - VM
- Load Balancer

Developer
The Azure portal

• An online management portal provides the easiest way to manage the resources you deploy into Azure.

• You can use this:
  – to create virtual networks,
  – create VMs,
  – define storage accounts, and so on,
  – set up Web Apps,

• The portal address:
  – https://portal.azure.com/
Azure services
Azure services

• Azure includes many services in its cloud computing platform. Let’s talk about a few of them.

• **Compute services:** This includes the Azure Virtual Machines—both Linux and Windows, Cloud Services, App Services (Web Apps, Mobile Apps, Logic Apps, API Apps, and Function Apps), Batch (for large-scale parallel and batch compute jobs), RemoteApp, Service Fabric, and the Azure Container Service.

• **Data services:** This includes Microsoft Azure Storage (comprised of the Blob, Queue, Table, and Azure Files services), Azure SQL Database, DocumentDB, StorSimple, and the Redis Cache.
Azure services – Cont.

- **Application services**: This includes services that you can use to help build and operate your applications, such as Azure Active Directory (Azure AD), Service Bus for connecting distributed systems, HDInsight for processing big data, Azure Scheduler, and Azure Media Services.

- **Network services**: This includes Azure features such as Virtual Networks, ExpressRoute, Azure DNS, Azure Traffic Manager, and the Azure Content Delivery Network.
Azure Resource Manager

• This is called the Resource Manager deployment model

• This is the Azure mechanism for creating, running and controlling application and resources.

• These deployment models are often referred to as *control planes* because they are used to control services, not just to deploy them.

• This is different from a data plane, which manages the data used by a service.

• Typically, your running Azure infrastructure will contain many resources, related to one another in some way.
An Example

- To run a web application you need to have two VMs running the web application, using a database to store data, and residing in the same virtual network.

- With Resource Manager, you deploy these assets into the same resource group and manage and monitor them together.

- You can deploy, update, or delete all of the resources in a resource group in one operation.

- In this example, the resource group would contain the following:
  - VM1, VM2, Virtual network, Storage account and the Azure SQL Database
App Service and App Service plans

• An App Service plan defines the capacity and resources to be shared among one or more app services that are assigned to that plan.

• The following are some of the criteria you can define when creating an App Service plan.

  – Location (such as Europe), Number of cores or instance size, amount of memory, amount of storage, maximum number of instances, auto-scaling options (depends on tier—automatic, manual, or none)

• Pricing tier (such as Free, Standard, or Premium) providing distinct settings for a variety of performance and service capabilities:
How does this help you?

• For example, let’s say you have **five** websites and **three** mobile apps that you want to host.

• You could run each one on its own VM, which would require **8** VMs.

• If you wanted redundancy (recommended), that would require **16** VMs. Even if you select small instances, the cost adds up really quickly.

• If you could run those eight applications on the same set of **two** VMs, it would be more cost-effective and easier to manage.

• This is what using App Service plans does for you.
Different type of Apps

- The App Service is a service that hosts one of five kinds of applications:
  - Web Apps:
    - For hosting websites and web applications.
  - Mobile Apps
    - For hosting mobile app back ends.
  - API Apps
    - For hosting RESTful APIs.
  - Logic Apps
    - For automating business processes and integrating systems and data across clouds without writing code.

Microsoft Azure Web Apps

• Quickly and easily deploy sites to a highly scalable cloud environment with the frameworks and open source apps of your choice by using Microsoft Azure websites

• Supports multiple frameworks (ASPNET, Classic ASP, PHP, Node.js)

• Pick from popular open source apps

• Pick your DB (SQL Azure, MySQL)

• Choose your tools (Visual Studio, Git, FTP, WebMatrix)

• Build on any platform (Windows, Mac, Linux)
Options for Creating Web Apps

- There are multiple options for creating a Web App and deploying the content to an app service. Let’s look at a few of these, including the following.

- **Azure Marketplace**: This contains all of the resources you can deploy in Azure. I’ll show you how you can use this to create Web Apps from pre-existing templates such as WordPress.

- **Visual Studio Code**: This is a free, open source, cross-platform code editor with debugging capabilities.

- **Visual Studio**: This is Microsoft’s full-featured development IDE.
Marketplace

There are many pre-created websites and templates in the Azure Marketplace that you can use.

