LIVE! 360
TECH EVENTS WITH PERSPECTIVE
The Ultimate Education Destination
ORLANDO 2024

Building and Deploying a .NET 9 App Using Azure, Bicep, and GitHub Actions

Scott Sauber
Director of Engineering
Lean TECHniques

Level: Introductory

Session Survey

- Your feedback is very important to us
- Please take a moment to complete the session survey found in the mobile app
- Use the QR code or search for “Converge360 Events” in your app store
- Find this session on the Agenda tab
- Click “Session Evaluation”
- Thank you!

LIVE! 360
TECH EVENTS WITH PERSPECTIVE

What you need

- .NET 9
- An Editor that supports .NET 9 and Bicep
  - Visual Studio 2022 Latest
  - Visual Studio Code
- Fork this repo:
  - https://github.com/scottsauber/workshop-dotnet-azure-github-bicep
- Azure will be provided for free
- Please let Scott know the following if you're participating:
  - Email you will use for Azure
  - Your GitHub username

Audience

- Anyone interested in Azure, GitHub, or Bicep
- .NET Developers
- People interested in DevOps but never got to do it

Agenda

- What is the final state of what we're building?
- What is Azure?
- What is Azure App Service? Plans?
- What is Bicep?
- What are GitHub Actions?
- Health Checks
- Azure Key Vault Integration
- Azure Application Insights Integration
- Hands on all throughout

Goals

- Learn GitHub Actions, Bicep, and Azure
- How they all integrate with a .NET app
- The feedback loop on this can be slow
- Take home a few things back to work, whether beginner or expert

Who am I?

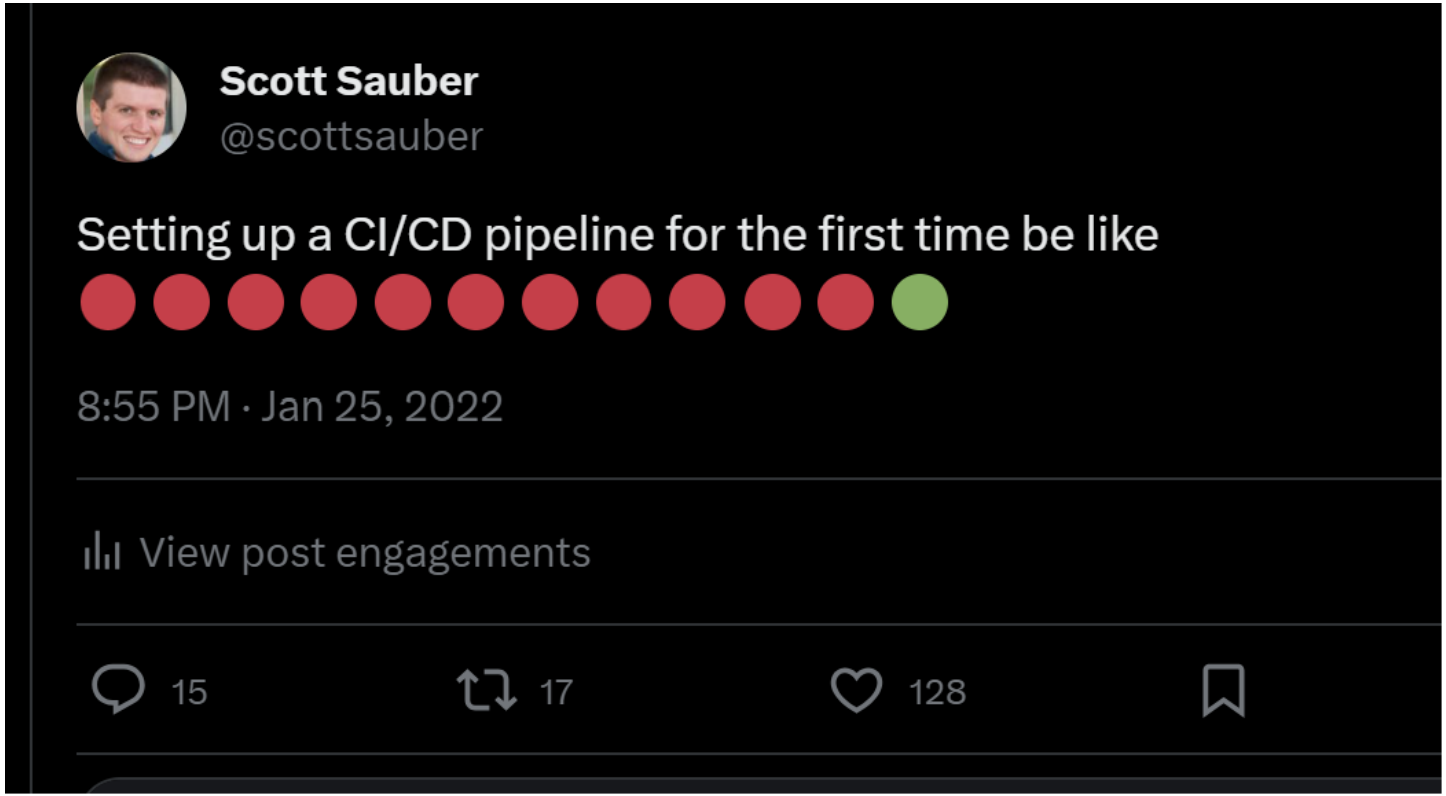
- Director of Engineering at Lean TECHniques
- Microsoft MVP
- Dometrain author
- Redgate Community Ambassador
- Co-organizer of Iowa .NET User Group

What are we building?

