Introduction

Primer to DevOps - Project lifecycle
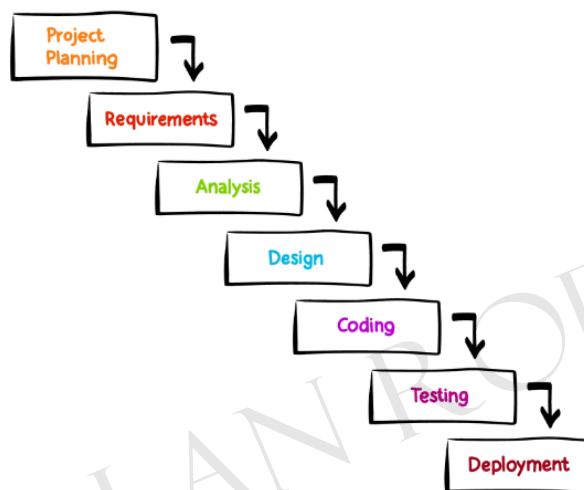
Primer to DevOps - Agile Methodology

Agile software development

In 2001, 17 software developers met to discuss lighter development methods and came up with the manifesto for Agile Software Development

In software development, agile practices include requirements discovery and solutions improvement through the collaborative effort of self-organizing and cross-functional teams with their customer(s)/end user(s), adaptive planning, evolutionary development, early delivery, continual improvement, and flexible responses to changes in requirements, capacity, and understanding of the problems to be solved
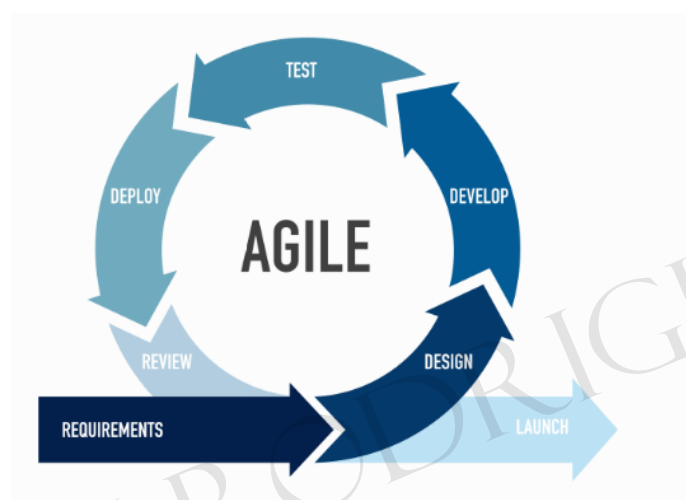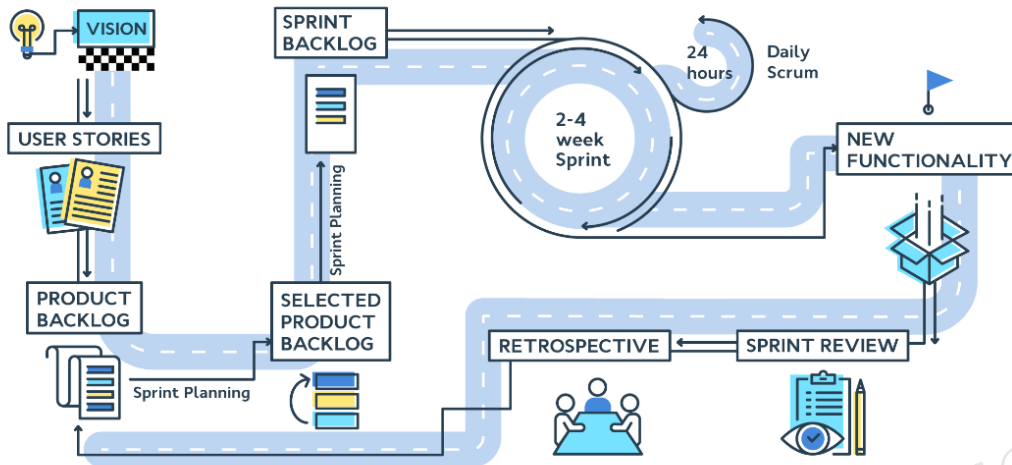

Primer to DevOps - Scrum process

This is an agile development methodolgy

SCRUM PROCESS

User Story - This is a feature that the user wants. This is written from the users perspective.

Product Backlog - This is the list of items that need to be done. The list is based off the requirements, features , enhancements. This is normally managed by the product owner or the product manager.

Items are then taken from the Product Backlog and then assigned to a sprint. The sprint is normally a duration of 2 weeks. The items assigned to the sprint are made part of the sprint backlog.

Sprint Review - Here the team showcases what was done in the sprint. Here the decision can be made as to whether the deliverable should be promoted to production.

Sprint Restropective - The team comes together to see what worked and what did not work.

Primer to DevOps - Development and Operations

Development team is developing an
application

app

Testing team tests the application

Application is deployed to a production
environment

And then the operations team takes over

Server operations - Take cares of the
servers

Database operators - Takes care of the
databases

New feature

app

Sometimes developers try to apply quick
fixes if they are under a time constraint.

Maybe the testing times for the applications
are cut short

Operations teams are hesistant to deploy
the changes.

They are in the direct line of fire. If the
changes goes wrong, then it could lead to a
downtime for the application.

There is always this constant tussle
between development and operations.

The entire purpose of DevOps is to ensure that developers and IT operations work in collaboration.

Meetings that are held with stakeholders normally have the Developers and Operations teams.

Automation is a key aspect.

Faster delivery of application features.

Adoption of the right tools.

Big shift in the mindset and processes.
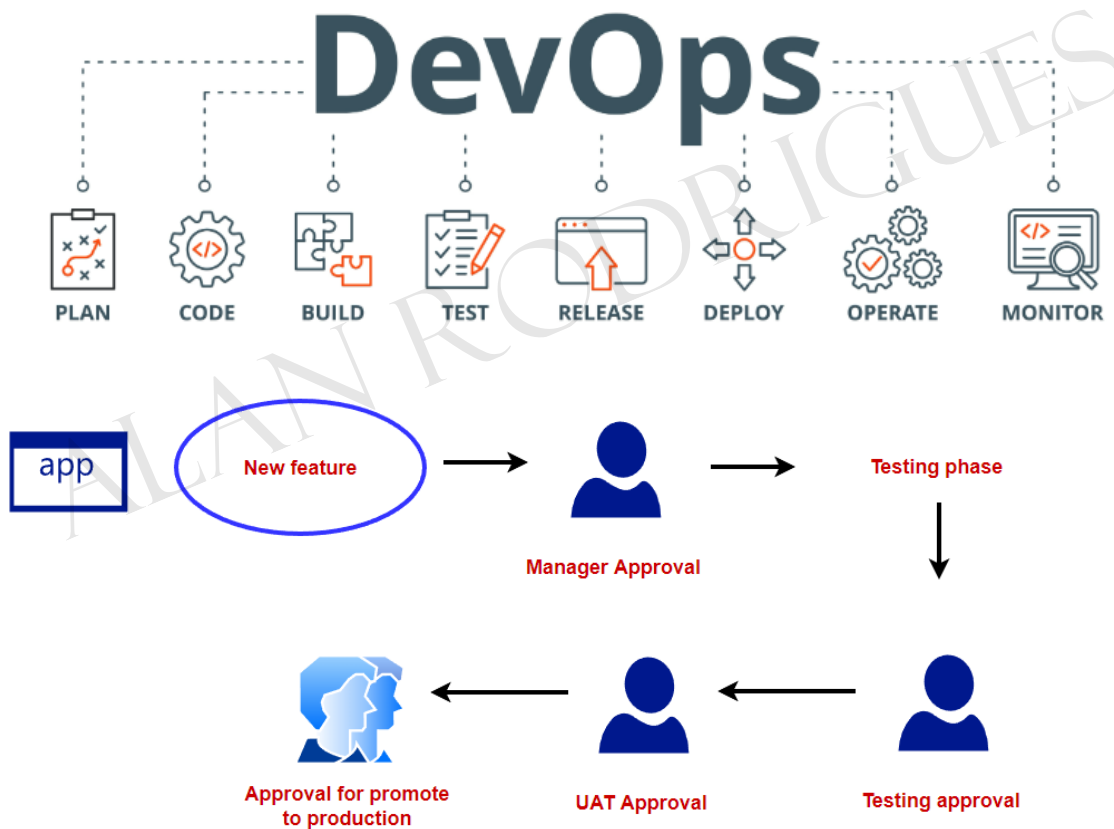
Configure processes and communications

Let's start with planning

Your working for a company - Cloudportalhub

This company acts as the IT department for a larger organization

The organization has many more departments such as Logistics, Human Resource etc.