- **Blogs + CMSs**: Joomla!, Drupal, DNN, Orchard CMS, Umbraco CMS, and MonoX
- **Starter Web Apps**: ASP.NET, HTML5, Node.js, PHP, Apache Tomcat, and some examples like the Bakery web app and the Java Coffee Shop web app
Mobile Apps

- Azure Mobile Apps, included as part of Azure App Service, is a backend as a service that provides multiple features to make it easier and quicker to create a mobile application.

- Mobile Apps is both flexible and scalable, so when your application becomes widely used, you can scale appropriately to handle your customers’ needs.

- Another advantage of Azure Mobile Apps is that you only have to write one version of your backend.

- The backend can be used by devices running iOS, Android, and Windows, allowing you to reach every user on every platform without extra work.
Features Provided by Azure Mobile Apps

• **Data storage** You can choose for your data storage to be powered by SQL Database, which has an interface simple enough to use without being a DBA. You can also integrate with SQL Server, Azure Table Storage, MongoDB, DocumentDB, or via an API to software as a service (SaaS) providers such as Salesforce.com and Office 365.

• **Push notifications** You can send information for customer and enterprise applications to any customer’s mobile device by using Microsoft Azure Notification Hubs.

  – Notification Hubs automatically handles the server-side code to push messages to the push notification services for iOS, Android, and Windows devices.
Features Provided by Azure Mobile Apps

• Because Mobile Apps runs in Azure, you can easily scale in and out to meet customer demand
  – You can even set up autoscaling that will automatically scale out as demand increases, handling millions of devices.

• You can use Microsoft Azure WebJobs to perform backend processing on the server at a scheduled time
  – For example, you might want to create a scheduled job that requests an update from your on-premises database and stores the new information in a table, waiting to be retrieved by your mobile application.
Microsoft Azure Virtual Machines

Getting started
- Management Portal
- Scripting (Windows, Linux, and Mac)
- REST API

Select image and VM size
- Windows Server 2012
- CentOS
- openSUSE
- Ubuntu
- ...and many more!

New disk persisted in storage
- Boot VM from new disk
- Blob storage
- Cloud

Virtual Machine Scale Sets

• Virtual machine scale sets are an Azure Compute resource you can use to deploy and manage a set of identical VMs.

• You can create a VM Scale Set in the Azure portal by selecting new and typing in "scale" in the search bar. You will see "Virtual machine scale set" in the results. From there you can fill in the required fields to customize and deploy your scale set.
  
  – VM scale sets can also be defined and deployed using JSON templates and REST APIs just like individual Azure Resource Manager VMs.
Creating and Managing VM Scale Sets

• Scaling a VM scale set out and in
  – To increase or decrease the number of virtual machines in a VM scale set, simply change the capacity property and redeploy the template.

• Monitoring your VM scale set
  – It shows basic properties and operations, including listing VMs in the set and a resource usage graph. For more detail you can use the Azure Resource Explorer to view VM scale sets. VM scale sets are a resource under Microsoft.Compute, so from this site you can see them by expanding the following links:
    – Subscriptions -> your subscription -> resourceGroups -> providers -> Microsoft.Compute -> virtualMachineScaleSets -> your VM scale set -> etc.
Data Management

• A persistent data store is at the heart of many applications.

• As you migrate existing applications to the Azure cloud or create new applications, you will likely find yourself needing to interact with a database.

• The Azure platform provides several options from which to choose. You can choose from relational database offerings such as Azure SQL Database, SQL Server running in Azure Virtual Machines, or non-Microsoft databases such as Oracle or MySQL.

• If a non-relational (or NoSQL) database fits your application needs better, services such as DocumentDB and Azure Table Storage might be a good fit.
SQL Database

- Cloud relational database based on SQL Server
- Use the same tools, data access frameworks, T-SQL-based Language
- Provides high availability and redundancy
- Reads are completed at the primary
- Writes are replicated to all secondaries.
SQL Data Sync

Hybrid applications; one-way publish or two-way sharing

Multiple locations (for example, branch office, retail offices); share data between locations, and/or aggregate data in the cloud

On-premises and cloud

Cloud and cloud

Geo-located web applications
Use with Microsoft Azure Traffic Manager
Scale-out via multiple copies of data (for example, separate reporting and OLTP workloads; multiple websites)
Networking

Connects Azure services to a customer on-premises network via private circuits or the Internet.

