Module 5

Implementing Azure App Service

Module Overview

- Introduction to App Service
- Planning app deployment in App Service
- Implementing and maintaining web apps
- Configuring web apps
- Monitoring web apps and WebJobs
- Implementing mobile apps
- Traffic Manager

Lesson 1: Introduction to App Service

- Demonstration: Preparing the Azure environment for the lab and demonstrations in this module
- Overview of App Service
- Overview of Web Apps
- Overview of Mobile Apps
- Overview of Logic Apps
- Overview of API Apps
- Overview of the App Service Environment

Azure App Service

Build and scale great cloud apps

=

- Web Apps
- Mobile Apps
- Logic Apps
- API Apps

App Service - one integrated offering

Web Apps
Web apps that scale with your business

Mobile Apps
Build Mobile apps for any device

LOGIC Apps
Automate business process across SaaS and on-premises

API Apps
Easily build and consume APIs in the cloud

Demonstration: Preparing the Azure environment for the lab and demonstrations in this module

To prepare the lab environment for this module, you must run the Setup-Azure Windows PowerShell module

Overview of App Service

Compute

- Service Fabric
- Container Service
- Azure Virtual Machines
- Azure Cloud Services

Networking

- Virtual Network
- Azure DNS
- Application Gateway
- Traffic Manager
- ExpressRoute
- Load Balancer

Data & Storage

- Storage
- DocumentDB
- Azure SQL Database
- StorSimple

Web & Mobile

- Web Apps
- Mobile Apps
- Notification Hub

Other services

- Service Bus
- Azure AD
- Azure AD DS
- MFA
- Automation
- Scheduler
- Azure Backup
- Site Recovery
- Key Vault
- Azure Security Center

Overview of Web Apps

Web Apps concepts:

- Gallery applications
- Auto scaling
- Continuous integration and deployment
- Deployment slots
- Testing in production
- Azure WebJobs
- Hybrid connections
- Azure virtual network integration
- Authentication and authorization

Overview of Mobile Apps

Requirements of mobile apps:

- Store and access structured data
- Receive notifications for alerts and updates from the cloud
- Authentication and authorization
- Defined business logic

Mobile Apps is not the same as Azure Mobile Services

You can move mobile apps from Mobile Services to Mobile Apps by using:

- Migration
- Upgrade

Overview of Logic Apps

Logic apps integrate your apps with cloud-based apps by using connectors

Core connectors:

- Office 365 Connector
- Microsoft OneDrive Connector
- Microsoft Yammer Connector
- Facebook Connector
- HTTP Connector

Enterprise integration connectors:

- SAP
- Oracle
- DB2
- Informix

Overview of API Apps

Features of API Apps:

- Visual Studio integration
- Simple access control
- Easy consumption
- Bring your existing API as is
- Use Swagger metadata
- Enable cross-origin resource sharing
- Trigger actions

Overview of the App Service Environment

The App Service Environment consists of:

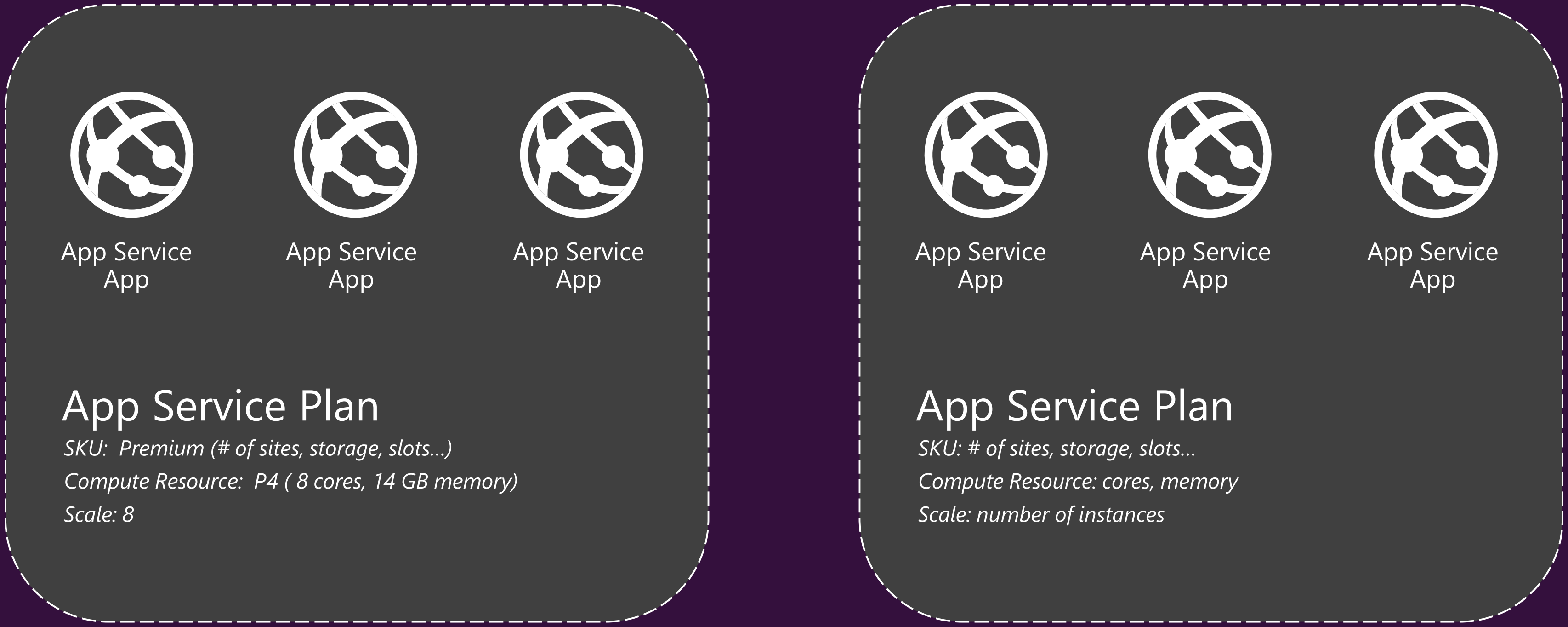
- A front-end pool with premium compute resources
- Up to three worker pools with premium compute resources
- Dedicated 500 GB of storage
- A database that stores the configuration information
- A virtual network
- A subnet for allocating IP addresses to apps
- Up to 10 virtual IP addresses