- .NET 9 API
- Running on Azure App Service
- Configured using Infrastructure as Code with Bicep
- Deployed via GitHub Actions

Features of what we're building

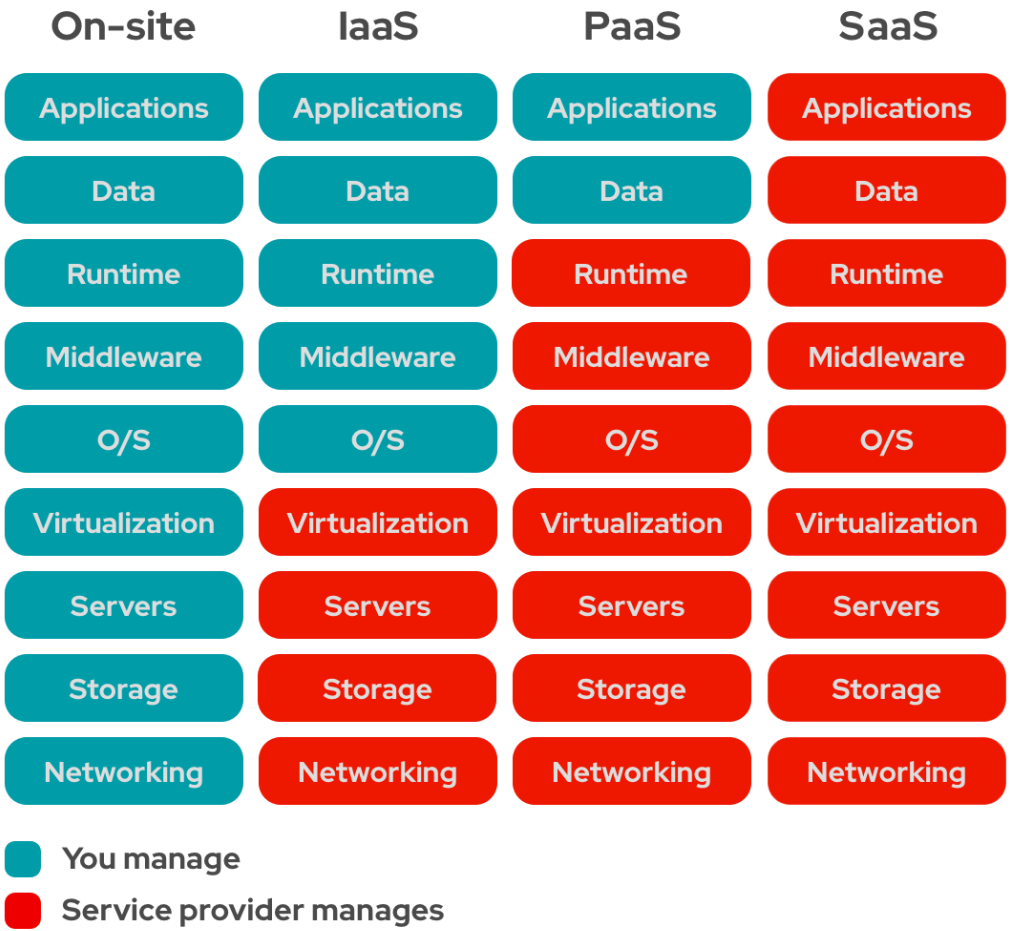
- Zero Downtime Deployments
- Infrastructure managed by code, not clicking in the Azure Portal
- Automated Build and Deploys
- Follows Azure Naming Standards
- WhatIf on PR for infrastructure changes
- Versioning your app so you know what's deployed
- Health Checks
- Secrets in Key Vault, not Source Control
- Observability using Application Insights

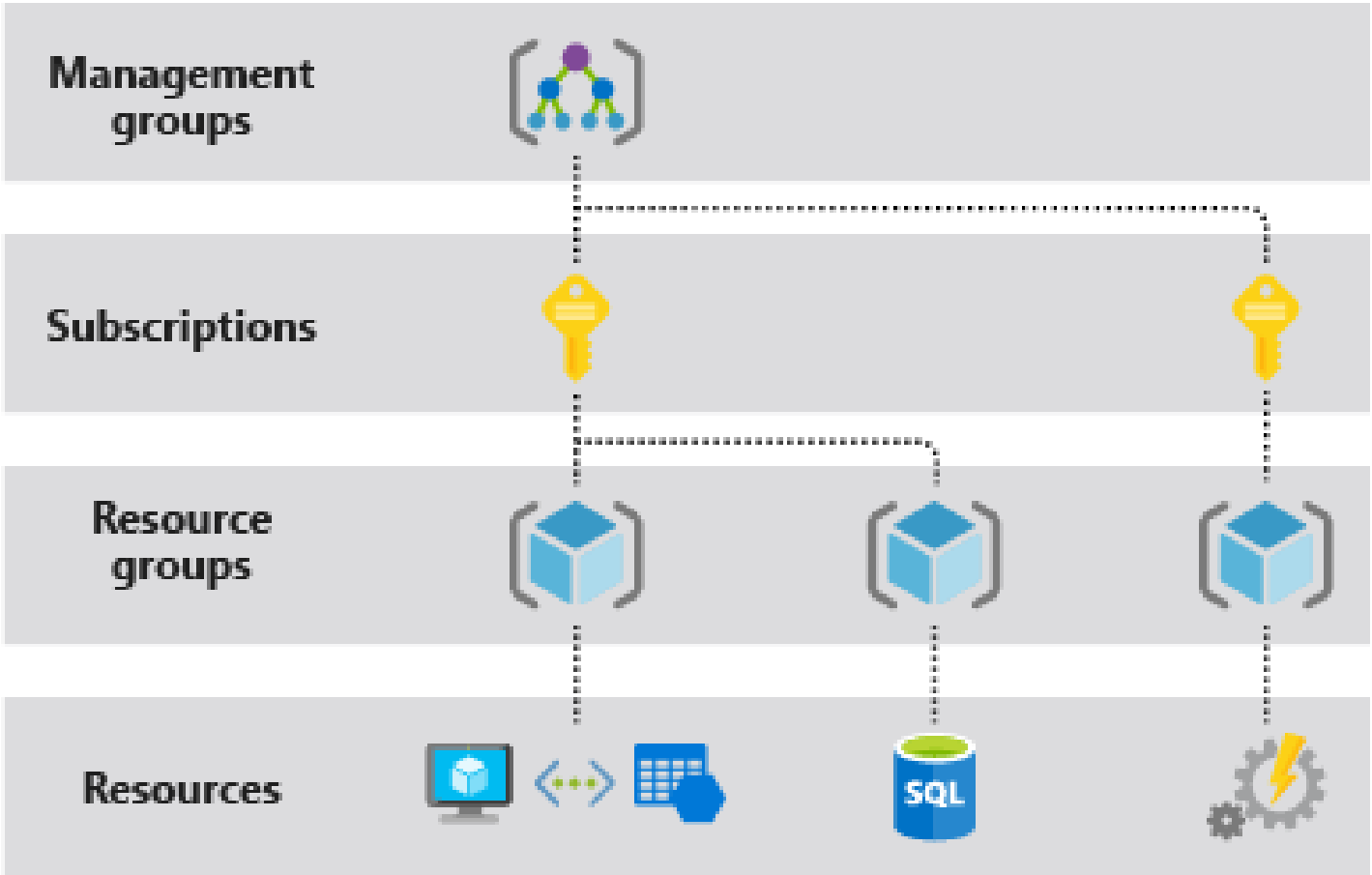


Azure

What is Azure?