LEADERSHIP

We will get right on it.
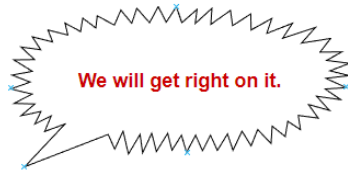
IT department

LEADERSHIP

Human Resource Business
department

Focus on Application
Delivery

Focus on adding
features

Focus on collaboration
with end users
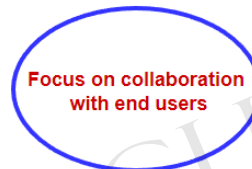
Traditional Approach

Approach to the project

Traditional Approach

LEADERSHIP

IT department

TEAM

Project team

Assign a Project Manager

Get the budget approved.

Hire the people, get the team
in place.

Project Manager would try to setup a plan.

Business Analysts would setup meetings with the end users to gather requirements.

Could take around 2 weeks to a month.

Design the screens

Design the data store

Design the application modules

Could take a month

Developers start with the coding phase

Testing the application

Could take months

Software in place

Our Approach

Scrum process

Scrum team

Small group of people focused on achieving an objective

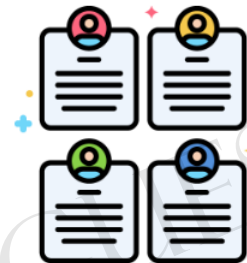**Gather requirements from the users**

User stories

Product Backlog

Sprint Backlog

Check the status of your sprints

We need a software to track all of this

Getting started with Azure Boards
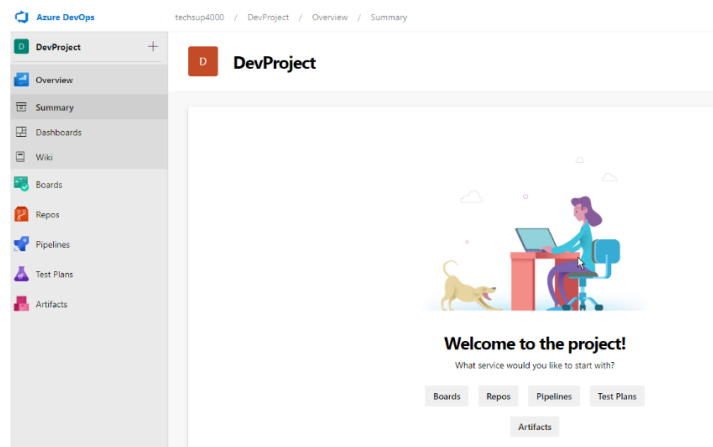
Each organization gets its own *free tier* of services (up to five users for each service type) as follows. You can use all the services, or choose only what you need to complement your existing workflows.

- Azure Pipelines ⧉ : One hosted job with 1,800 minutes per month for CI/CD and one self-hosted job
- Azure Boards ⧉ : Work item tracking and Kanban boards
- Azure Repos ⧉ : Unlimited private Git repos
- Azure Artifacts ⧉ : Package management
- Unlimited Stakeholders
  - Five Azure DevOps users (Basic)
  - Free tier of Microsoft-hosted CI/CD (one concurrent job, up to 30 hours per month)
  - 2 GiB of Azure Artifacts storage
  - One self-hosted CI/CD concurrent job

https://docs.microsoft.com/en-us/azure/devops/user-guide/plan-your-azure-devops-org-structure?view=azure-devops

Creating user stories

Azure Boards

Kanban

This is a framework available that can be used to implement agile and DevOps

Use of Kanban boards

Understanding permissions
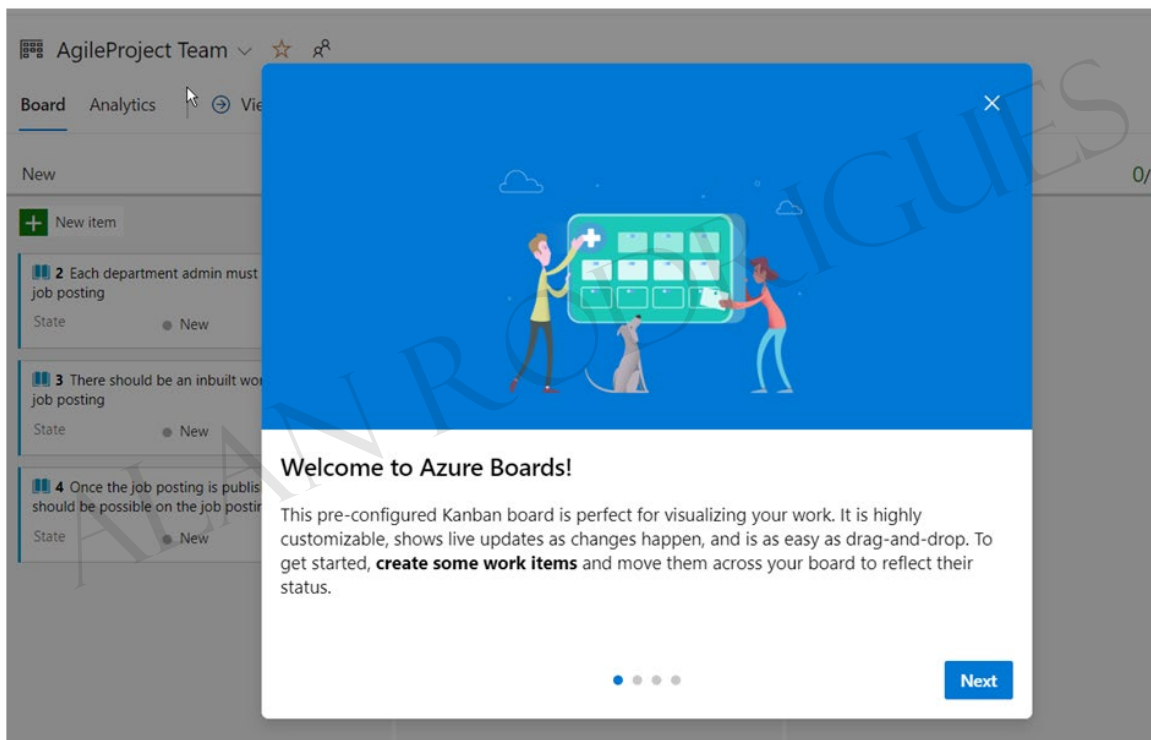
Permissions

When a user is added to Azure DevOps, they are added to a default security group

Permissions can be assigned at an organization, project or object level.

