SQLSATURDAY
COSTA RICA | 10 AUG 2024

Deploying Enterprise Workloads in Azure with Bicep Language

Agradecimiento a los patrocinadores

Gold

Microsoft

ULATINA Powered by Arizona State University

solcomp Soluciones Computarizadas

SQLSATURDAY
COSTA RICA | 10 AUG 2024

Learning objectives

After completing this session, you will be able to:

1. Work with Bicep Language to deploy your solutions in Azure.
2. Understand how Bicep Modules work.
3. Deploy a multi-tier application with a front-end, back-end, and database.

Hello 👋 !

Thank you for joining me today

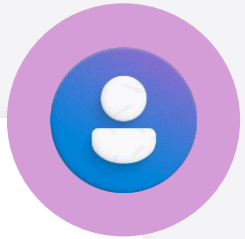
Dave Rendon

Azure MVP, Microsoft Certified Trainer

twitter.com/daverndn
linkedin.com/in/daverndn
Blog.azinsider.net

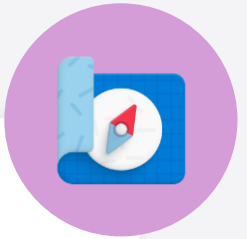
Let’s have a great time together!

We all contribute to a great session



- Be present
- Be open to questions and different perspectives

Resources



- GitHub Repo
https://github.azinsider.net

- Architecture Reference
https://bit.ly/ase-bicep

- Azure Verified Modules
https://azure.github.io/Azure-Verified-Modules

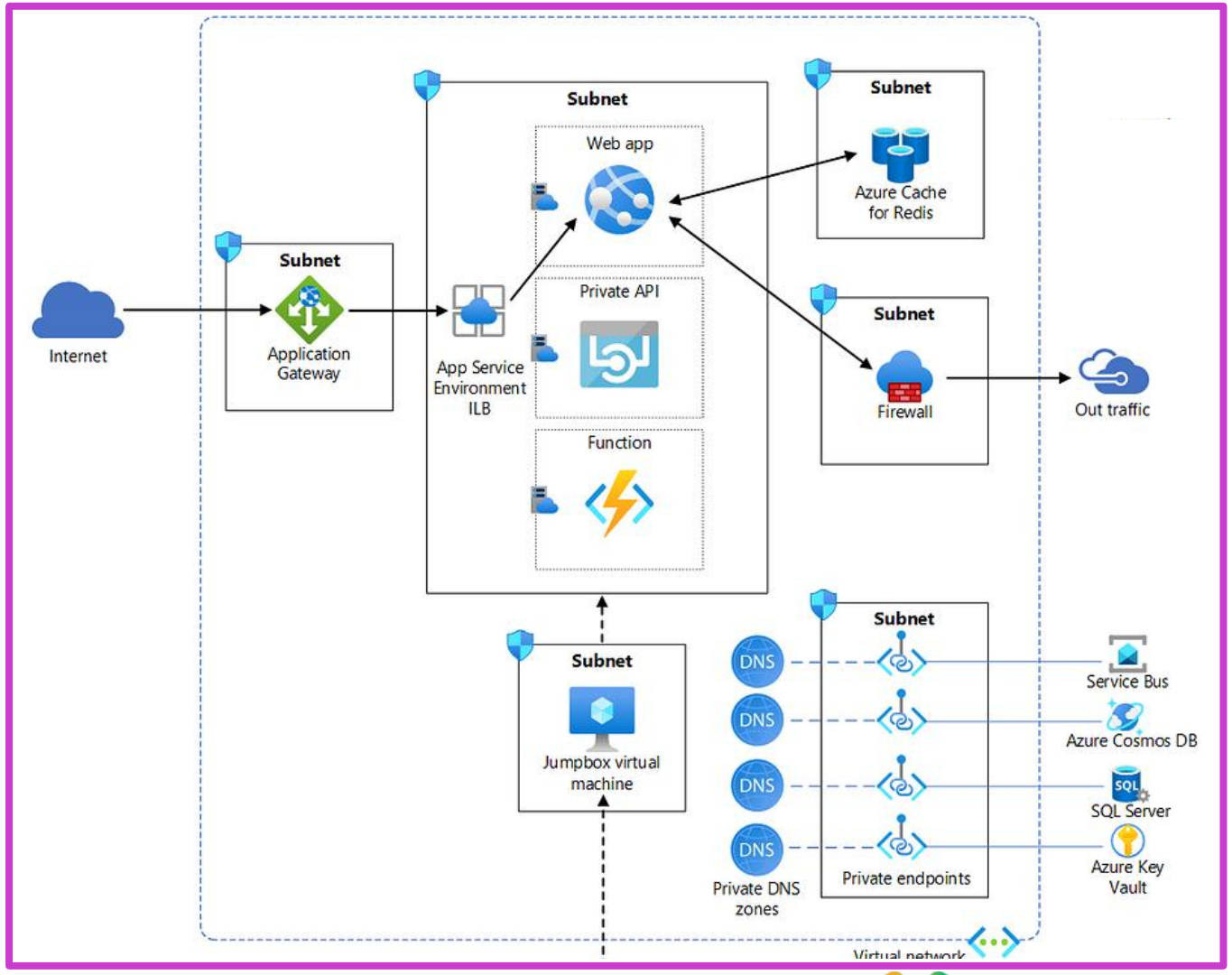
Deploying Enterprise Workloads

Requirements:

- Scalability
- High availability
- Security

Example:

- Deploy a multi-tier application with a front-end, back-end, and database.
- Use Bicep to define and deploy resources like VMs, Load Balancers, and Databases.



Consider Azure Bicep Language

Simpler syntax for writing templates

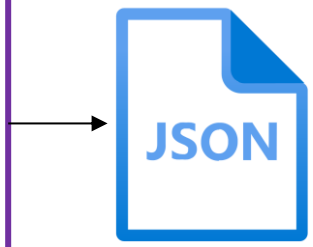
Smaller module files you can reference from a main template