- Microsoft offering for cloud hosting
- Offers many services from hosting web apps to databases to caching to messaging to…
- You should probably be picking PaaS offerings (i.e. not VMs)



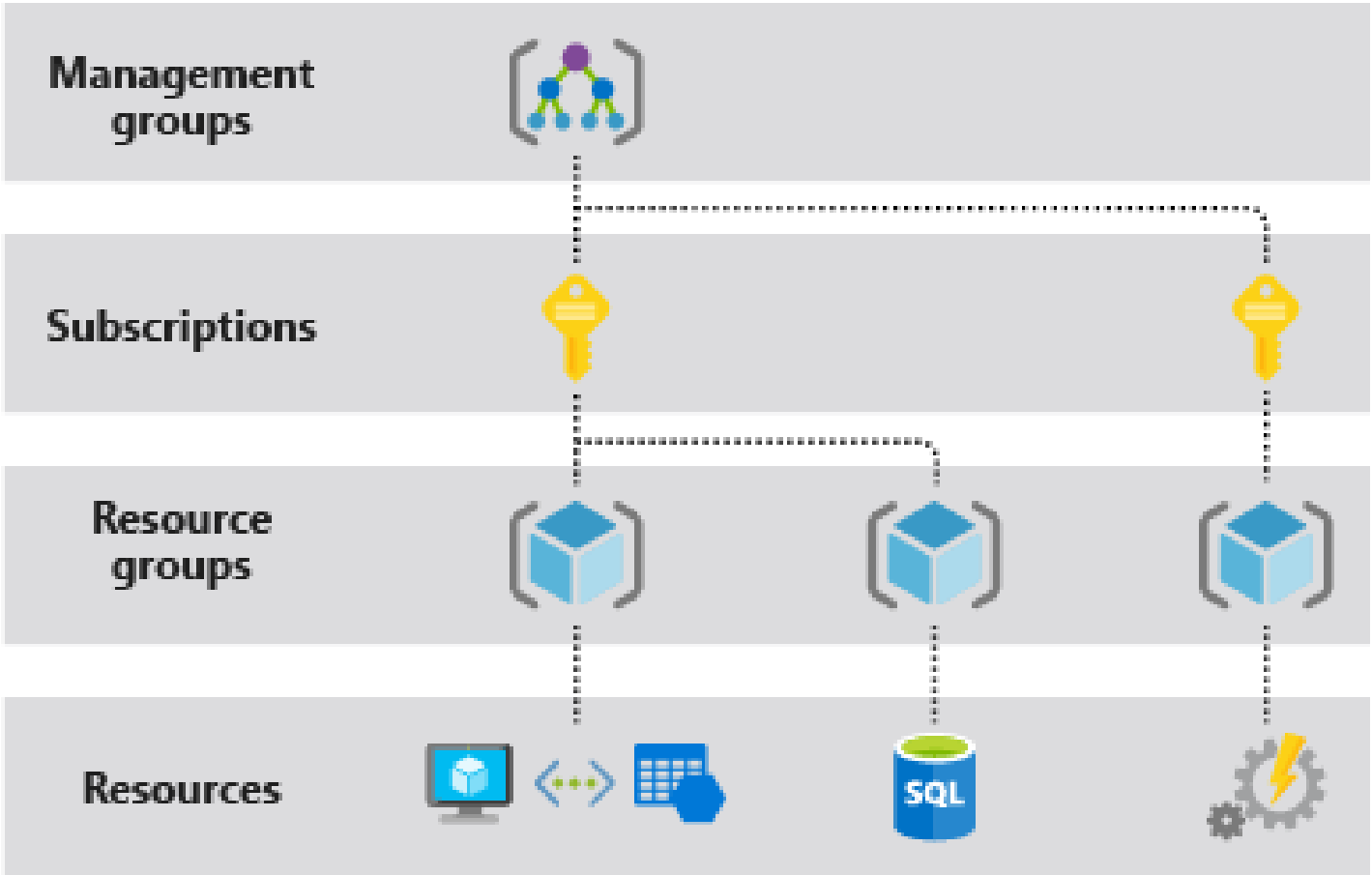


Subscriptions

- Top-ish level organization (ignoring Tenants, Management Groups)
- Recommended per team per env
- Naming convention: sub-<team>-<env>
  - Ex: sub-accounting-dev
- Role access separation
- Billing separation

Resource Groups

- Related groups of resources (ie web, db, key vault, etc)
- Quickly view all resources related to app
- RG = folder, Resources = files
- Recommended per app per env
- Naming convention: rg-<product>-<env>
  - Example: rg-fancyapp-dev
- May have many RG's in 1 subscription
- Role access separation
- Billing separation



Azure App Service

App Service

- PaaS offering for hosting applications
- ✅ Handles OS patches, Framework patches
- ✅ Zero downtime deployments via Slots
- ✅ SSL Certs
- ✅ Custom domains
- ✅ Autoscaling
- ✅ Very simple
- ✅ And more
- ❌ Less control because PaaS

App Service Plan

- Kinda like VM for your App Service(s)
- Pick how much memory, storage, CPU
- Multiple app service on one ASP (should you?)
- Many apps can get away with P0V3 ($62/mo for Linux)
- Need to be at least Standard to get Slots

$ an issue?