Default security groups

Project	Organization or Collection
- Build Administrators	- Project Collection Administrators
- Contributors	- Project Collection Build Administrators
- Project Administrators	- Project Collection Build Service Accounts
- Project Valid Users	- Project Collection Proxy Service Accounts
- Readers	- Project Collection Service Accounts
- Release Administrators	- Project Collection Test Service Accounts
- *TeamName* Team	- Project Collection Valid Users
	- Project-Scoped Users
	- Security Service Group

https://docs.microsoft.com/en-us/azure/devops/organizations/security/about-permissions?view=azure-devops&tabs=preview-page

Azure Boards - Integration with Microsoft Teams
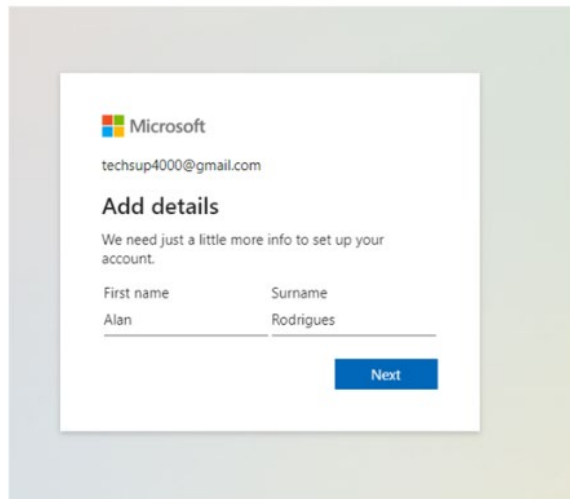
Microsoft Teams

Microsoft

How do you want to use Teams?

○ **For school**
To connect students and faculty for courses and projects, in a classroom or online

○ **For friends and family**
For everyday life, to make audio or video calls

● **For work and organizations**
To work with teammates wherever they are

Next
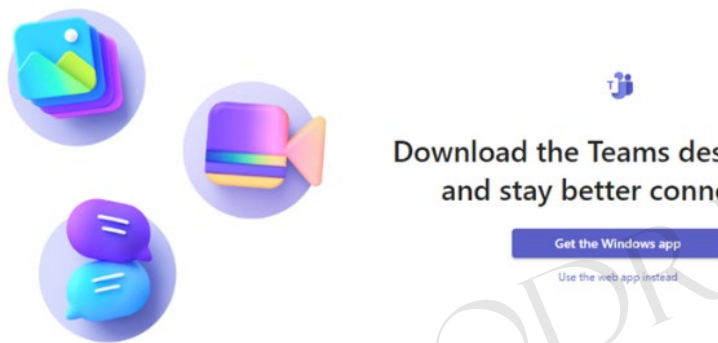
Design and implement source control

What is source control

Source Control

Versioning control system

FileA   Version 1

FileB   Version 1   Version 2   Version 3

FileC   Version 1   Version 2

1.0.0   FileA   FileB   FileC

1.0.1   FileA   FileB   FileC

1.1.0   FileA   FileB   FileC
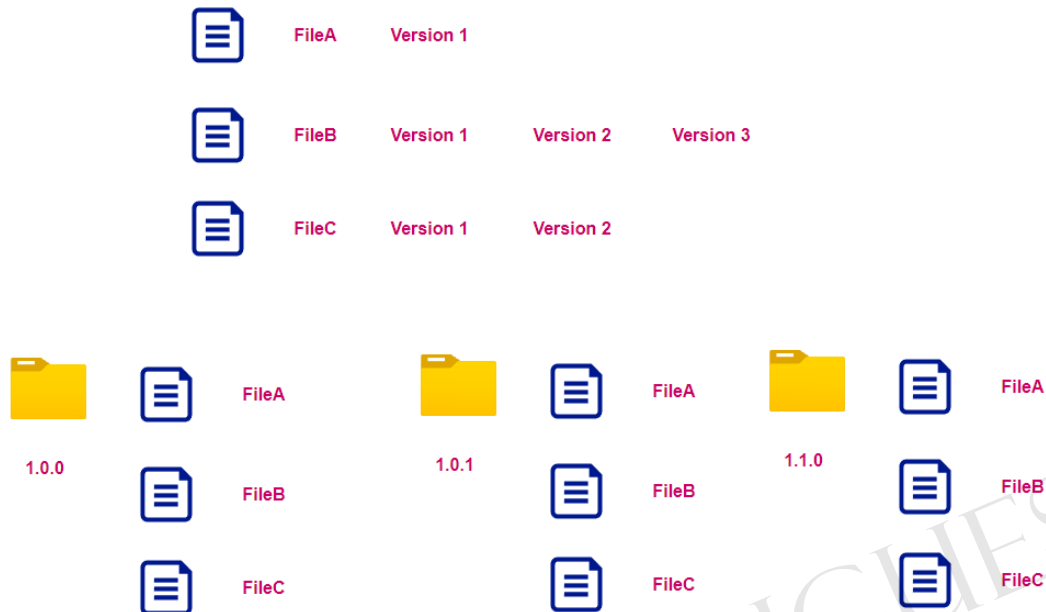
We need a tool that can help us maintain different versions of a file

It can also help us revert back to a prior version

Change 1

Change 2

Change 3

Change 4

I made a mistake, we need the change till Chnage 27 which seems to have the correct code for a partciular module. How can we retrieve that change.

Application Code

Change 50

Developers

What is Git

Source Control

Versioning control system

FileA   Version 1

FileB   Version 1   Version 2   Version 3

FileC   Version 1   Version 2

Git is the most commonly used version control system

This is because of its distributed nature.

Subversion

Central Server

Subversion Server software

Subversion client

Checkout the files required.

Make changes to the code.

Check in the code with the required changes.

Application Code

Subversion client

Subversion client

Git

Git client

Git client

Git client

Application Code

Application Code

Application Code
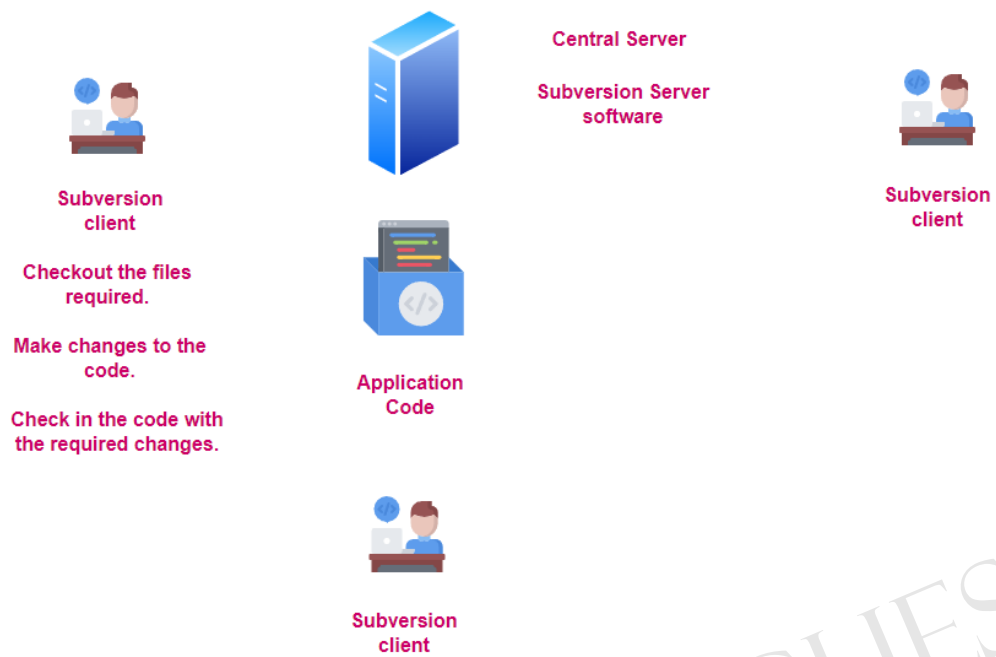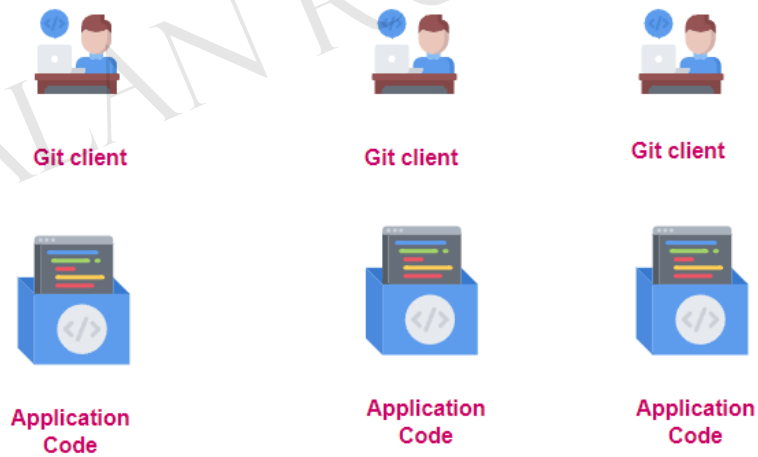
Other advantages