Automatically detect dependencies between your resources

Visual Studio Code extension with validation and IntelliSense

Bicep file

```
resource storageAccount
'Microsoft.Storage/storageAccounts@2021-01-01' =
{
  name: storageAccountName
  location: location
  tags: {
    displayName: storageAccountName
  }
  kind: 'StorageV2'
  sku: {
    name: 'Standard_LRS'
  } }
```



Why use Bicep Language?

To be able to use a concise syntax	Simplicity
To create reusable modules	Reusability
To leverage built-in error checking	Error Handling
To be able to integrate VS code support	Administration

Simplicity

- Bicep provides a more readable and concise syntax compared to traditional ARM templates.

Reusability

- Break down complex deployments into smaller, manageable, and reusable components.

Error Handling

- Bicep includes built-in type safety, which helps catch errors during development. It provides clear and concise error messages.

Administration

- Bicep integrates seamlessly with Visual Studio Code, offering features like IntelliSense, syntax highlighting, and code snippets.

Getting Started with Bicep Language

- Installation:
  - Install Bicep CLI.
  - Ensure Azure CLI is installed and updated.

- Bicep Structure:

```
resource myStorageAccount 'Microsoft.Storage/storageAccounts@2021-02-01' = {
   name: 'mystorageaccount'
    location: 'West US'
    sku: {
      name: 'Standard_LRS'
    }
    kind: 'StorageV2'
  }
```

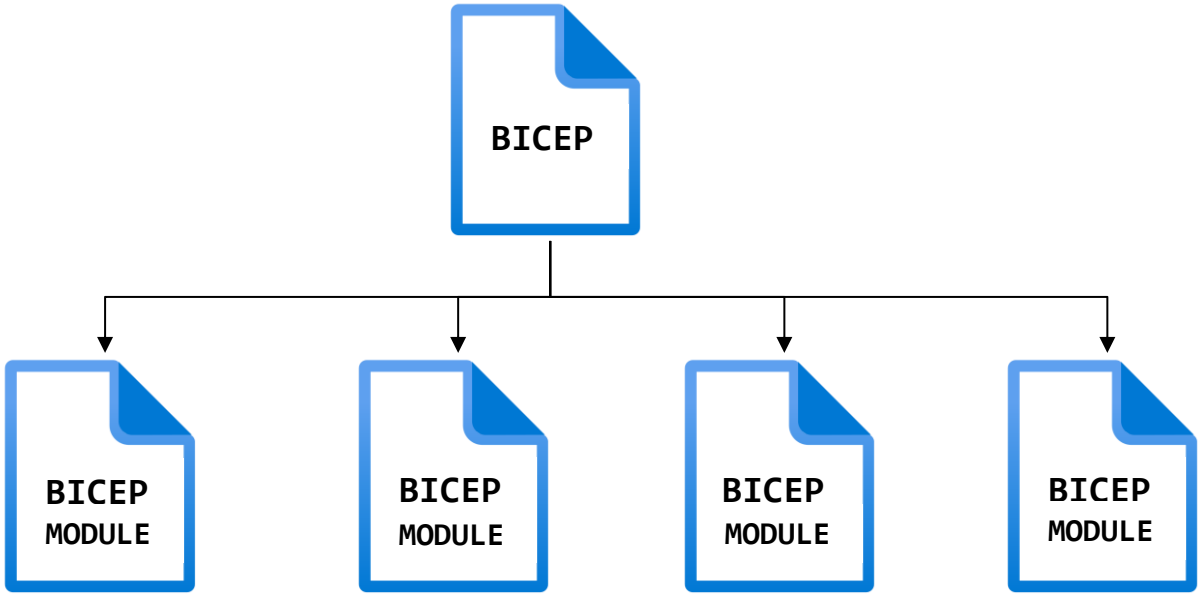
- Tooling:
  - Visual Studio Code extension for Bicep.
  - Bicep Playground for online experimentation.

