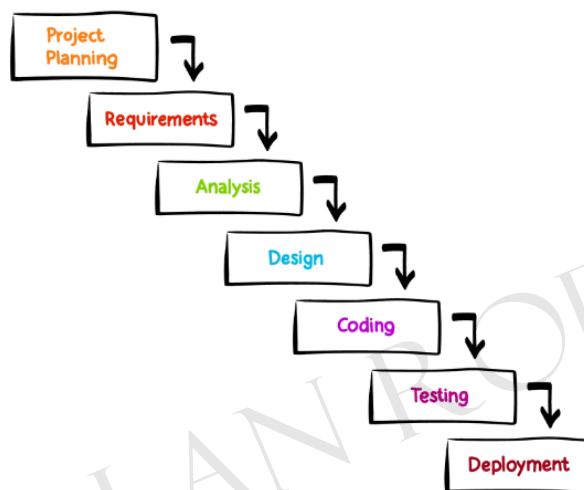


Important Note – All of the images in the diagrams should be used for individual learning purposes only. They are not to be re-distributed in any form or manner.

Introduction

Primer to DevOps - Project lifecycle

Waterfall-Model



Advantages

1. Clear demarcation between one phase and another
2. The requirements can be solidified up front.
3. It gives the Project manager a clear insight into aspects such as delivery dates, cost estimates etc.

Disadvantages

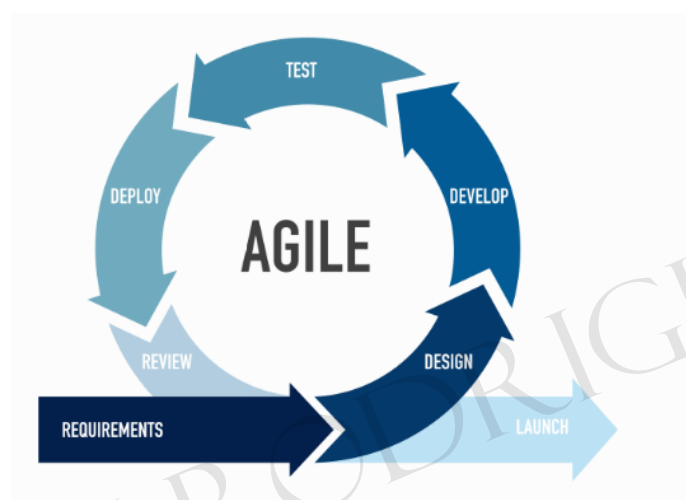
1. The customer might get to see the end product only at the end of the entire lifecycle
2. Changes in scope.
3. Bugs in the testing phase

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

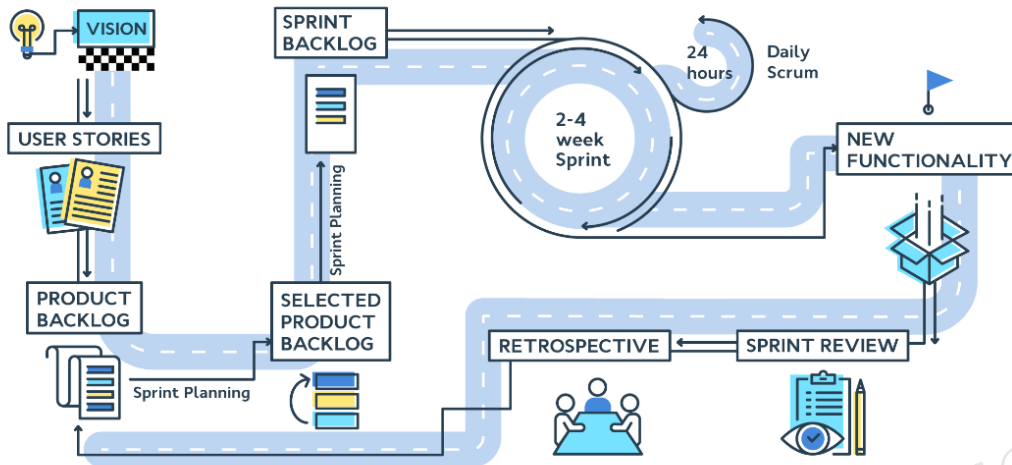


Primer to DevOps - Scrum process

Scrum

This is an agile development methodology

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.

Development team is developing an application



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



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 hesitant 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.