Better overall performance

Easily manage aspects like creating branches

Adding to the git repository

Using Git

.git folder

		Git Version 1	Git Version 2	Git Version 3
	FileA	Version 1	Version 2	Version 2
	FileB	Version 1	Version 1	Version 2

Local database

Maintains the entire history of your changes

Contains information on the commits, remote repository address

1. Install the git tool

2. Initialize an empty repository onto your file system

3. Start working with your files on your local system

Working Directory	Staging Area	.git directory (Repository)

FileA (Version 1)

FileB (Version 1)

git add command

Working Directory	Staging Area	.git directory (Repository)

FileA (Version 1)

FileA (Version 1)

FileB (Version 1)

FileB (Version 1)

git commit
command

Working Directory

Staging Area

.git directory
(Repository)

FileA (Version 1)

FileA (Version 1)

FileA (Version 1)

FileB (Version 1)

FileB (Version 1)

FileB (Version 1)

Git branches

Branches

main

Initial commit

HEAD

app

**Code base for
an application**

Initial commit

merge with
your main
branch

feature branch

New feature
request

Using a remote repository

Git

Git client

Git client

Git client

Application Code

Application Code

Application Code

Git client

Remote repository

Git client

Application Code

Application Code

Application Code

1. Clone the remote repository
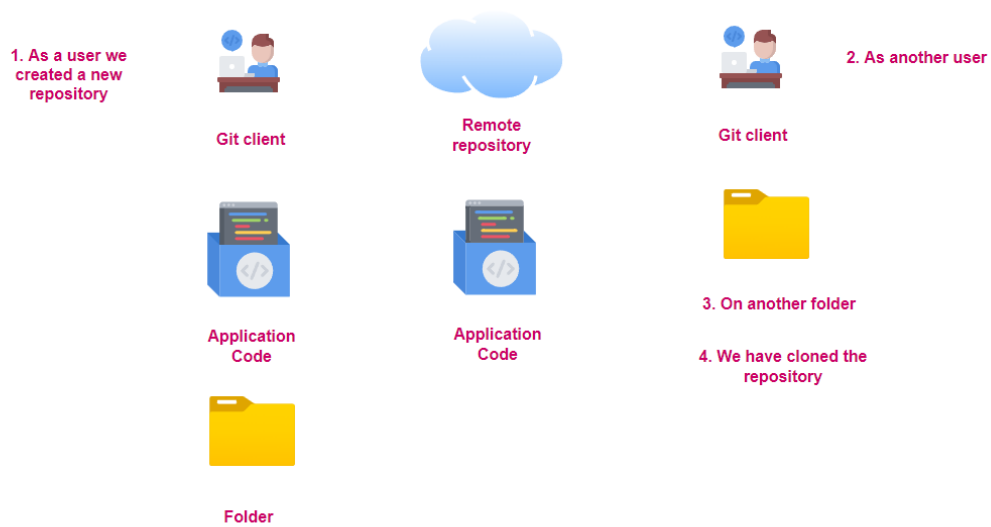
2. Make changes to the local repository

3. Merge the changes from the local to the remote repository

1. Clone the remote repository

2. Make changes to the local repository

3. Merge the changes from the local to the remote repository

Now let's clone the repository as another user

1. As a user we created a new repository

Git client

Remote repository

2. As another user

Git client

Application Code

Application Code

3. On another folder

4. We have cloned the repository

Folder

More on merges

We have seen the merge in Git working in 2 ways

Developer 1

main branch

Merge has taken place on Github

main branch

Developer 2

main branch

feature branch

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

PS C:\Application> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Application> git log --pretty=format:"%h %s" --graph
* d8af16d Added FileC
* cc636ac Made a change to FileA
* 872b55b This is the initial commit
PS C:\Application> git log --graph
* commit d8af16d7f8092531b2507ef9315524b2477929bd (HEAD -> main, origin/main, FeatureA)
| Author: Alan <techsup4000@gmail.com>
| Date:   Sat Jul 16 14:58:41 2022 +0400
|
|     Added FileC
|
* commit cc636aca1f0ad304bc92c7fcf88e7b2d2557ef30
| Author: Alan Rodrigues <techsup4000@gmail.com>
| Date:   Sat Jul 16 13:01:26 2022 +0400
|
|     Made a change to FileA
|
* commit 872b55b8641cb4a4de54c89565e9c6faab20c481
  Author: Alan Rodrigues <techsup4000@gmail.com>
  Date:   Sat Jul 16 11:25:16 2022 +0400