Bicep Modules

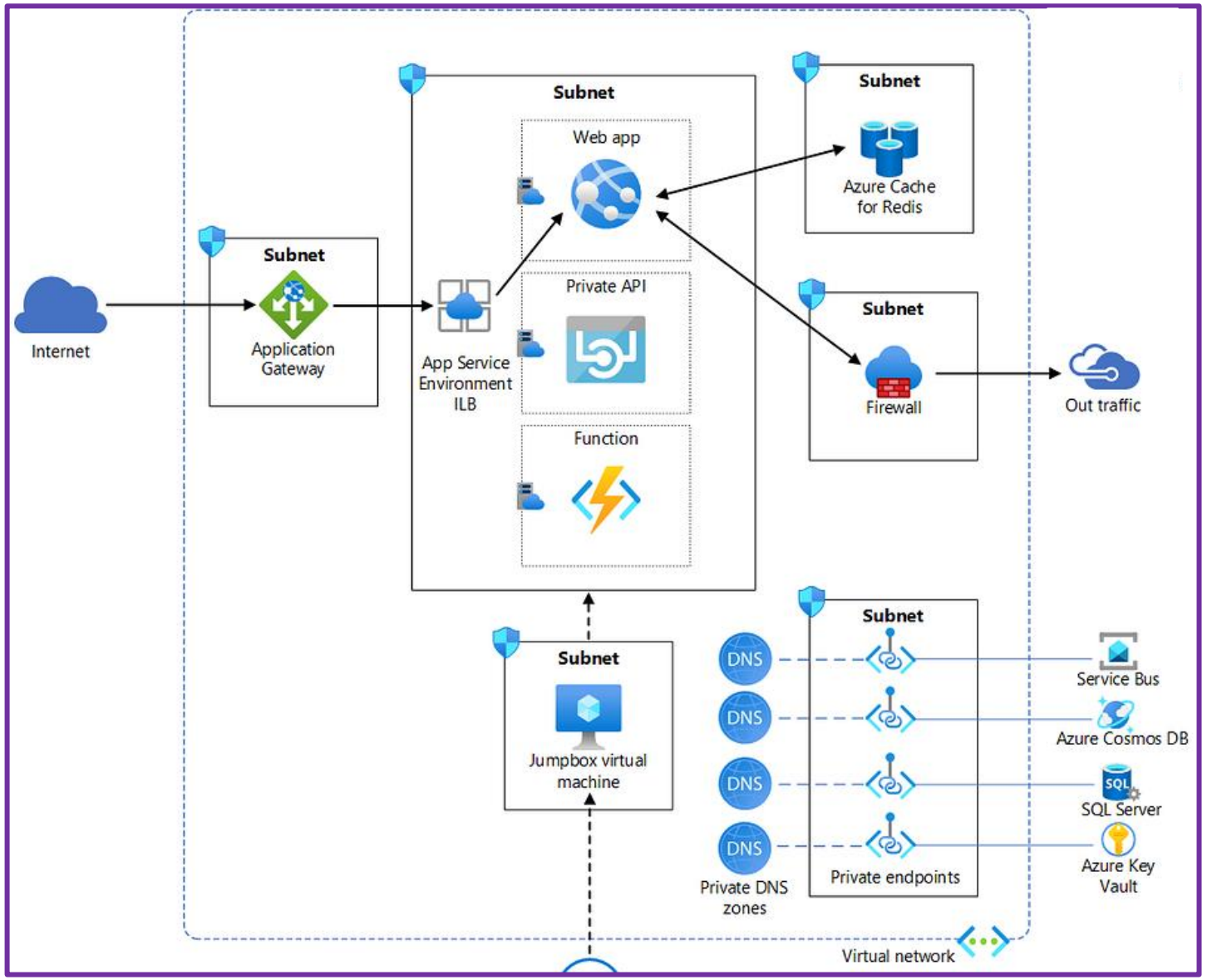
Break down complex deployments into smaller, reusable modules.

```
module <symbolic-name> '<path-to-file>' = {
  name: '<linked-deployment-name>'
  params: {
     <parameter-names-and-values>
      }
 }
```

```
module stgModule '../storageAccount.bicep' = {
  name: 'storageDeploy'
  params: {
    storagePrefix: 'examplestg1'
  }
}
```