App Service Environment (ASE)

- New Premium Tier Feature
- Dedicated compute resources and network resources
- Increased Scaling Options
- Directly created in a Virtual Network
- Support all Web App features and capabilities
- Support Web, Mobile and API Apps
- Global Scale

Microsoft Azure

Conceptual Model



App Service Plan

SKU: Premium (# of sites, storage, slots…)

Compute Resource: P4 (8 cores, 14 GB memory)

Scale: 8

App Service Plan

SKU: # of sites, storage, slots…

Compute Resource: cores, memory

Scale: number of instances

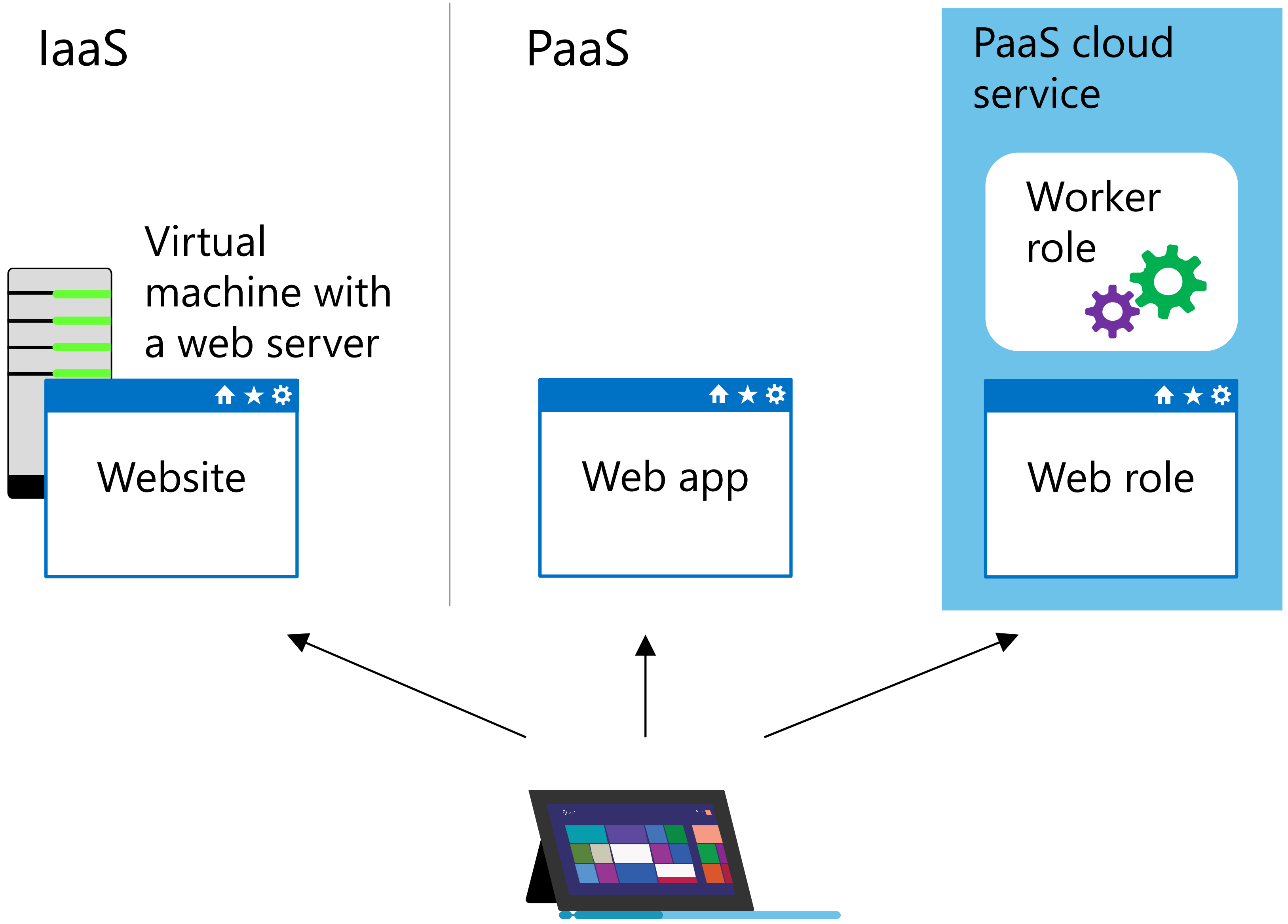
Note: For App Service Environment the SKU will always be Premium

Microsoft

Lesson 2: Planning app deployment in App Service

- Comparing web apps, PaaS cloud services, and virtual machines
- Managing App Service plans
- Comparing app-deployment methods in App Service