- Predictable performance
- Security
- High throughput
- Lower cost
## A Full List of Services

https://docs.microsoft.com/en-gb/azure/index#pivot=services&panel=all

<table>
<thead>
<tr>
<th>Compute</th>
<th>Networking</th>
<th>Storage</th>
<th>Web + Mobile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux VMs</td>
<td>Virtual Network</td>
<td>Storage</td>
<td>App Service</td>
</tr>
<tr>
<td>Windows VMs</td>
<td>Load Balancer</td>
<td>Data Lake Store</td>
<td>Web Apps</td>
</tr>
<tr>
<td>VM Scale Sets</td>
<td>Application Gateway</td>
<td>StorSimple</td>
<td>Mobile Apps</td>
</tr>
<tr>
<td>App Service</td>
<td>VPN Gateway</td>
<td>Backup</td>
<td>API Apps</td>
</tr>
<tr>
<td>Azure Container Service</td>
<td>Azure DNS</td>
<td>Site Recovery</td>
<td>Logic Apps</td>
</tr>
<tr>
<td>Azure Container Registry</td>
<td>Traffic Manager</td>
<td>Internet of Things</td>
<td>CDN</td>
</tr>
<tr>
<td>Functions</td>
<td>CDN</td>
<td>IoT Developer Center</td>
<td>Media Services</td>
</tr>
<tr>
<td>Service Fabric</td>
<td>ExpressRoute</td>
<td>Azure IoT Hub</td>
<td>Search</td>
</tr>
<tr>
<td>Batch</td>
<td>Intelligence + Analytics</td>
<td>Azure IoT Suite</td>
<td>Mobile Engagement</td>
</tr>
<tr>
<td>Cloud Services</td>
<td>HDInsight</td>
<td>Event Hubs</td>
<td>Notification Hubs</td>
</tr>
<tr>
<td>Databases</td>
<td>Machine Learning</td>
<td>Notification Hubs</td>
<td></td>
</tr>
<tr>
<td>SQL Database</td>
<td>Stream Analytics</td>
<td>Stream Analytics</td>
<td></td>
</tr>
<tr>
<td>SQL Data Warehouse</td>
<td>Cognitive Services</td>
<td>Machine Learning</td>
<td></td>
</tr>
<tr>
<td>SQL Server Stretch Database</td>
<td>Data Lake Analytics</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DocumentDB</td>
<td>Data Lake Store</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Redis Cache</td>
<td>Data Factory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Factory</td>
<td>Power BI Embedded</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security + Identity</td>
<td>Analysis Services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Security Center</td>
<td>Bot Service</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Key Vault</td>
<td>Developer Tools</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Azure Active Directory</td>
<td>Visual Studio Team Services</td>
<td>Monitoring + Management</td>
<td></td>
</tr>
<tr>
<td>Azure Active Directory B2C</td>
<td>Azure DevTest Labs</td>
<td>Application Insights</td>
<td></td>
</tr>
<tr>
<td>Azure Active Directory Domain Services</td>
<td>Application Insights</td>
<td>Automation</td>
<td></td>
</tr>
<tr>
<td>Multi-Factor Authentication</td>
<td>API Management</td>
<td>Backup</td>
<td></td>
</tr>
<tr>
<td>Security Information</td>
<td>HockeyApp</td>
<td>Site Recovery</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scheduler</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Traffic Manager</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Azure Monitor</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Billing</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Azure Resource Manager</td>
<td></td>
</tr>
</tbody>
</table>
Deploying ASP.NET Core 2.0 Application in Azure

• There are multiple ways of deploying Core MVC web application to Azure web apps, like:
  – through Azure web portal, Azure CLI and Repository Stacks, provided to programmers of ASP.NET.
  – we can also deploy application using Visual Studio.

• We will be using Visual Studio 2017 for performing deployment.
Prerequisites

• For accomplishing above task, we need below set up as requirement,

1. For creating a web app, we need Microsoft Azure Account. If you don’t have one, create your free trial account by clicking here.