Deploying Enterprise Workloads using Bicep Language



main.bicep

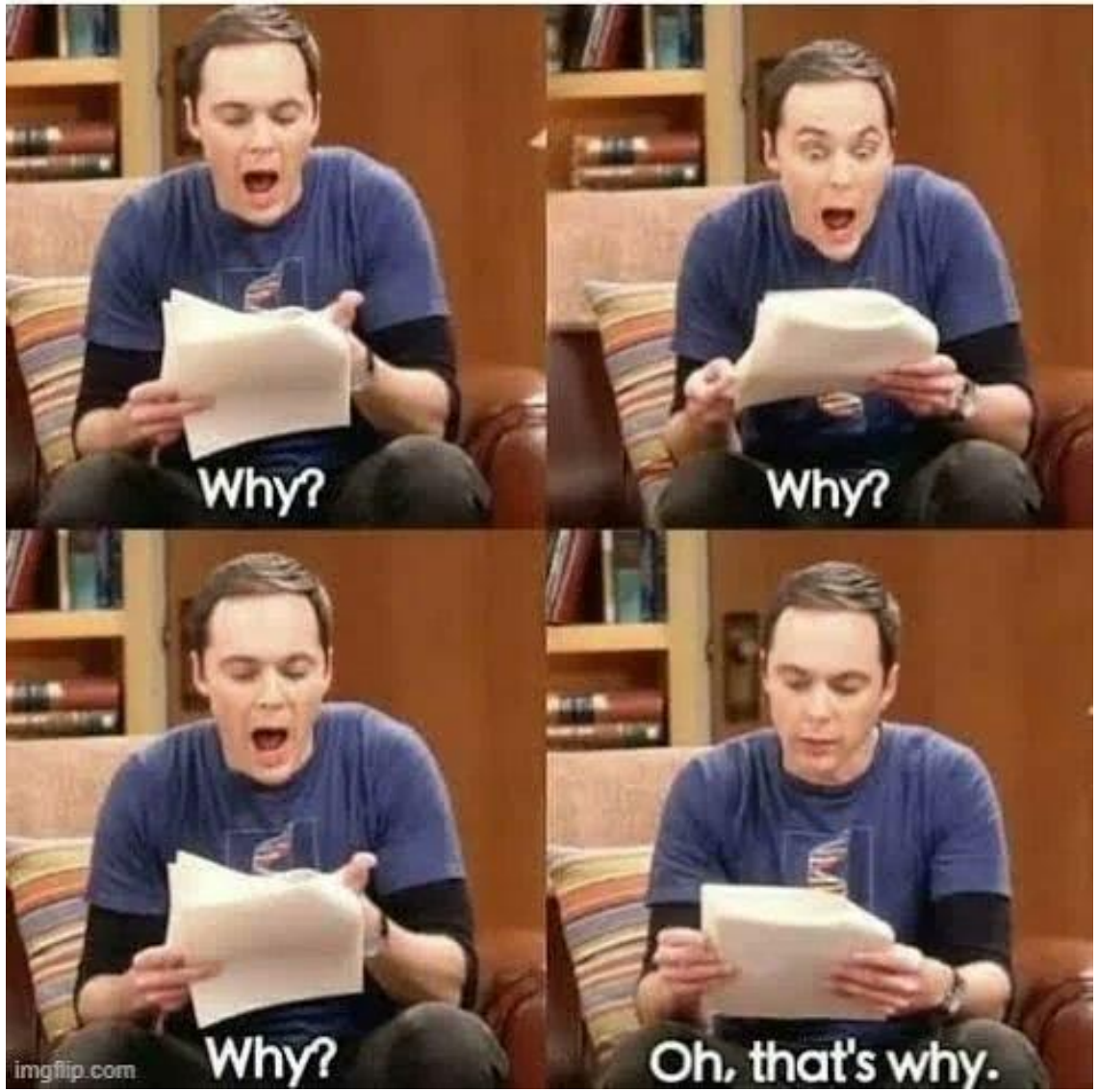
main.bicepparam

modules

```
VERBOSE: Authentgit clone https://github.com/daveRendon/azinsider.git
Cloning into 'azinsider'...e drive ...
remote: Enumerating objects: 3497, done.
remote: Counting objects: 100% (1735/1735), done.
remote: Compressing objects: 100% (1057/1057), done.
remote: Total 3497 (delta 636), reused 1518 (delta 603), pack-reused 1762
Receiving objects: 100% (3497/3497), 21.64 MiB | 25.10 MiB/s, done.
Resolving deltas: 100% (1161/1161), done.
```

```
PS /home/david> cd azinsider/application-workloads/enterprise-deployment-using-azure-app-service-environment
```

WHEN YOU LOOK AT CODE YOU WROTE LAST YEAR



```
  + Microsoft.Storage/storageAccounts/resourcesfbhikmhsrcdfo/blobServices/default/containers/rscontai

      apiVersion:             "2021-09-01"
      id:
"/subscriptions/d988cbee-043f-4c46-9a59-dedb2119e48c/resourceGroups/azinsider_demo_eastus2/providers/
ices/default/containers/rscontainer"
      name:                   "rscontainer"
      properties.publicAccess: "Blob"
      type:                   "Microsoft.Storage/storageAccounts/blobServices/containers"

  + Microsoft.Web/hostingEnvironments/ase-azinsider [2022-09-01]

      apiVersion:                         "2022-09-01"
      id:
"/subscriptions/d988cbee-043f-4c46-9a59-dedb2119e48c/resourceGroups/azinsider_demo_eastus2/providers/
      kind:                               "ASEV3"
      location:                           "eastus"
      name:                               "ase-azinsider"
      properties.dedicatedHostCount:      0
      properties.internalLoadBalancingMode: "Web, Publishing"
      properties.virtualNetwork.id:
"/subscriptions/d988cbee-043f-4c46-9a59-dedb2119e48c/resourceGroups/azinsider_demo_eastus2/providers/
et"