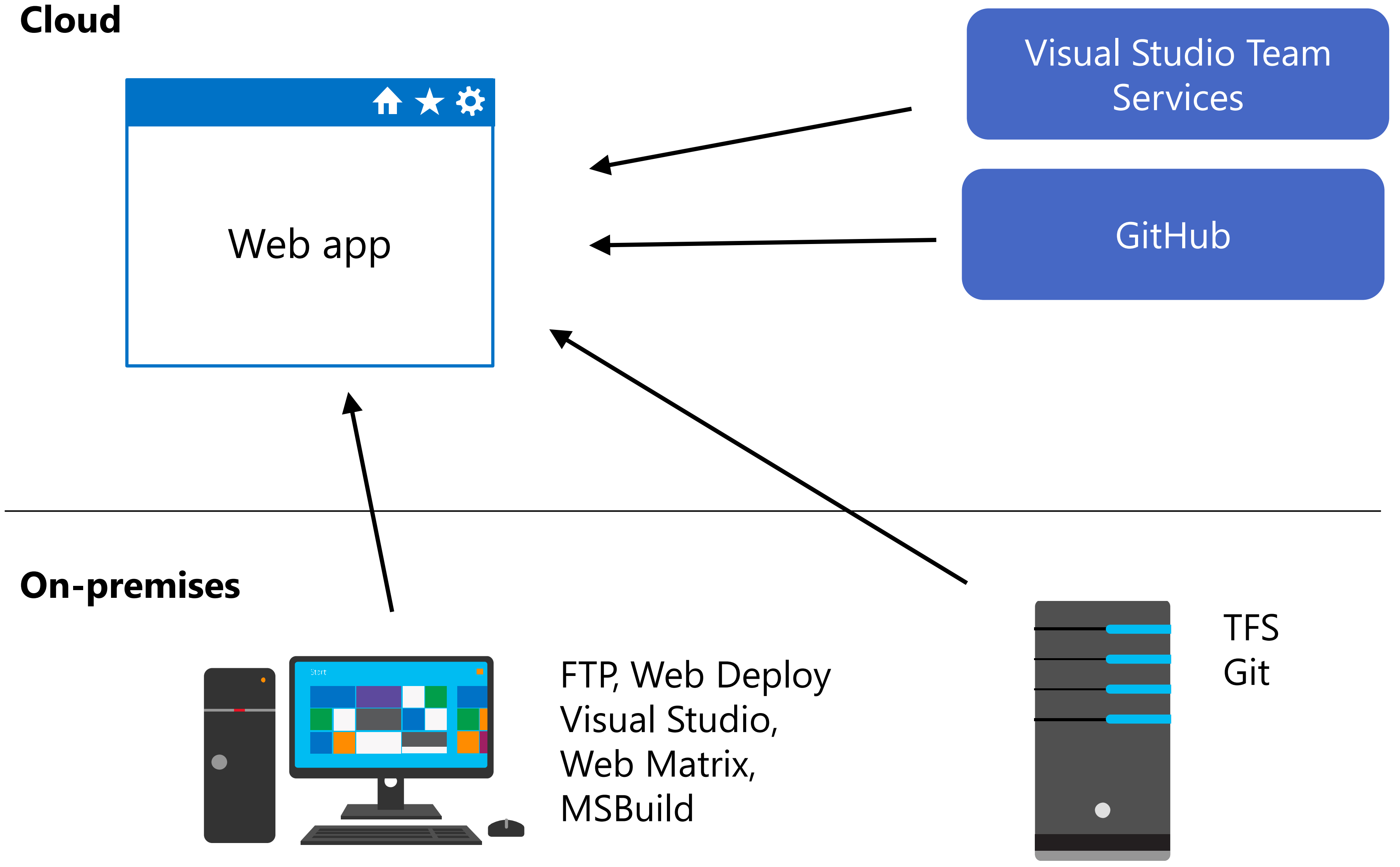
Comparing web apps, PaaS cloud services, and virtual machines



Managing App Service plans

Tier	Free	Shared	Basic	Standard	Premium
Websites	10	100	Unlimited	Unlimited	Unlimited
Storage	1 GB	1 GB	10 GB	50 GB	500 GB
Compute instance	Shared	Shared	Dedicated	Dedicated	Dedicated
Custom domains	No	Yes	Yes	Yes	Yes
SSL for custom domains	No	No	Yes	Yes	Yes
Integrated load balancer	No	Yes	Yes	Yes	Yes
Always On	No	No	Yes	Yes	Yes
Staged publishing	No	No	No	Yes	Yes
SLA	None	None	99.9%	99.9%	99.9%