- Savings Plan – commit to $ amount
- Save 25% 1 yr, 45% 3 yrs
- Reservation – commit to compute tier
- Save 35% 1 yr, 55% 3 yrs

Live Demo

Bicep

Infrastructure as Code (IAC)

- Source code defining what resources to provision
- Stored in version control
- Declarative – what resources to create, not how to create them
- Deployed via pipeline

Without IAC

- Clickety Clack Configuration™
- Repeat yourself for each environment
- “It worked in Dev/UAT/Staging, not Prod”
- “It works on my machine”

What is Azure Bicep?

- Used to configure Azure resources
- Built and maintained by Microsoft
- Domain-specific language (fancy word for custom)

What is Azure Bicep?

- Provides intellisense, error checking, “whatif,” and orders the resource creations
- Built on top of Azure Resource Manager (ARM) – don’t use ARM directly
- No state file

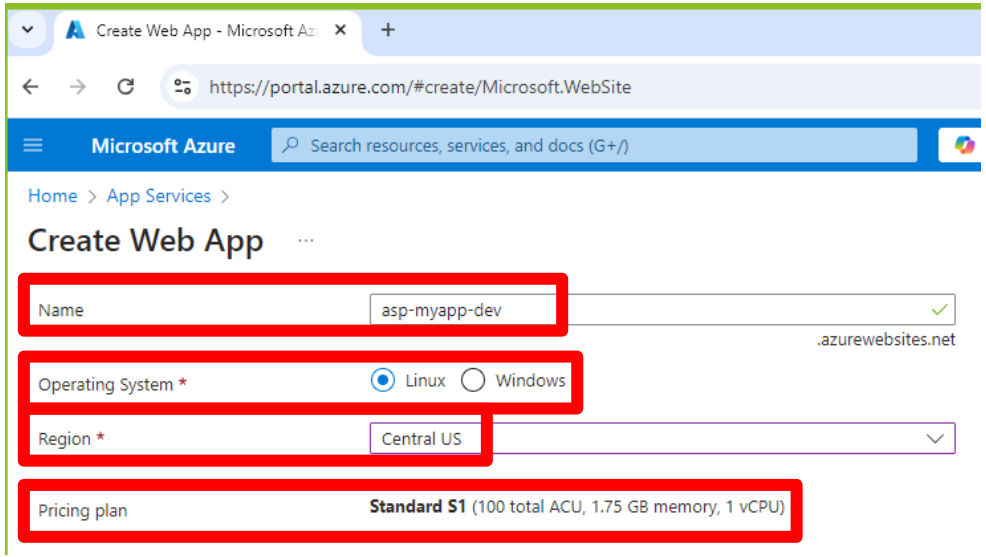
What is Azure Bicep?

appservice.bicep

```
1  resource appServicePlan 'Microsoft.Web/serverfarms@2022-09-01' = {
2    name: 'asp-myapp-dev'
3    location: 'centralus'
4    sku: {
5      name: 'S1'
6    }
7    kind: 'linux'
8  }
```

appservice.bicep

```
1  resource appServicePlan 'Microsoft.Web/serverfarms@2022-09-01' = {
2    name: 'asp-myapp-dev'
3    kind: 'linux'
4    location: 'centralus'
5    sku: {
6      name: 'S1'
7    }
8  }
```



```
resource appServicePlan 'Microsoft.Web/serverfarms@2022-09-01' = {
  name: 'asp-workshop-demo'
  location: 'centralus'
  sku: {
    name: 'S1'
  }
  kind: 'linux'
}
```

```
resource appService 'Microsoft.Web/sites@2022-09-01' = {
  name: 'app-workshop-demo'
  location: 'centralus'
  properties: {
    serverFarmId: appServicePlan.id
    // others
  }
}
```

```
param appName string
@allowed(['dev', 'prod'])
param environment string
param location string

resource appServicePlan 'Microsoft.Web/serverfarms@2022-09-01' = {
  name: 'asp-${appName}-${environment}'
  location: location
  sku: {
    name: 'S1'
  }
  kind: 'linux'
}
```

dev.biccepparam file

```
using '../main.bicep'

param environment = 'dev'
```

But how do I deploy it?

```
az deployment group create
  --name dev-deployment-1
  --template-file infrastructure/main.bicep
  --parameters infrastructure/environments/dev.bicepparam
  --resource-group rg-some-name-here
  --verbose
```

Key Concepts – Quiz time!

- Resources
- Modules
- Parameters
- .bicepparam
- Outputs
- --whatif

Benefits

- No manual work of configuring in the portal (and repeating for each env)
- Eliminate configuration drift
- Traceability of who, did what, and when
- Give Contributor access to the pipeline – not to individuals

Additional Resources

- Documentation for various Bicep resources:
  - https://learn.microsoft.com/en-us/azure/templates/microsoft.web/sites?pivots=deployment-language-bicep

Live Demo

Break then Hands On 40 minutes

LIVE! 360
TECH EVENTS WITH PERSPECTIVE

CI/CD Pipelines

Continuous Integration

- Automated verification of your application that generates artifacts
- Compiles the app
- Runs the tests
- Independent witness - eliminates “works on my machine”

Continuous Delivery

- Takes the artifacts from CI and deploys them automatically
- Doesn’t deploy all the way to Production
- Deploying to Production is a button click