Primer to DevOps - About DevOps

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

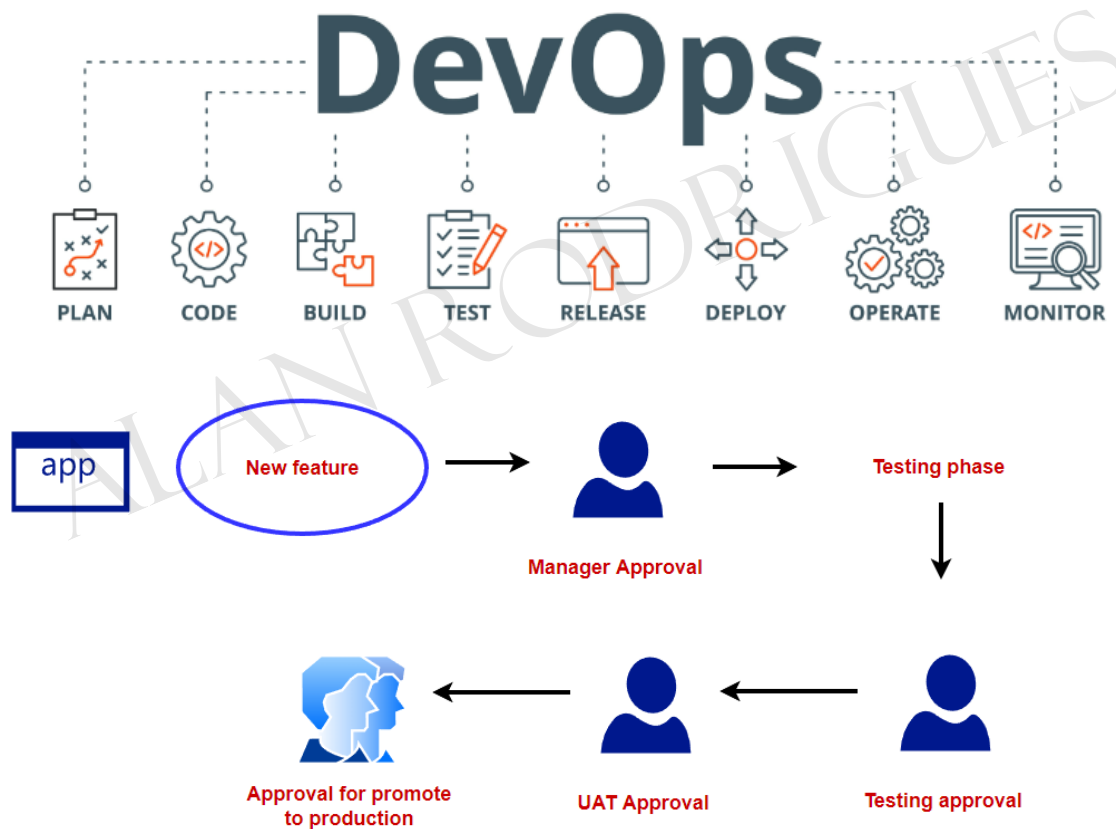
Meetings that are held with stakeholders normally have the Developers and Operations teams.

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

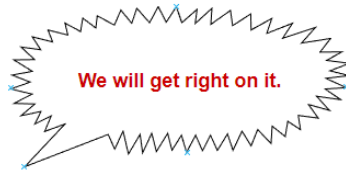
This company acts as the IT department for a larger organization

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





LEADERSHIP

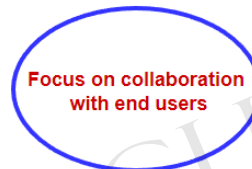


IT department



LEADERSHIP

Human Resource Business
department



Traditional Approach

Traditional Approach

Approach to the project



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

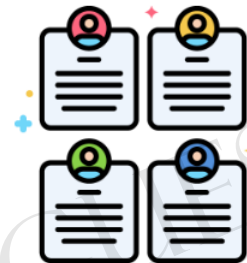


Scrum team

Small group of people focused on achieving an objective



Gather requirements from the users



User stories



Product Backlog



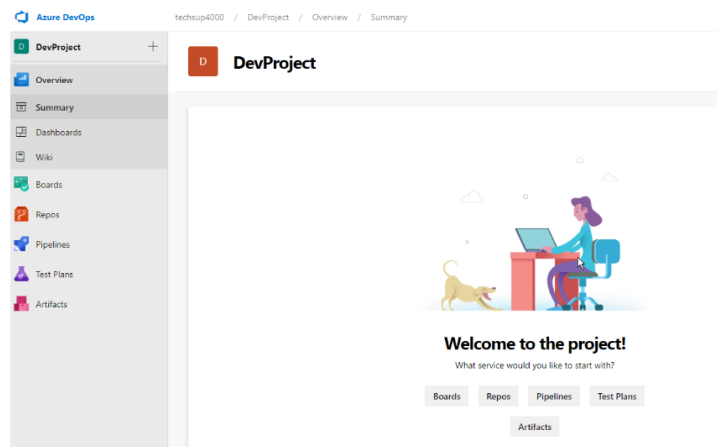
Sprint Backlog



Check the status of your sprints

We need a software to track all of this

Getting started with Azure Boards



Project

This provides a place for storing the source code. Here users can plan, track progress, collaborate.

When you create a project, a team of the same name as the project is created.

Organization

This can contain a group of related projects.

For example, you could have an organization for each business unit.