      This is the initial commit
PS C:\Application> []
```

Earlier example

1. We had the main branch

2. We created the FeatureA branch

3. In FeatureA branch we added a File - FileC.txt

4. We then merged FeatureA and the main branch.

5. When we made changes to the FeatureA branch, there were no changes made to the main branch during that time.

Hence a simple fast-forward merge happened wherein the main branch just pointed to the latest commit of the feature branch.

Branching strategy

main branch
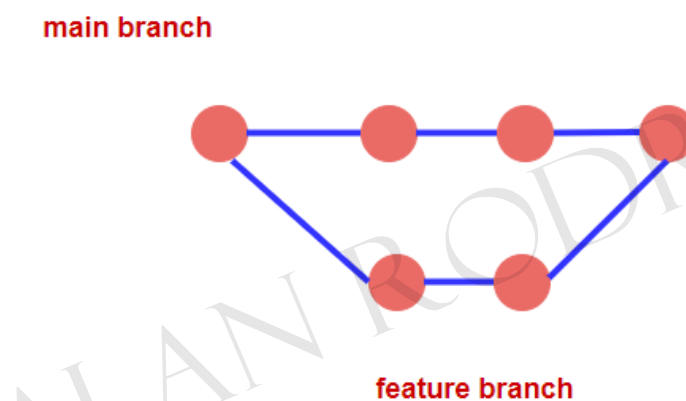
feature branch

1. Always maintain a high quality for your main branch

2. Contain the working copy of your production code

3. Create feature branches for your features and bug fixes

4. Use pull requests to merge your feature branches into your main branch.

5. Don't create long feature branches, keep them short-lived.

feature branch

release branch
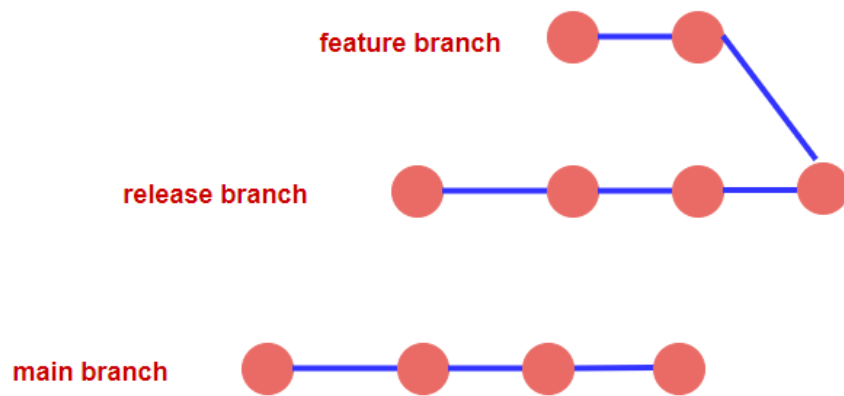
main branch

You also need to ensure that your feature branch is merged onto both your release and main branches

Here you need to maintain your release and main branches as well.

Design and implement build pipelines

What is Continuous Integration

Automatically building and testing code every time a team member commits code changes to version control.

Our local machine

In Visual Studio , when we run an application, a build of the application is conducted first.

The build tool builds the project and the depedencies into a set of binaries.

The binaries can then run on the target machine with the help of the .NET runtime.

app

Coded an application using the .NET 6.0 SDK

app

The machine needs to have .NET runtime installed to run the application.

Deploy the application on a virtual machine

Now going back to continuous integration

app

You normally have multiple developers working on an application

Each developer might be working on multiple changes that can take time.

app

Developer pusblishes the updates.

If there is an issue it becomes difficult to diagnose the issue

8 changes

3 weeks

app

Developer publishes the update to the code base

1 change

2 days

Automation is key

| Changed Code is published | The code needs to be built | The code needs to be tested |

And this is the concept behind continuous integration

Self-hosted agent