Comparing app-deployment methods in App Service



Lesson 3: Implementing and maintaining web apps

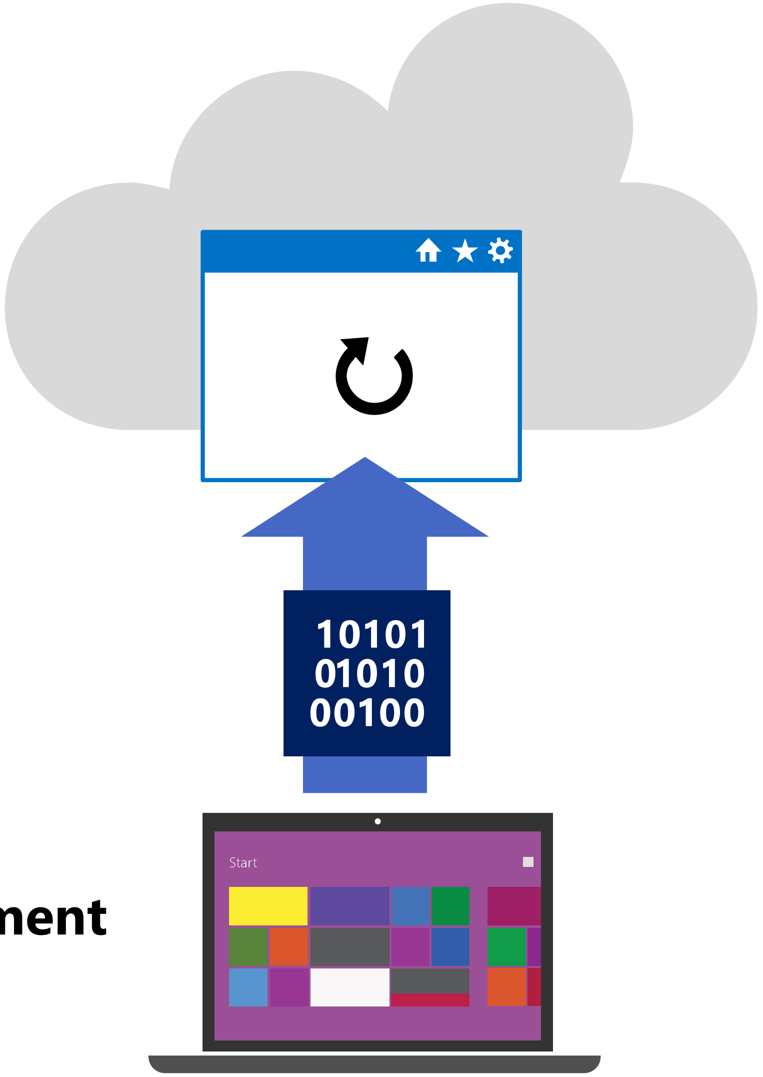
- Creating web apps
- Deploying web apps
- Updating web apps
- Demonstration: Deploying web apps by using Web Deploy

Creating web apps

- Create new web apps in Azure by using:
  - The Azure portal
  - **New-AzureRMWebApp**
- Set up deployment credentials that will be:
  - Used by FTP and Git
- Download a publishing profile:
  - The publish profile includes:
    - Deployment credentials
    - Database connection strings
    - Details for Web Deploy and FTP deployment
  - Developers can import the publish profile into Visual Studio, and deploy it to Azure

Deploying web apps

- Advantages of Web Deploy:
  - Only the changes are uploaded
  - Transfers use HTTPS
  - Permissions can be set on files
  - Databases can be published
  - Connection strings can be set
- You can use MSDeploy.exe in:
  - Visual Studio
  - WebMatrix
  - PowerShell
    - **New-AzureRmResourceGroupDeployment**



Updating web apps

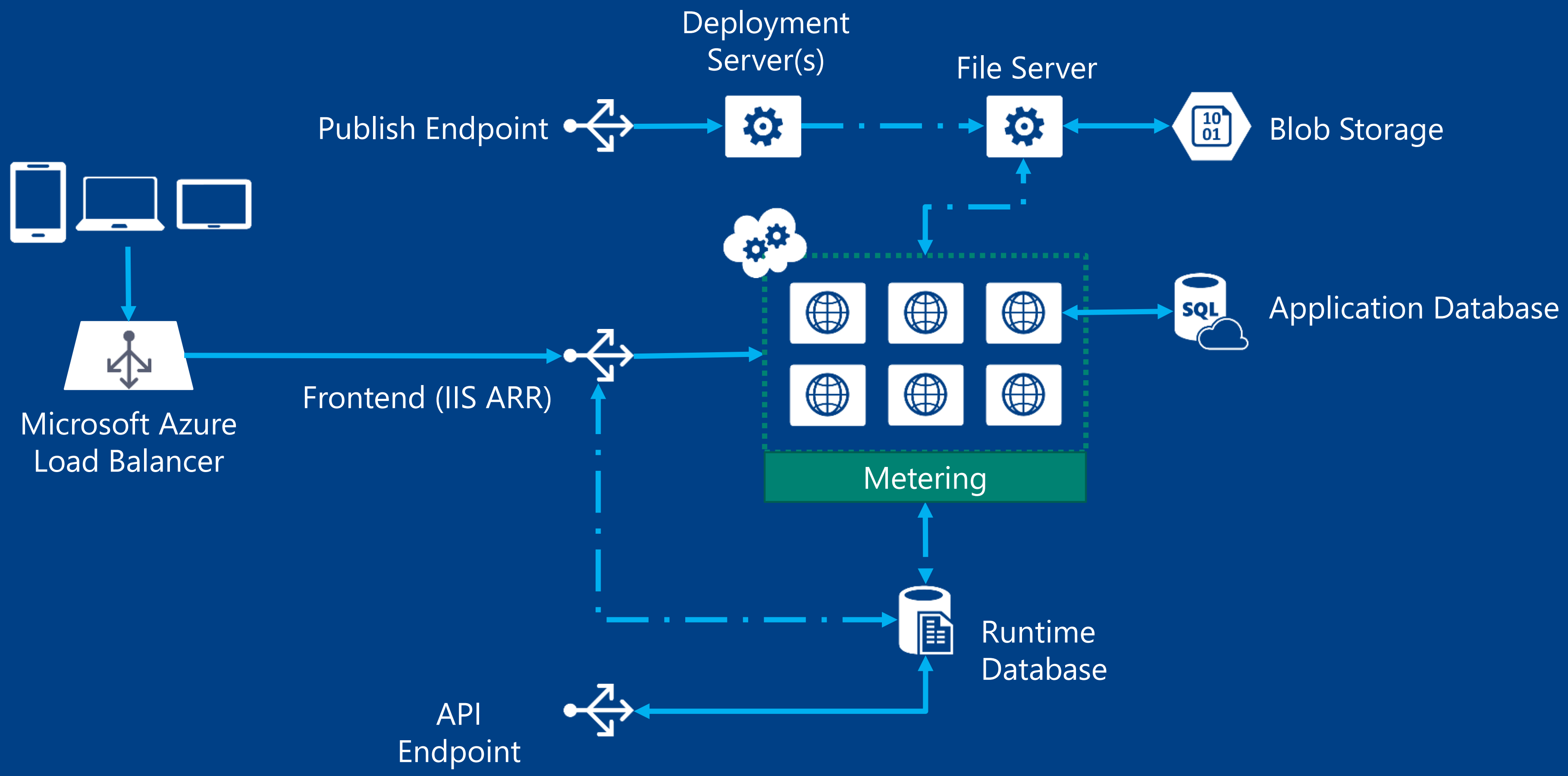
- Deploy updates by using:
  - FTP
  - Web Deploy
- Continuous deployment and delivery:
  - Connect a project to a web app in Azure
  - Check in the changes
  - Build and deploy automatically