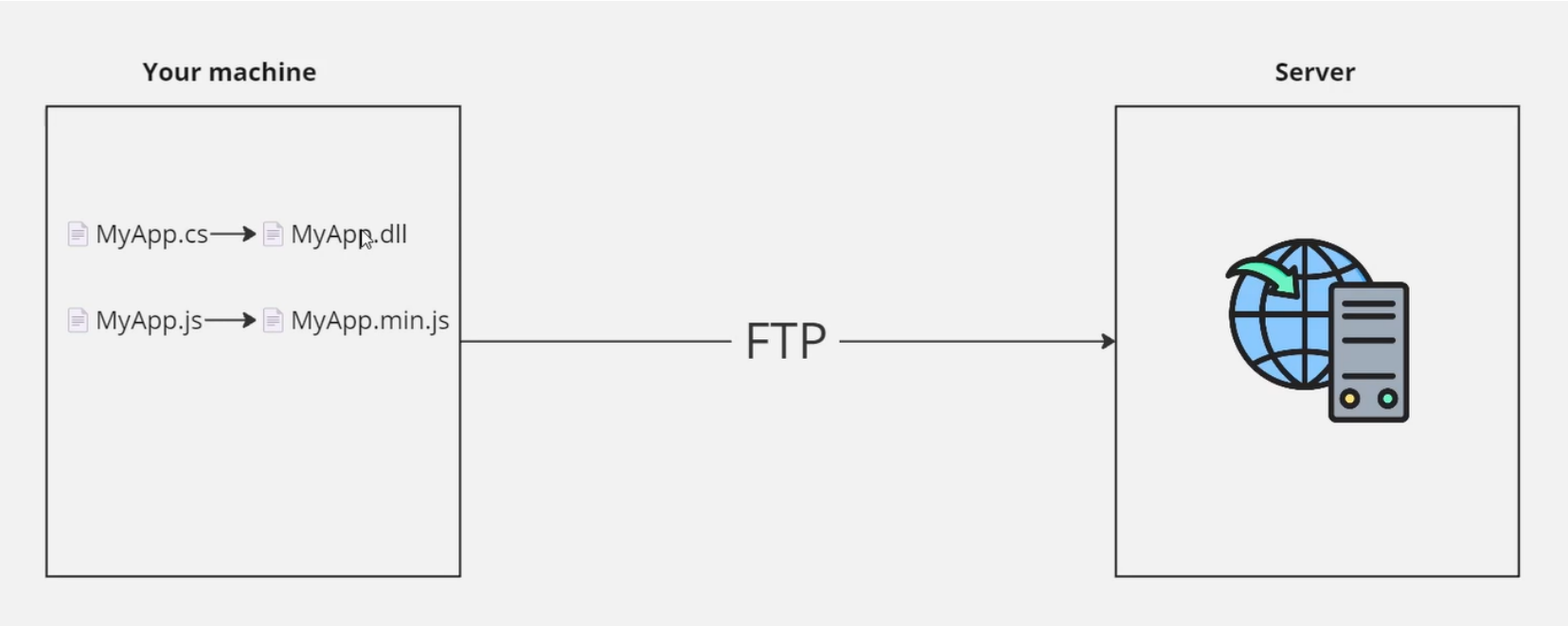
Continuous Deployment

- Deploys all the way to Production automatically
- If the pipeline is green, it’s going to Production

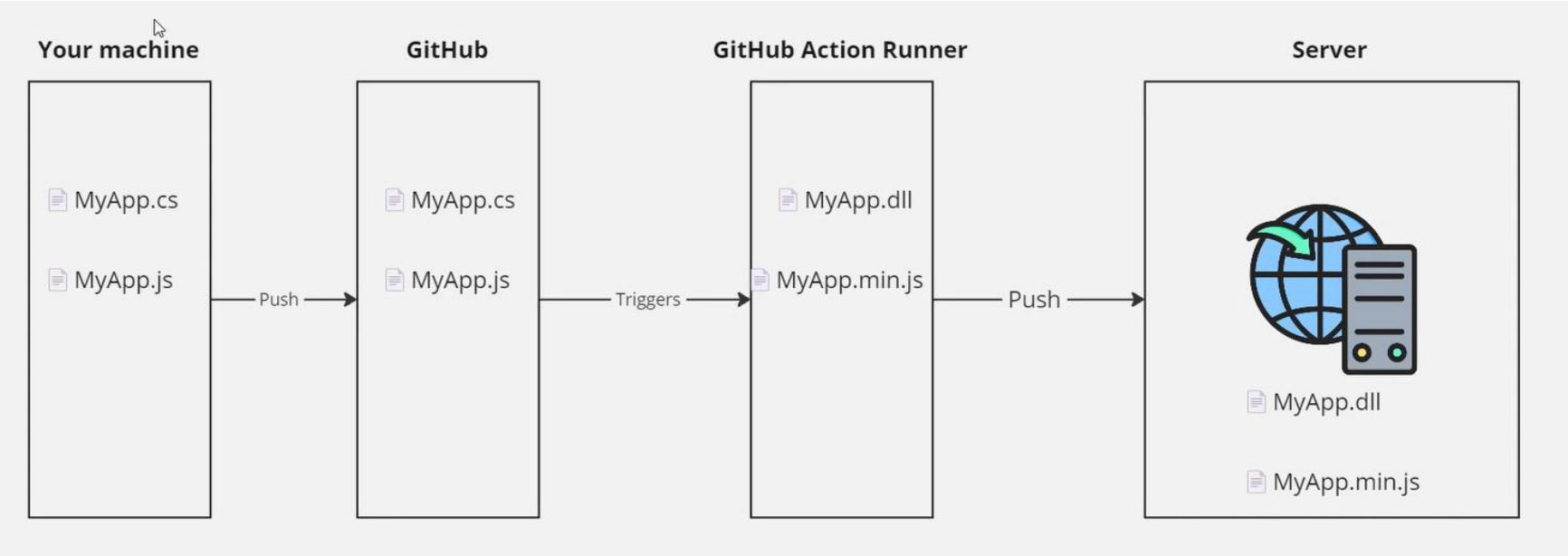
Confident Green

- If our build passes – why aren't we shipping to Production?
- Likely lack of confidence
- Likely missing automated tests or zero downtime deployments, let's fix that
- Ok now why?
- Repeat