```
1   # ASP.NET Core (.NET Framework)
2   # Build and test ASP.NET Core projects targeting the full .NET Framework.
3   # Add steps that publish symbols, save build artifacts, and more:
4   # https://docs.microsoft.com/azure/devops/pipelines/languages/dotnet-core
5
6   trigger:
7   - main
8
9   pool:
10    vmImage: 'ubuntu-latest'
11
12  variables:
13    solution: '**/*.sln'
14    buildPlatform: 'Any CPU'
15    buildConfiguration: 'Release'
16
17  steps:
    Settings
18  - task: UseDotNet@2
19    displayName: Install .NET 6
20    inputs:
21      packageType: 'sdk'
22      version: 6.x
23
    Settings
24  - task: DotNetCoreCLI@2
25    displayName: Build
26    inputs:
27      command: build
28      projects: '**/*.csproj'
29      arguments: '--configuration $(buildConfiguration)'
```

The entire pipeline is making use of a Microsoft hosted agent.

This is the simplest way to run the pipeline.

A fresh virtual machine is created for each new job of the pipeline.

After the job is run, the virtual machine is discarded.

All maintenance and upgrades are handled by Microsoft when it comes to the hosted agents.

You want to create your own agent.

You want to persist the builds.

Have custom software installed.

Security at every stage

Security should never be an after thought

It needs to be embedded at every possible stage in your lifecycle.

Planning stage

If its a web application, what are the possible threats.

Possible ways to mitigate those threats.

Development

Using security plug-ins in the Intgerated development environment.

Conduct peer reviews.

Adhere to coding standards.

Build

Continuous Integration

Have tests within the pipeline.

Unit tests.

Static Code Analyzers.

Deployment

Penetration testing.

Infrastructure scanning.

What have we seen so far

Azure Repos

Makes a change to the code base

Azure Pipelines

The pipeline can build the application

It can trigger a pipeline

Azure Repos

Makes a change to the code base

Azure Pipelines

Unit tests

Test for libraries

It can trigger a pipeline

We can add security to our pipeline

GitHub

Makes a change to the code base

Azure Pipelines

The pipeline can build the application

It can trigger a pipeline

GitHub

Jenkins
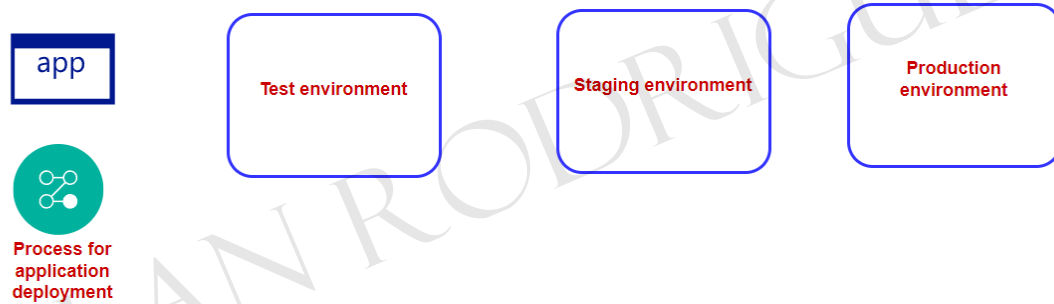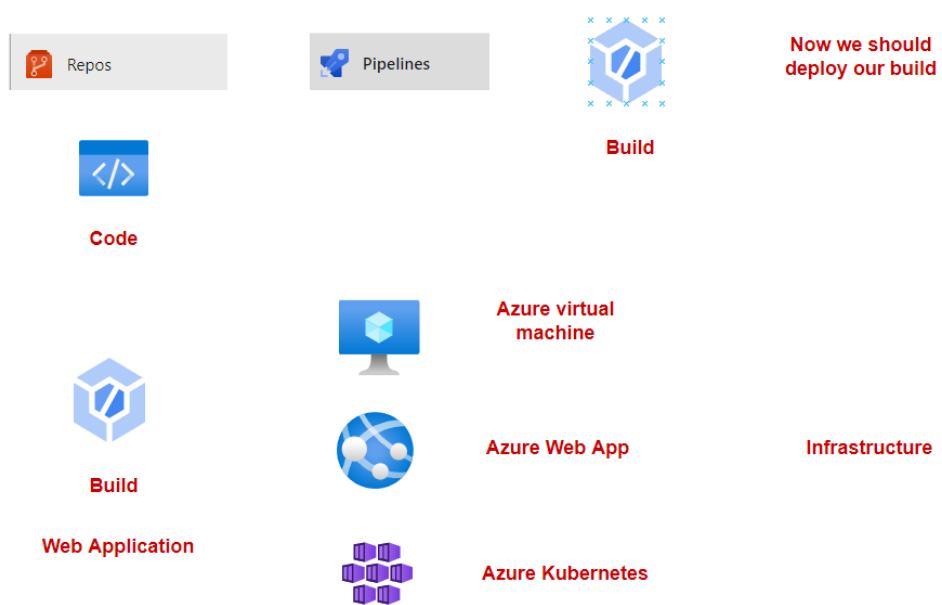
Continuous Integration tool
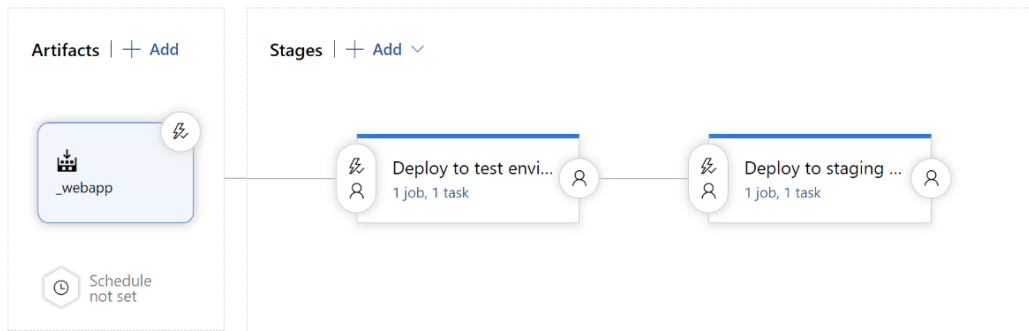
The pipeline can build the application

Azure Repos

Virtual Machine

Design and implement release pipelines

Understanding deployment

Repos

Pipelines

Build

Now we should deploy our build

Code

Azure virtual machine

Build

Azure Web App

Infrastructure

Web Application

Azure Kubernetes

app

Test environment

Staging environment

Production environment

Process for application deployment

Multiple stages in the pipeline

Artifacts | + Add

_webapp

Schedule not set

Stages | + Add ∨

Deploy to test envi...
1 job, 1 task

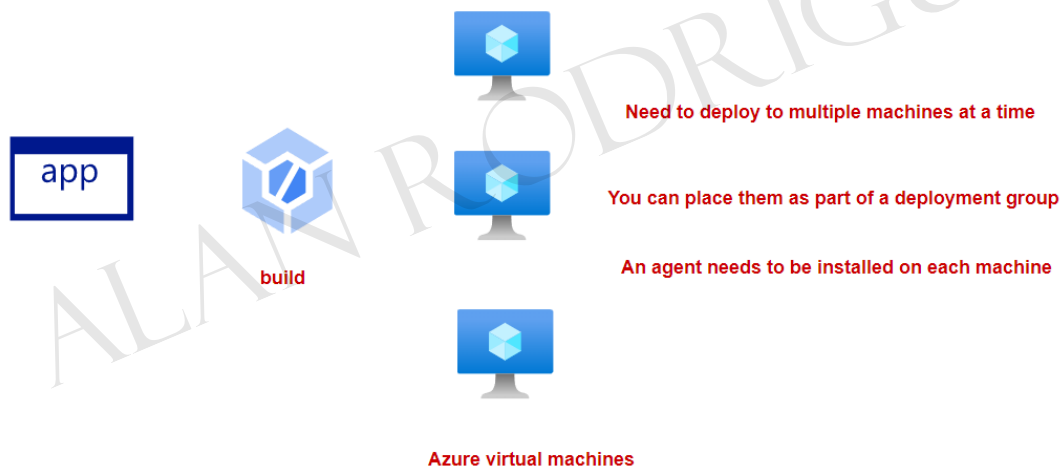Deploy to staging ...
1 job, 1 task

<u>Pre-deployment approval</u>
A user needs to validate
the change request so that
the deployment can be
done to that stage

<u>Post-deployment approval</u>
A user needs to sign-off on
the deployment done so
that it can move to another
stage.

<u>Pre-deployment gate</u>
A user wants to ensure
that no work items are
active for the build to be
deployed to a stage.

<u>Post-deployment gate</u>
A user wants to ensure
that no issues are tagged
against a deployment
before it can proceed to
the next stage.

Lab - Deployment Groups – Implementation

app

build

Need to deploy to multiple machines at a time

You can place them as part of a deployment group

An agent needs to be installed on each machine

Azure virtual machines

Azure Web App - Azure SQL database

Azure Web App

**Azure SQL
database**

Table

1. Build the ASP.NET Core
application

2. Release pipelines - 2 tasks

2a. Deploy the table and data in the
Azure SQL database.
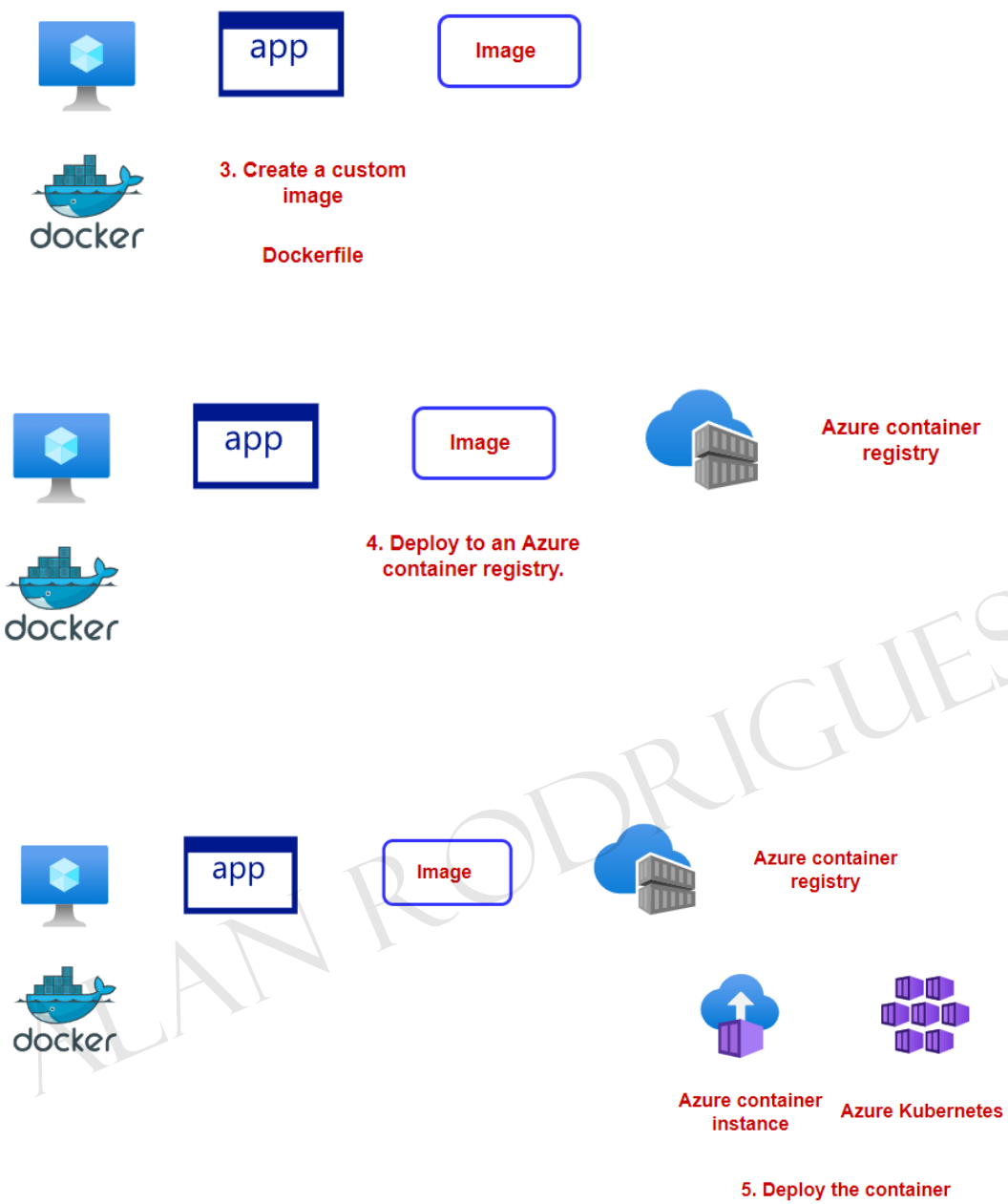
2b. Deploy the .NET application to
the Azure Web App.

The next step – Containers

1. Install Docker on a virtual machine

2. Publish the .NET application onto the VM.

3. Create a custom image

Dockerfile

4. Deploy to an Azure container registry.

Azure container registry

Azure container registry

Azure container instance

Azure Kubernetes

5. Deploy the container

About Container jobs

Container jobs

The compute instance in this case is a Microsoft-hosted agent

Your build runs on a compute instance

The agent has certain capabilities

The containers have their own tools required for the build to run.
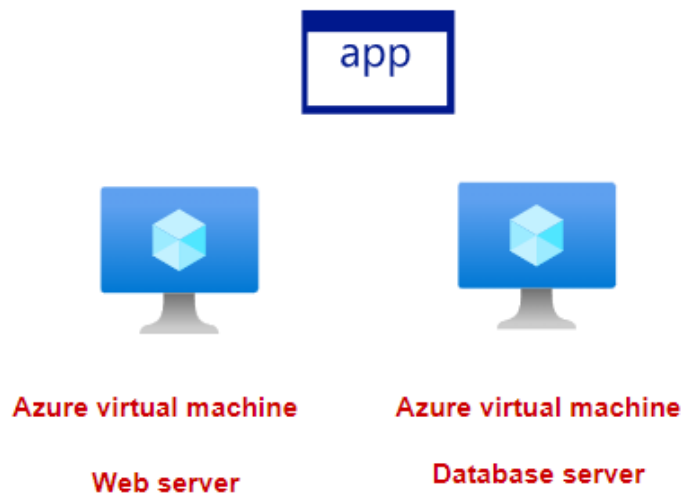
Container

Container

Container