Staging and production slots:

- Some settings change on slot swap
- Some settings do not change on slot swap
- App settings and Connection strings support “Slot setting”

Microsoft Azure

App Service Web App Architecture



Microsoft

Demonstration: Deploying web apps by using Web Deploy

In this demonstration, you will learn how to:

- Create a new web app in the Azure portal
- Browse the new web app from the Azure portal
- Obtain a publish profile
- Deploy a line-of-business application in Web Apps

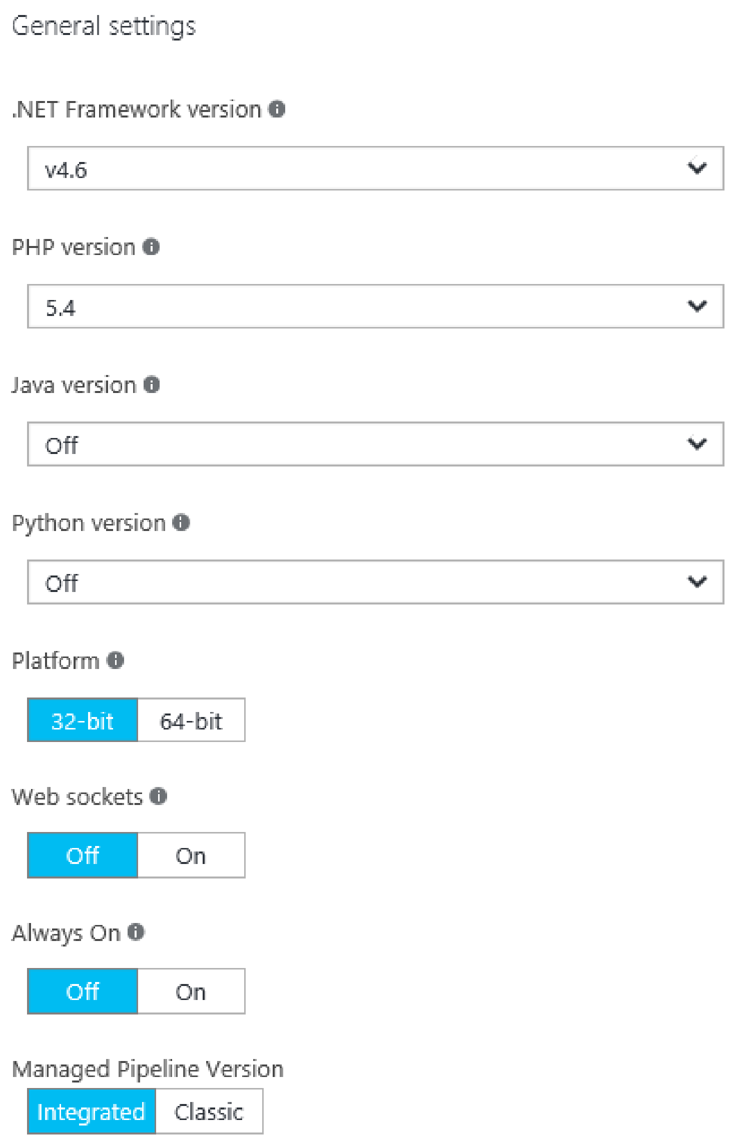
Lesson 4: Configuring web apps

- Configuring a web app’s application and authentication settings
- Configuring virtual networks and hybrid connectivity
- Configuring availability and scalability
- Implementing WebJobs
- Demonstration: Configuring web-app settings and auto scaling, and creating a WebJob

Configuring a web app's application and authentication settings

Web App settings:

- Framework versions
- Platform and Web sockets
- Always On
- Managed Pipeline Version
- Auto Swap
- Debugging
- Certificates, Domain Names, and SSL Bindings
- App Settings
- Connection Strings
- Default Documents
- Diagnostic logs
- Authentication and Authorization