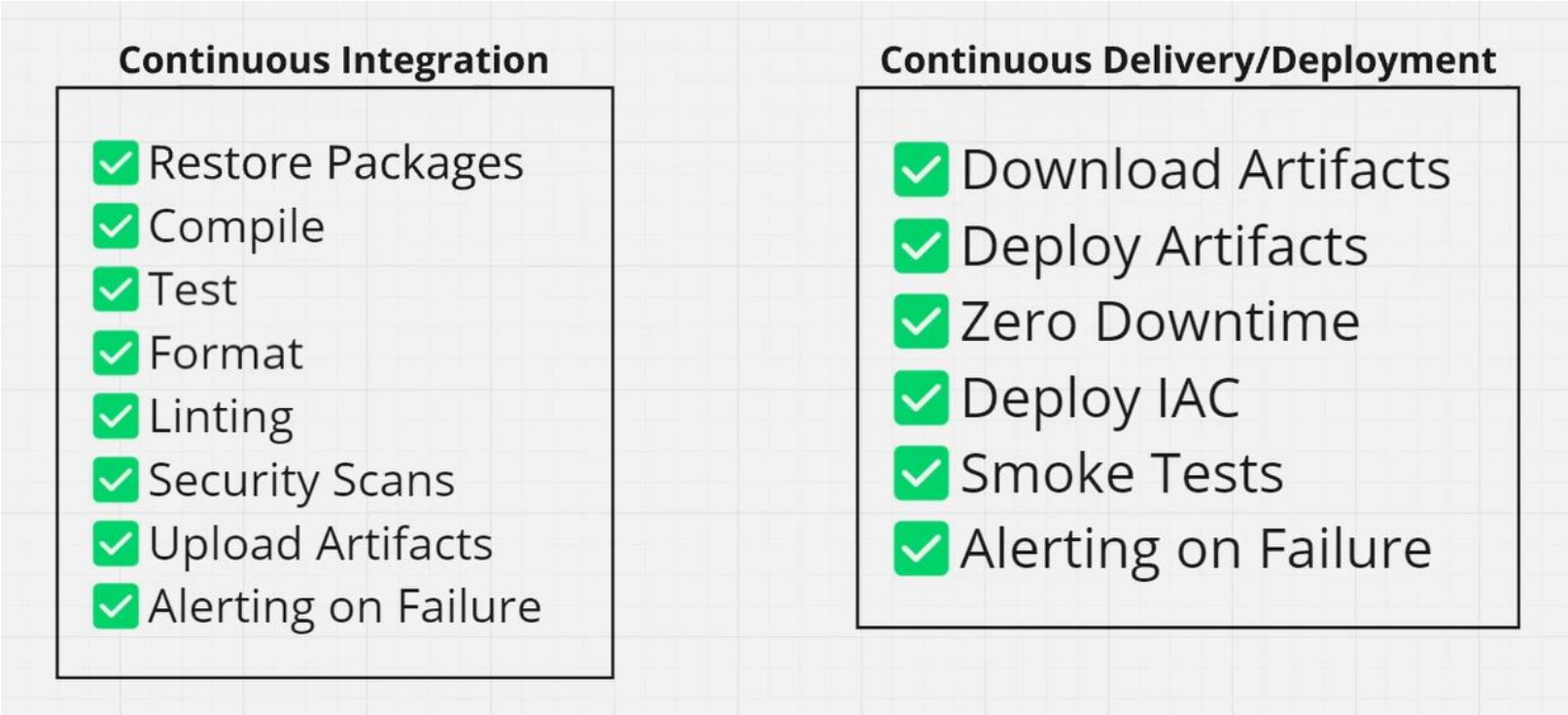
Before CI/CD



After CI/CD



What's in a Pipeline?



GitHub Actions

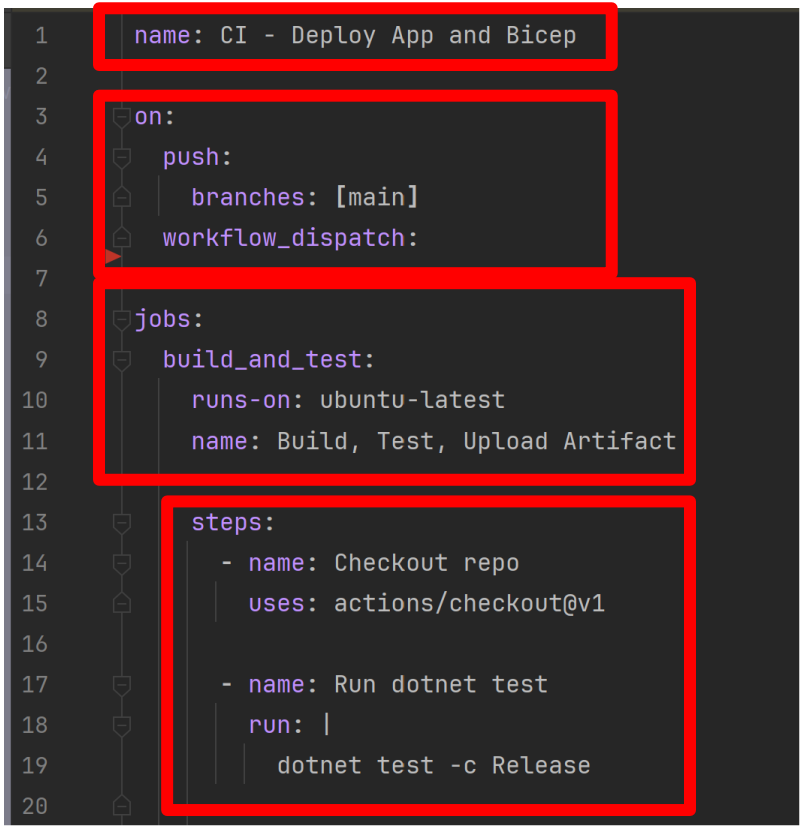
What is GitHub Actions?

- Thing doer on a trigger
- Trigger could be PR, push to main branch, open an issue, etc
- Automatically build and deploys your application
- Including the infrastructure (i.e. Bicep)