      properties.zoneRedundant:           false
      type:                               "Microsoft.Web/hostingEnvironments"

Resource changes: 33 to create.

Are you sure you want to execute the deployment?
```

Deployment is in progress

Deployment name : AzInsiderDeployment-08-10-2024-496
Subscription : SpringToys
Resource group : azinsider_demo_eastus2

Start time : 8/10/2024, 1:38:30 PM
Correlation ID : 602809e0-6070-44a8-8cec-3546998e4f40

Deployment details

Resource	Type	Status	Operation details
ase	Deployment	Created	Operation details
jumpbox	Deployment	OK	Operation details
jumpbox	Deployment	OK	Operation details
ase-vnet-route	Route table	OK	Operation details
vnet-azinsider	Virtual network	OK	Operation details

Best Practices

- Consistent Naming Conventions: Follow naming standards for resources.
- Parameterization: Use parameters to make templates flexible and reusable.
- Modularity: Create reusable modules for common patterns.
- Documentation: Comment your code to improve readability and maintainability.
- Version Control: Keep Bicep files under source control to manage changes.

Conclusion

SQLSATURDAY
COSTA RICA | 10 AUG 2024

Bicep Language

- Infrastructure as Code for Azure
- Ideal for large environments
- Azure Verified Modules

Why Bicep

- Simplicity
- Reusability
- Error Handling
- Administration

Best Practices

- Modularity
- Version Control
- Consistent Naming Convention

Other concepts

- Conditional Deployments
- Functions

References

Simplifying Enterprise Deployments for Azure App Service Environments with Bicep Language

https://bit.ly/ase-bicep

Azure Verified Modules

https://azure.github.io/Azure-Verified-Modules/

Bicep Samples

http://github.azinsider.net/

Thank You 👋 !

Dave Rendon

Azure MVP, Microsoft Certified Trainer

twitter.com/daverndn
linkedin.com/in/daverndn
Blog.azinsider.net

GRACIAS

Gold