General settings

.NET Framework version: v4.6

PHP version: 5.4

Java version: Off

Python version: Off

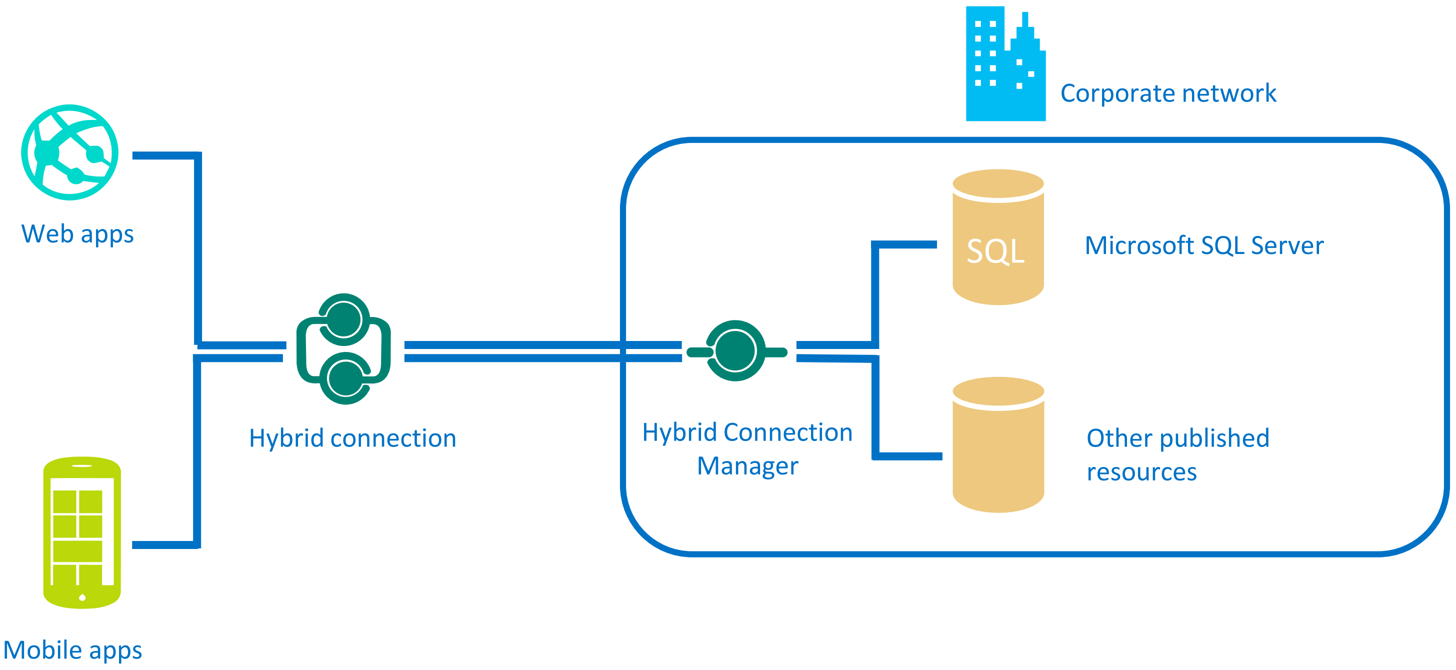
Platform: [✓ 32-bit] 64-bit

Web sockets: [✓ Off] On

Always On: [✓ Off] On

Managed Pipeline Version: [✓ Integrated] Classic

Configuring virtual networks and hybrid connectivity



Configuring availability and scalability

- Scaling a Free tier web app is not possible
- Scaling Shared and Basic tier web apps:
  - Instance size
  - Instance count
- Scaling Standard and Premium tier web apps:
  - Instance size
  - Instance count
  - Scheduled scaling
  - Scale by metric
  - Auto scaling

Implementing WebJobs

- WebJobs are scripts that run:
  - On demand
  - Continuously
  - On a configurable schedule
- WebJobs can be:
  - Batch files (.cmd, .bat)
  - PowerShell scripts (.ps1)
  - Bash shell scripts (.sh)
  - PHP scripts (.php)
  - Python scripts (.py)
  - Node.js JavaScripts (.js)

Demonstration: Configuring web-app settings and auto scaling, and creating a WebJob

In this demonstration, you will learn how to:

- Configure web-app settings
- Configure auto scaling
- Create a WebJob

Lesson 5: Monitoring web apps and WebJobs

- Monitoring web apps
- Configuring application and site diagnostics
- Using Kudu
- Demonstration: Using Kudu to monitor a WebJob

Monitoring web apps

- Access diagnostic logs by using:
  - FTP
  - Windows PowerShell
  - Azure command-line tools
- View logs in Visual Studio by using Application Insight
- Monitor web apps in the Azure portal by:
  - Adding metrics
  - Configuring alerts