GitHub Actions Concepts

- Workflows
- Triggers
- Jobs
- Steps
- Inputs
- Secrets

Example

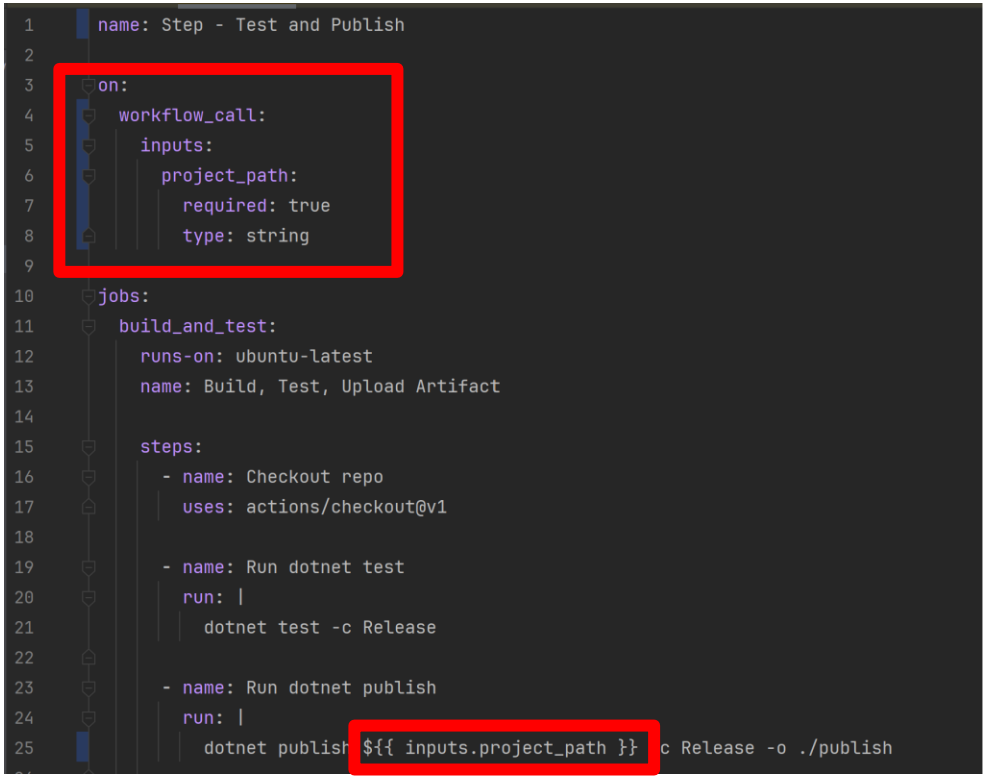


```yaml
name: CI - Deploy App and Bicep

on:
  push:
    branches: [main]
  workflow_dispatch:

jobs:
  build_and_test:
    runs-on: ubuntu-latest
    name: Build, Test, Upload Artifact

    steps:
      - name: Checkout repo
        uses: actions/checkout@v1

      - name: Run dotnet test
        run: |
          dotnet test -c Release
```

How do I reuse workflows?

```yaml
name: CI - Deploy App and Bicep

on:
  push:
    branches: [main]
  workflow_dispatch:

jobs:
  build_and_test:
    runs-on: ubuntu-latest
    name: Build, Test, Upload Artifact

    steps:
      - name: Checkout repo
        uses: actions/checkout@v1

      - name: Run dotnet test
        run: |
          dotnet test -c Release

      - name: Run dotnet publish
        run: |
          dotnet publish ./src/WorkshopDemo/WorkshopDemo.csproj -c Release -o ./publish
```

How do I reuse workflows?



```yaml
name: Step - Test and Publish

on:
  workflow_call:
    inputs:
      project_path:
        required: true
        type: string

jobs:
  build_and_test:
    runs-on: ubuntu-latest
    name: Build, Test, Upload Artifact

    steps:
      - name: Checkout repo
        uses: actions/checkout@v1

      - name: Run dotnet test
        run: |
          dotnet test -c Release

      - name: Run dotnet publish
        run: |
          dotnet publish ${{ inputs.project_path }} -c Release -o ./publish
```

Consume reusable workflow

```yaml
name: CI - Test and Publish

on:
  push:
    branches: [main]
  workflow_dispatch:

jobs:
  build_and_test:
    uses: ./.github/workflows/step-build-and-test.yml
    with:
      project_path: ./src/WorkshopDemo/WorkshopDemo.csproj
```

Consume from another repo

```yaml
name: CI - Test and Publish

on:
  push:
    branches: [main]
  workflow_dispatch:

jobs:
  build_and_test:
    uses: my-org-or-username/repo-name/step-build-and-test.yml
    with:
      project_path: ./src/WorkshopDemo/WorkshopDemo.csproj

```

Live Demo

Break then Hands On 40 minutes

Health Checks

What are Health Checks?

- Health Checks check an app's status
- Might stop a rolling deployment
- Might restart the app on failure
- App Services allow configuring Health Check endpoint
- By default it checks root URL

What are Health Checks?

- /api/healthz
- Why z?
- Z-pages from Google

C#

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddHealthChecks();

var app = builder.Build();

app.MapHealthChecks("/healthz");