Design and Implement Infrastructure as Code

About managing infrastructure

Managing infrastructure

app

Azure virtual machine

Web server

Azure virtual machine

Database server

Azure Web App

Azure SQL database

1. Creating your infrastructure

2. Making changes

3. Replicating the same infrastructure across multiple environments

4. Sometimes you might not document properly how you built your infrastructure

You develop your infrastructure as code

This helps clearly define how you infrastructure should look like

You can easily make changes to your infrastructure with the help of changes to the code file

You can also version control the code file that is used to define your infrastructure

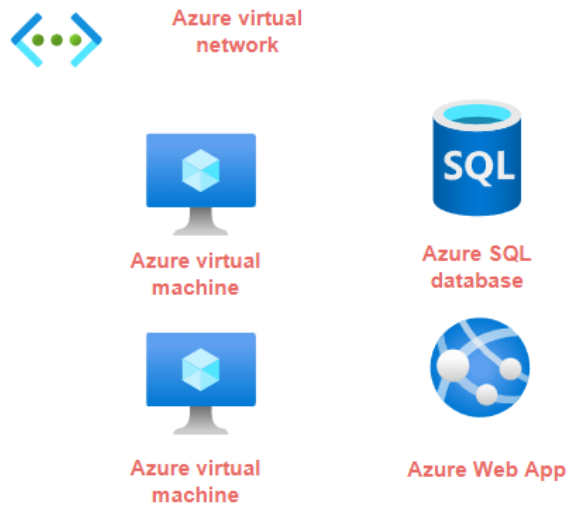Replicating your infrstructure across multiple environments

Azure Web App

Azure SQL database

Development

Azure Web App

Azure SQL database

Staging

Azure Web App

Azure SQL database

Production
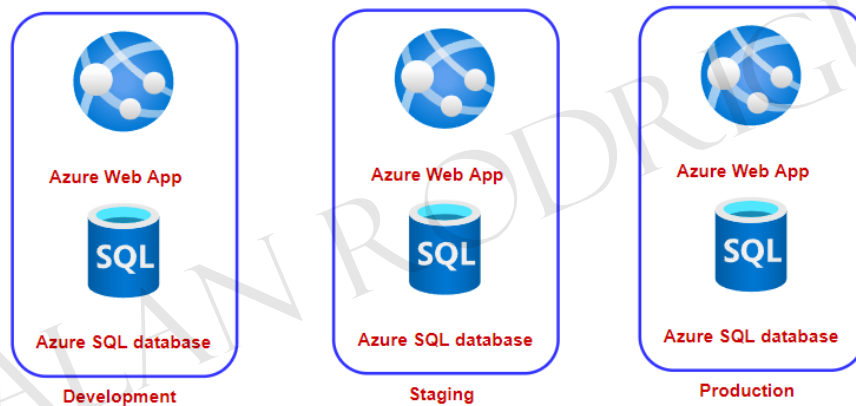
ARM Templates

Bicep

Terraform

Remember you can also create scripts to manage your Azure resources

Azure CLI

Azure PowerShell modules

About ARM templates

Azure virtual network

Azure virtual machine

Azure SQL database

Azure virtual machine

Azure Web App

You define your infrastruture as code

Create an Azure Resource Manager template

This is a JavaScript Object Notation file that actually contains the definition of the infrastructure

You can store the ARM templates in your source code repository along with your application code

Microsoft has also release a new language called Bicep that has the same capabilities as ARM templates.

Bicep just uses a syntax that is easier to use.

Modularize templates

Azure virtual network

Azure virtual machine

Azure SQL database

Azure virtual machine

Azure Web App

Sometimes your templates can become really large because of the number of resources that need to be deployed.

In this case, you can see whether you can create nested or linked templates.

Here the definition of the nested resource is defined withint the main template itself.

Nested Template

Here resources are defined as different templates. And the templates are referenced within other templates.

Linked Template

Deployment modes

Incremental mode - Here resources defined in the template are deployed. It does not interfere with other resources defined in the resource group which are not defined in the template.

Complete mode - Here resources defined in the template will only be in the resource group. If there are other resources in the resource group, they will be deleted.

About Terraform

Deploying your infrastructure

Application Server

Database server

1. Deploy the infrastructure

2. Configure your infrastructure

Infrastructure as code

Code file

Application Server

Database server

Change the code whenever required

Share the code

Create different versions of the code

What makes Terraform popular

It works with a variety of cloud platforms.

The code is human-readable

Avid community

Open-source project

1. Write your Terraform configuration file

This defines the resources that need to be deployed

Application Server
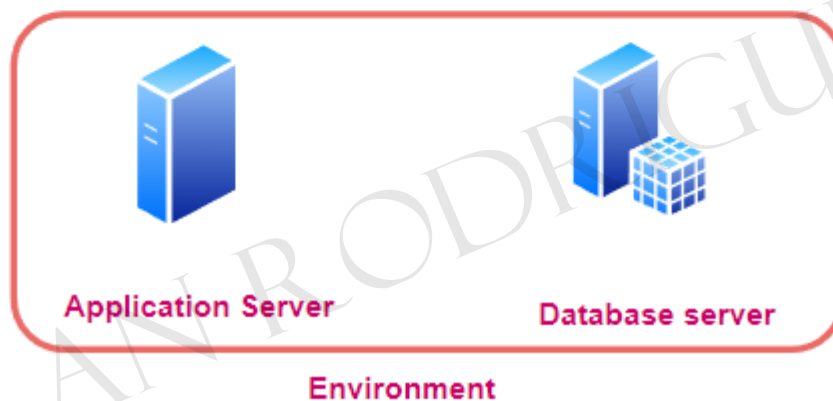
Database server

Environment

2. Terraform plan

Terraform looks at the configuration file and decides what needs to be deployed or changed.

It actually maintains a state file.

3. Terraform apply

Apply all of the changes as per the Terraform configuration file

Application Server

Database server

Environment

Configuration File

Terraform binary

Provision the infrastructure as per the details specified in the configuration file.

Desired State Configuration

The machine has tools or software installed to support the application.

Azure virutal machine

For example - Internet Information Services for a web application.

It has to be ensured that the configuration of the machine does not drift or it could impact the application.

For this you can make use of PowerShell DSC

Azure Automation DSC

Here there is an in-built server that ensures all machines receive the desired configurations.

You can manage all configurations and nodes from the Azure Portal.

Using VM extensions

Custom Script Extension

Extension

This extension can be used to download and run scripts on the Azure virtual machine.

Example - Used for post-deployment configuration or software installation.

Extension   **Register the machine with a deployment group**

Extension   **Install Internet Information Services and ASP.NET Hosting Bundle.**

Virtual Machine Scale Sets

Azure virtual network

Azure virtual machine scale set

Azure Load Balancer

Condition

Metrics