Configuring application and site diagnostics

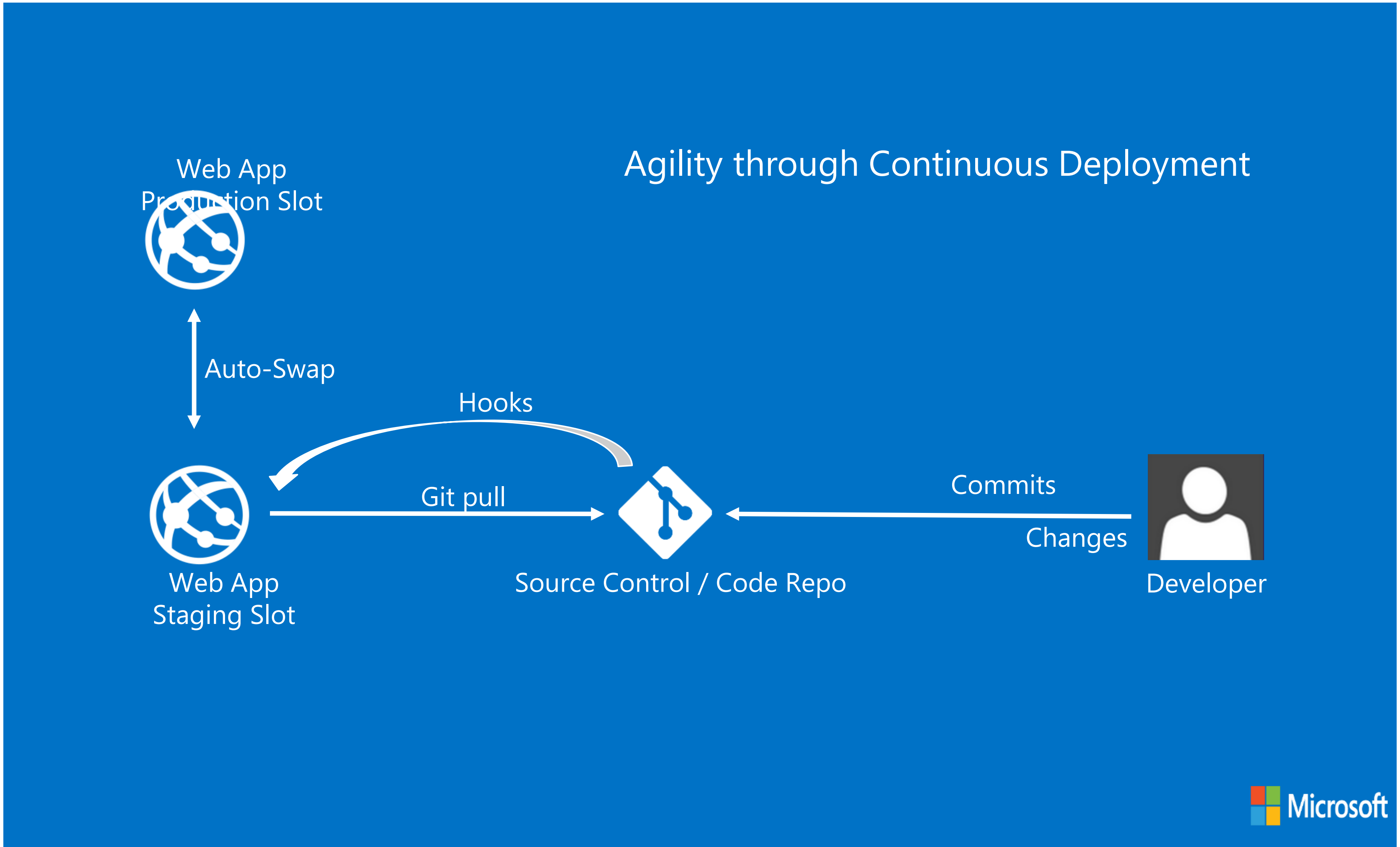
- Configure the following application logging settings:
  - Log storage location
  - Logging level
- Configure the following site diagnostics settings:
  - Web server logging
  - Detailed error messages
  - Failed request tracing

Using Kudu

- Kudu:
  - Provides Git support for web apps
  - Runs WebJobs
  - Implements a diagnostic user interface
- To access the Kudu user interface for a web app, enter https://mysite.scm.azurewebsites.net
- In the Kudu user interface, you can:
  - Run commands
  - View processes
  - Access log files
  - Add extensions

Continuous Deployment for Web Apps

Microsoft Azure



Source Control for Web/API/Mobile Apps

Git

Visual Studio Online

CodePlex

GitHub

BitBucket

DropBox

FTP

Choose your own adventure!

Demonstration: Using Kudu to monitor a WebJob

In this demonstration, you will learn how to use Kudu to monitor a WebJob’s status

Lesson 6: Implementing mobile apps

- Creating and configuring Mobile Apps
- Configuring authentication
- Deploying a mobile app
- Demonstration: Implementing a mobile app

Creating and configuring Mobile Apps

The features of Mobile Apps:

- Single sign-on
- Offline synchronization
- Push notifications
- Auto scaling
- WebJobs
- Connect to a SaaS API
- Virtual network integration
- Staging environment

Configuring authentication

- Register with a provider:
  - Azure Active Directory
  - Microsoft account
  - Facebook
  - Twitter
  - Google
- Configure authentication in the mobile app
- Cache the authentication token on the client device

Deploying a mobile app

- Using a publish profile:
  1. Download the profile from the Azure portal
  2. Import the profile into Visual Studio
  3. Complete the publishing wizard
- Using a Git repository:
  1. Install Git
  2. Create a local repository
  3. Set up the credentials
  4. Configure continuous deployment