3. And .NET Core 2.0 – Download .NET Core 2.0
Deploying ASP.NET Core 2.0 Application in Azure

• Create a web app
  – In the Visual Studio Start Page, select File > New > Project...

• Complete the New Project dialog:
  – In the left pane, select .NET Core.
  – In the center pane, select ASP.NET Core Web Application.
  – Give a valid name to the application such as, ‘WebMVCCore2’.
  – Click OK to move ahead with next template selection screen.
Create a Web App
MVC Web Application Creation
ASP.NET Core 2 APP

• Make sure, you select ’ASP.NET 2.0’ from dropdown available to choose framework.

• The Change Authentication dialog appears.
  – Select Individual User Accounts.
  – Select OK to return to the New ASP.NET Core Web Application, then select OK again.
Run The App Locally

• Visual Studio creates the solution
  – Choose **Debug** then **Start Without Debugging** to run the app locally.
  – Click the **About** and **Contact** links to verify the web application works.
• Select **Register** and register a new user. You can use a fictitious email address. When you submit, the page displays the following error:

• "**Internal Server Error: A database operation failed while processing the request. SQL exception: Cannot open the database. Applying existing migrations for Application DB context may resolve this issue.**"

• Select **Apply Migrations** and, once the page updates, refresh the page.
Deploying to Azure Web Apps

• Now let’s deploy this ASP.Net Core MVC web application to Microsoft public Cloud i.e. Microsoft Azure Web Apps.

• In Solution Explorer, right click on project and click on ‘Publish’, as shown below,
Publish

• Clicking on, ‘Publish’ will open up below window.
  – We will be using ‘Web Deploy’ deployment method.
  – As we are pushing this application for the first time, Select ‘Microsoft Azure App Service’,
  – selecting ‘Create New’ option and click ‘Publish’ as highlighted below.
Creating new Azure Web App for Deploying ASP.NET Core MVC Web App

- **App Name**: Enter a unique name
- **Subscription** – Your Azure subscription
- **Resource Group** - You can select existing resource group or create new.
- Here I have created resource group with name ‘RND’,
Creating Required Services for Your App

- Select the **Services** tab on the previous page to create a new database.

Select the green + icon to create a new SQL Database
Configure Your SQL Database

- Select **New...** on the **Configure SQL Database** dialog to create a new database.
- The **Configure SQL Server** dialog appears.
- Enter an administrator username and password, and then select **OK**. Don't forget the user name and password you create in this step. You can keep the default **Server Name**.
- Enter names for the database and connection string.
Configure SQL Server

- Select **OK**.
- Visual Studio returns to the **Create App Service** dialog.
- Select **Create** on the **Create App Service** dialog.
Database Connection

- Click the **Settings** link in the **Publish** dialog.
- On the **Settings** page of the **Publish** dialog:
  - Expand **Databases** and check **Use this connection string at runtime**.
  - Expand **Entity Framework Migrations** and check **Apply this migration on publish**.
- Select **Save**. Visual Studio returns to the **Publish** dialog.
Test your app in Azure

• Click **Publish**. Visual Studio will publish your app to Azure and launch the cloud app in your browser.

• **Test your app in Azure**
  - Test the **About** and **Contact** links
  - Register a new user
Update the app

- Edit the *Pages/About.cshtml* Razor page and change its contents. For example, you can modify the paragraph to say "Hello ASP.NET Core!":

```html
@page
@model AboutModel
@if(ViewData["Title"] = "About")
{
    ViewData["Title"] = "About";
}
<h2>@ViewData["Title"]</h2>
<h3>@Model.Message</h3>
<p>Hello ASP.NET Core!</p>
```

In the Explorer Right-click on the project and select **Publish...** again.
Testing Your Update

WebApplication1  Home  About  Contact

About
Your application description page.
Hello ASP.NET Core!

© 2017 - WebApplication2
Clean up

- When you have finished testing the app, go to the Azure portal and delete the app.
- Select **Resource groups**, then select the resource group you created.
Deleting Resource

- In the **Resource groups** page, select **Delete**.

- Enter the name of the resource group and select **Delete**. Your app and all other resources created in this tutorial are now deleted from Azure.
References


- Microsoft Azure documentation
  - https://docs.microsoft.com/en-gb/azure/index#pivot=services&panel=all