🧪 Lab: Deployment Strategies with .NET + Azure App Service

🔷 Prerequisites

- Azure Subscription
- .NET 9+ SDK installed
- GitHub account with a repo for your .NET app
- Azure CLI installed

Step 1: Create and Push a .NET App

```
dotnet new webapp -o DeploymentLab
cd DeploymentLab
git init
git add .
git commit -m "Initial commit"
git remote add origin https://github.com/<your-username>/<repo>.git
git push -u origin main
```

Step 2: Create an Azure App Service with Deployment Slots

1. In **Azure Portal**, search for **App Services → Create App Service**.
    - Runtime stack: `.NET 6 (LTS)`
    - OS: Windows or Linux
    - Region: closest to you
2. Once created, go to **Deployment Slots → Add Slot**.
    - Name it `staging`.
    - Copy settings from production.

✅ You now have:

- `production` slot → Blue
- `staging` slot → Green

Step 3: Configure GitHub Actions for Deployment

1. In Azure Portal → App Service → **Deployment Center**.

2. Select **GitHub Actions (CI/CD)**.
3. Authorize GitHub → select repo & branch.
4. Azure will create a `.github/workflows/azure-webapps.yml`.

Example workflow (simplified):

```
name: Build and Deploy

on:
  push:
    branches:
      - main

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
    - uses: actions/checkout@v3
    - name: Setup .NET
      uses: actions/setup-dotnet@v3
      with:
        dotnet-version: '9.0.x'
    - run: dotnet build --configuration Release
    - run: dotnet publish -c Release -o publish_output

    - name: Deploy to Azure WebApp
      uses: azure/webapps-deploy@v2
      with:
        app-name: '<your-app-service-name>'
        slot-name: 'staging'   # 👈 Deploy to staging slot
        publish-profile: ${{ secrets.AZURE_WEBAPP_PUBLISH_PROFILE }}
        package: publish_output
```

Step 4: Practice Blue-Green Deployment

1. Push a change to GitHub → GitHub Actions deploys it to **staging (Green)**.
2. Test your app at `https://<appname>-staging.azurewebsites.net`.
3. Once validated → Azure Portal → **Deployment Slots** → **Swap** → Swap `staging` with `production`.
    - Green becomes live, Blue becomes idle.
4. Rollback? Just swap again.

✅ You've implemented **Blue-Green Deployment**.

Step 5: Practice Canary Deployment

1. In **Azure Traffic Manager** or **Front Door**:
    - Create a new profile.
    - Add two endpoints:
        - Production slot (weight 80)

        - Staging slot (weight 20)
2. Users are now split → 80% see Blue, 20% see Green.
3. Monitor performance via **Azure Application Insights**.
4. If stable → gradually increase Green's weight to 100.

✅ You've implemented **Canary Deployment**.

Step 6: Monitor and Validate

- Enable **Application Insights** on both slots.
- Track metrics (errors, response times, failures).
- Use GitHub Actions logs for deployment validation.

🔷 What You Learned

- **Blue-Green Deployment** using Azure Deployment Slots + Swap.
- **Canary Deployment** using Azure Traffic Manager weighted routing.
- Hands-on GitHub Actions CI/CD pipeline for .NET apps.