Public IP Address

Deployments, Packages and Test Plans

The fear of changes

Azure Web App

Azure SQL database

User's ask for changes to the application

Make code changes that need to be deployed

But there is always that fear factor

What happens if the change fails

The application stops working

Customers are not happy

Risk to the business

That is why companies strive a lot to make changes as seamless as possible

Blue-Green Deployments

Blue environment

Green environment

All user traffic is directed to
the Blue environment which
is the current production
environment.

If a software change needs
to be made , then a duplicate
environment is setup. The
changed application is
deployed to this
environment.

Test is conducted on this
environment.

Blue environment

Green environment

Once testing is completed , users are then directed to the
Green environment.

This now becomes the production environment.

Azure Web App - Deployment Slots

Deployment Slots

Staging Environments for App Service Plans

app

Standard , Premium and Isolated App Service Plan

Applications in deployment slots have their own host names

Version 1

Version 2

Production Slot

Staging slot

1. You have the chance to validate all application changes in the staging deployment slot

2. You can then swap the staging slot with the production slot

3. This helps eliminate the downtime for your application when new changes are deployed

4. You can also easily roll back the changes

Canary deployments

app

When a new feature is released for the application, it is only made available to a subset of users.

Canary deployments

You can direct a percentage of users onto maybe a new environment with the new feature.

Or maybe use feature toggles to toggle a feature on or off.

Production Environments

Users are directed to the production environment

Azure Traffic Manager

Azure Traffic Manager

This is a DNS-based traffic load balancer.

You can distribute traffic to public facing applications across different Azure regions.

You can direct traffic based on different routing methods.

North Europe

Azure Web App

Azure Traffic Manager Profile

Priority Routing Method

UK South

Azure Web App

North Europe

Azure Web App

Azure Traffic Manager Profile

Weighted Routing method

UK South

Azure Web App

Using a rolling deployment

Rolling deployments

app
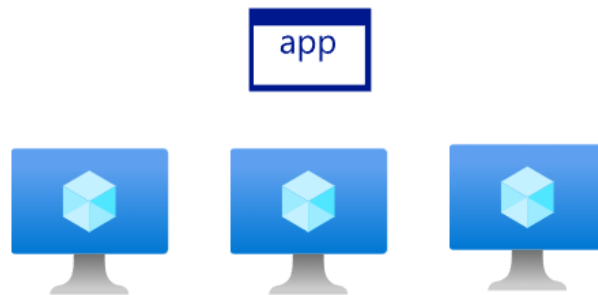
Azure virtual machines

Here the application is running
on a set of virtual machines

Application - Version 1

app

app

This machine would replace one
machine in the existing set.

So this is done in a rolling fashion across the machines with each
machine being replaced.

At any point in time, you would have machines having the older and
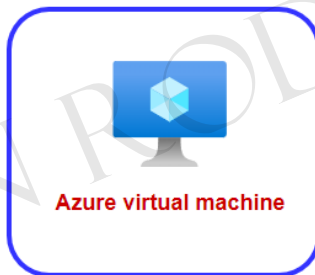newer application version.

Using a Load Balancer

Using an Azure Load Balancer

Azure Load Balancer

Health probe

Load Balancing Rules

Azure virtual machines

Backend pool

Azure virtual machine

All users are directed to the current production environment

Blue environment

Azure virtual machine
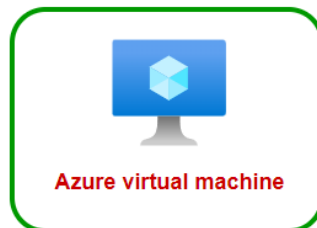
New application version

Green environment

Package Management

Azure App Config

Azure Web Apps

Feature Flag

Enabled/Disabled

Configuration settings

Azure App Config

Azure Artifacts

Newtonsoft.Json

NuGet.org	.NET
npm	JavaScript
Maven packages	Java
PyPI Python package Index	Python

Developers can publish their packages to Azure Artifacts

Other developers can then consume their packages from Azure Artifacts

Developers can additionally also consume their packages from the public feeds such as NuGet.org, npm etc.

Develop a security and compliance plan

Lab - Azure Key Vault

Azure Key Vault

app

Password

Key vault

Secrets

Using OWASP Tool - Build pipeline

OWASP Zap scanner

Agent job
Run on agent

- **Azure Web App Deploy: newapp787878**
  Azure Web App
- **Azure CLI - Storage account**
  Azure CLI
- **Azure CLI - Create container**
  Azure CLI
- **Azure CLI - Download report**
  Azure CLI
- **PowerShell Script - Convert report**
  PowerShell
- **Publish Test Results**
  Publish Test Results

Azure Web App

Run a scan against the Azure Web App

Container Instance

**Docker Image
The container will actually run the scan against the web site**
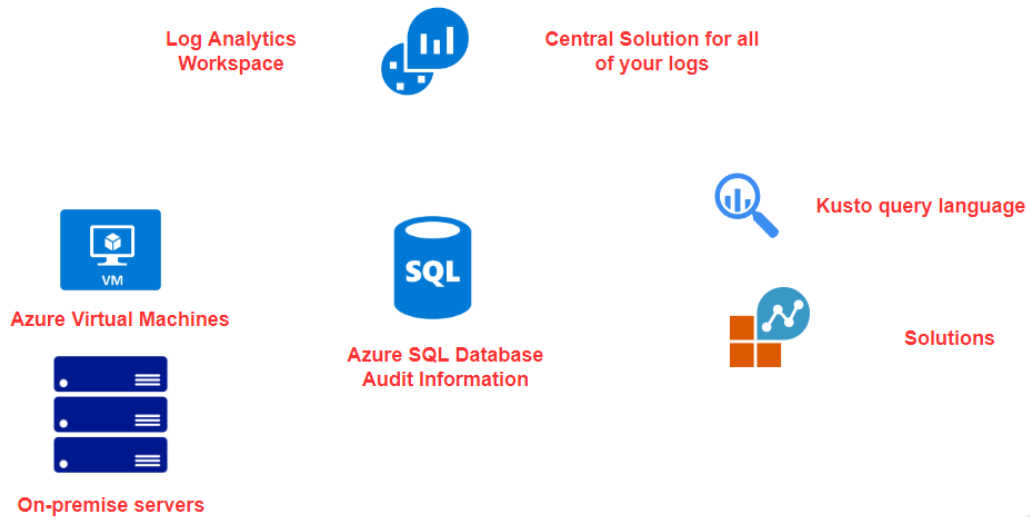
Azure Storage Account - File share

The results reports will be available in the file share

Implement an instrumentation strategy

Log Analytics workspace

Log Analytics Workspace

Central Solution for all of your logs

Kusto query language

Azure Virtual Machines

Azure SQL Database Audit Information

Solutions

On-premise servers