app.Run();
```

Hands On
15 Minutes

Azure Key Vault

What is Azure Key Vault

- Secret Store for Azure
- Don’t store secrets in Version Control
- Traceability
- Rotate Secrets
- $0.03 per 10K requests

Best Practices

- Separate Key Vault per app per env
- 1 app * 3 envs = 3 key vaults
- 2 apps * 3 envs = 6 key vaults
- Don’t leak keys across envs or apps

.NET Integration

- Plugs into IConfiguration
- Loads keys on app boot (saves $ and more performant)
- Azure.Security.KeyVault.Secrets

Managed Identities

- Essentially the user (Service Principal) a service (ie App Service) runs as
- Allows you to say “this App Service can talk to this Key Vault/DB/etc”
- Microsoft handles the credentials for you behind the scenes

RBAC for Key Vault

- Role Based Access Control allow syou to specify who can connect to Key Vault
- Could be a group (ie Developers), could be an application
- Applications usually just need Read not Write

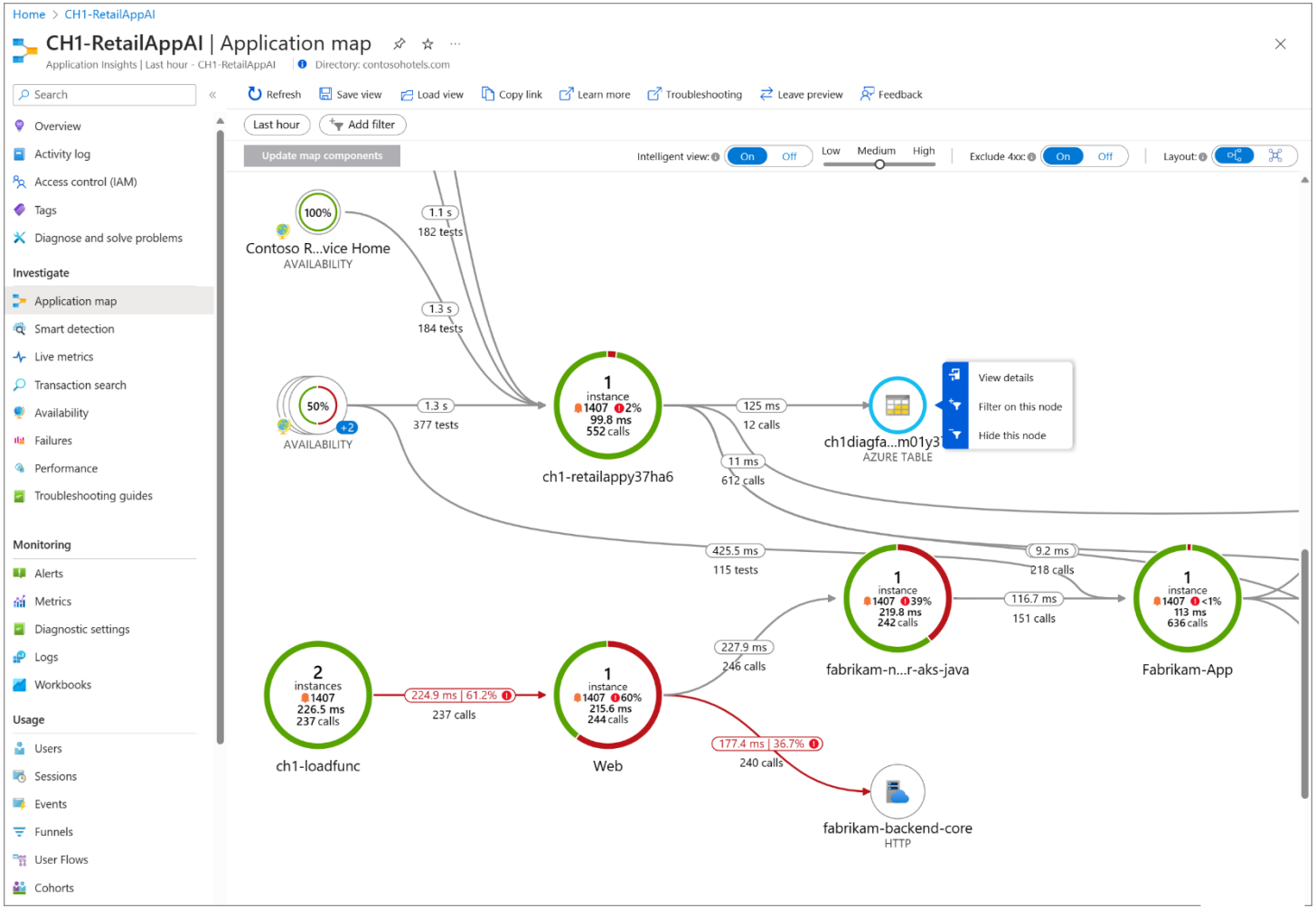
Live Demo

Hands On
30 Minutes

Azure App Insights

What is Application Insights?

- Observability Platform for Azure
- Logs
- Metrics
- Traces
- Application Maps
- Diagnose Performance Issues



Search results | Learn more | Feedback

End-to-end transaction

Operation ID: b8e278fcfd20456a88ab48ac9cde6476

= Request (incoming) = Dependency (outgoing) = Exception = Profiler trace = Debug snapshot

EVENT	RES.	DURATION
App Service Customer Details Central US		557 ms
http://fabrikamfiberapp.azurewebsites.net/Customers/Details/8469		557 ms
fabrikamfiberapp GET Customers/Details	500	430.3 ms
fabrikamxyz FabrikamISQL		57 ms
fabrikamaccount POST fabrikamaccount/Tables	409	112 ms
fabrikamaccount POST fabrikamaccount/fabrikamfiber	409	111 ms
fabrikamaccount GET fabrikamaccount/fabrikamfiber	200	48 ms
EXCEPTION System.FormatException		
Failed Central US		0

0MS 200 MS300 MS400 MS500 MS

10 All 1 Traces 0 Events

View all telemetry

Create work item | Open profiler traces

fabrikamfiberapp

GET Customers/Details

Request Properties

Show all

Event time	8/23/2018, 6:14:49 AM
Request name	GET Customers/Details
Response code	500
Successful request	False
Response time	430.3 ms
Request URL	http://fabrikamfiberapp.azurewebsites.net/Customers/Details/8469

Call Stack

Show Just My Code

```
System.FormatException:
   at System.Number.StringToNumber (mscorlib, Version=4.0.0.0, Cult
   at System.Number.ParseInt32 (mscorlib, Version=4.0.0.0, Culture=
   at System.Int32.Parse (mscorlib, Version=4.0.0.0, Culture=neutra
   at FabrikamFiber.DAL.Data.AddressValidator.ValidZipCode (Fabrika
   at FabrikamFiber.DAL.Models.Address.FullAddress (FabrikamFiber.D
   at ASP._Page_Views_Customers_Details_cshtml.Execute (App_Web_sw4
   at System.Web.WebPages.WebPageBase.ExecutePageHierarchy (System.
   at System.Web.Mvc.WebViewPage.ExecutePageHierarchy (System.Web.M
   at System.Web.WebPages.StartPage.RunPage (System.Web.WebPages, V
   at System.Web.WebPages.StartPage.ExecutePageHierarchy (System.We
   at System.Web.WebPages.WebPageBase.ExecutePageHierarchy (System.
   at System.Web.Mvc.RazorView.RenderView (System.Web.Mvc, Version=
   at System.Web.Mvc.BuildManagerCompiledView.Render (System.Web.Mv
   at System.Web.Mvc.ViewResultBase.ExecuteResult (System.Web.Mvc,
```

Live Demo

Hands On
25 Minutes

Takeaways

- How to leverage Azure
- Why IAC is useful and how Bicep workws
- How GitHub Actions fits into the big picture
- Some takeaway tips and tricks, even if you had prior experience with some of this

Resources

- This slide deck
- https://github.com/scottsauber/workshop-dotnet-azure-github-bicep
  - “final” branch has the final state of things

Questions?

Session Survey

- Your feedback is very important to us
- Please take a moment to complete the session survey found in the mobile app
- Use the QR code or search for “Converge360 Events” in your app store
- Find this session on the Agenda tab
- Click “Session Evaluation”
- Thank you!

LIVE! 360
TECH EVENTS WITH PERSPECTIVE