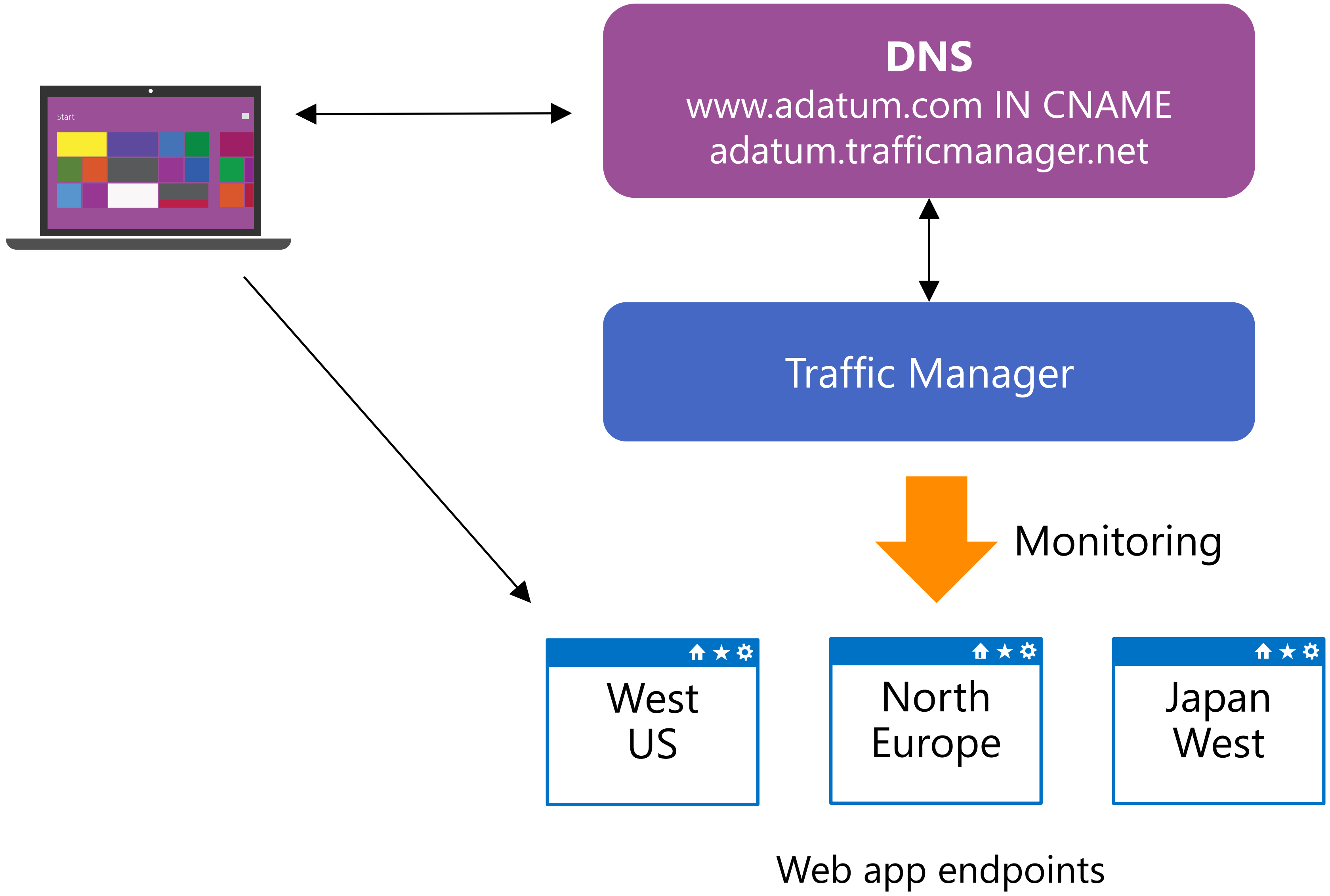
Demonstration: Implementing a mobile app

In this demonstration, you will learn how to create a new mobile app

Lesson 7: Traffic Manager

- Overview of Traffic Manager
- Configuring Traffic Manager
- Traffic Manager best practices
- Demonstration: Configuring Traffic Manager

Overview of Traffic Manager



Configuring Traffic Manager

1. Add a DNS CNAME record
2. Create a Traffic Manager profile
3. Configure a DNS prefix
4. Choose a load-balancing method:
   - Priority
   - Weighted
   - Performance
5. Add endpoints to the Traffic Manager profile
6. Configure monitoring

Traffic Manager best practices

- Be careful when changing the DNS TTL value
- Ensure that all endpoints are in the same subscription
- Use only production endpoints
- Name endpoints clearly
- Make endpoints consistent:
  - Same web app and port number
  - Same monitoring settings
- Disable endpoints for web-app maintenance

Demonstration: Configuring Traffic Manager

In this demonstration, you will learn how to:

- Create a new Traffic Manager profile
- Add an endpoint to a Traffic Manager profile by using the Azure portal
- Test Traffic Manager

Lab: Implementing web apps

- Exercise 1: Creating web apps
- Exercise 2: Deploying a web app
- Exercise 3: Managing web apps
- Exercise 4: Implementing Traffic Manager

Estimated Time: 60 minutes

Lab Scenario

The A. Datum Corporation’s public-facing web app currently runs on an IIS web server at the company’s chosen ISP. A. Datum wants to migrate this web app into Azure. You must test the Web Apps functionality by setting up a test A. Datum web app. An internal team provides you with a test web app to deploy. You must ensure that they can continue to stage changes to the test web app before deploying those changes to the public-facing site. A. Datum is a global company, so you also want to test Azure Traffic Manager, and show your organization’s decision makers how it distributes traffic to instances close to users of the web app.

Lab Review

- In Exercise 2, you deployed the A. Datum production web app to Azure. In Exercise 3, you deployed a new version of the site to a staging slot. How can you tell, within Internet Explorer, which is the production site and which is the staging site?
- At the end of Exercise 4, you used an FQDN within the trafficmanager.net domain to access your web app. How can you use your own registered domain name to access this web app?

Module Review and Takeaways

- Review Question