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

User Stories

1. Each department admin must be able to add a job posting
2. There should be an inbuilt workflow for the job posting
3. Once the job posting is published, no edits should be possible on the job posting

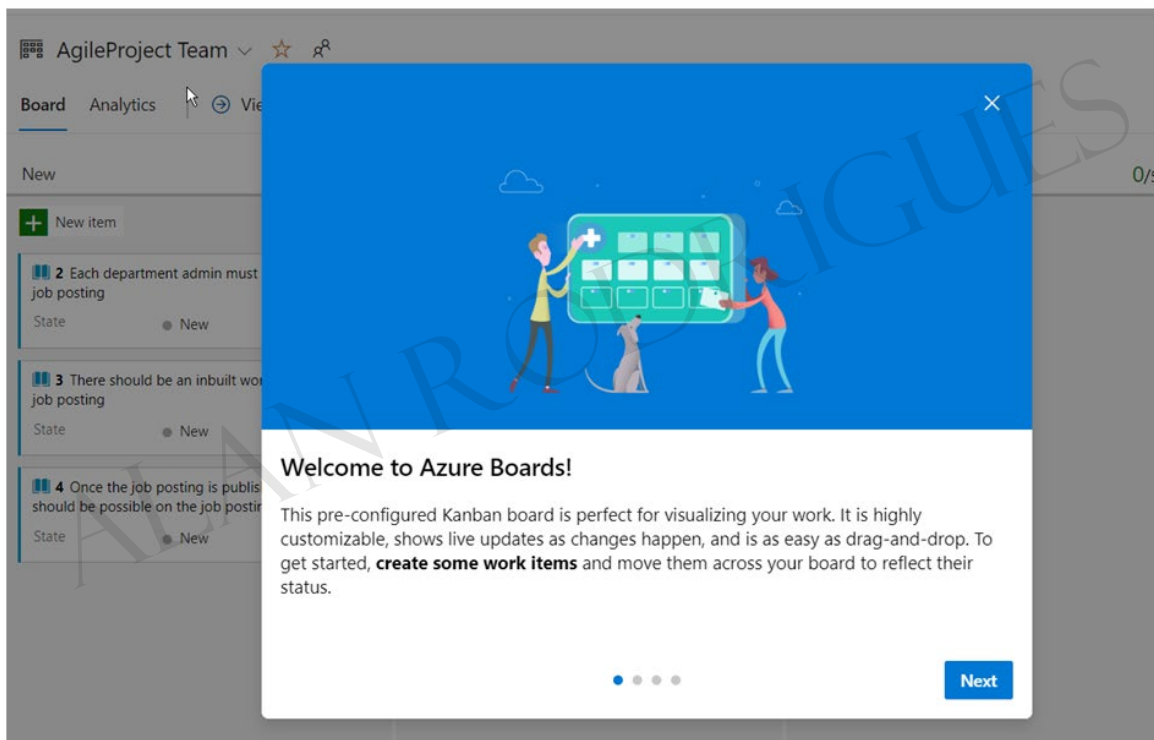
Epic

Jobs application - Job Postings

Kanban

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

Use of Kanban boards



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
- <i>TeamName</i> 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

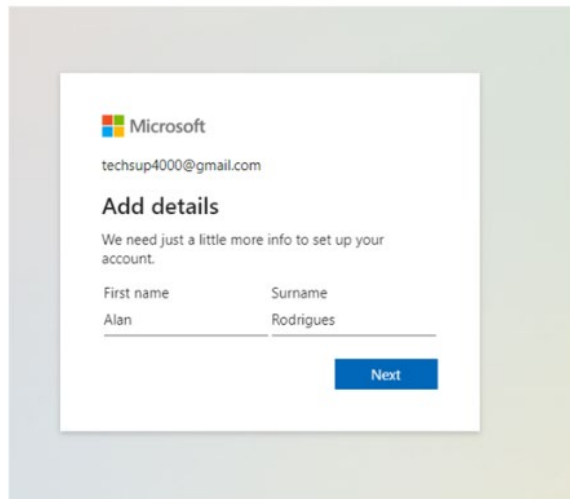
Microsoft Teams



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



Microsoft

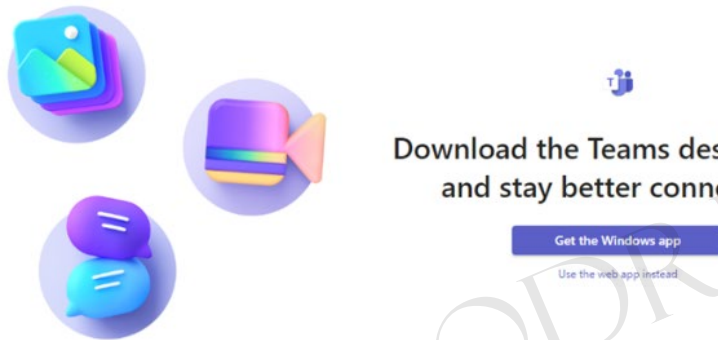
techsup4000@gmail.com

Add details

We need just a little more info to set up your account.

First name	Surname
Alan	Rodrigues

Next

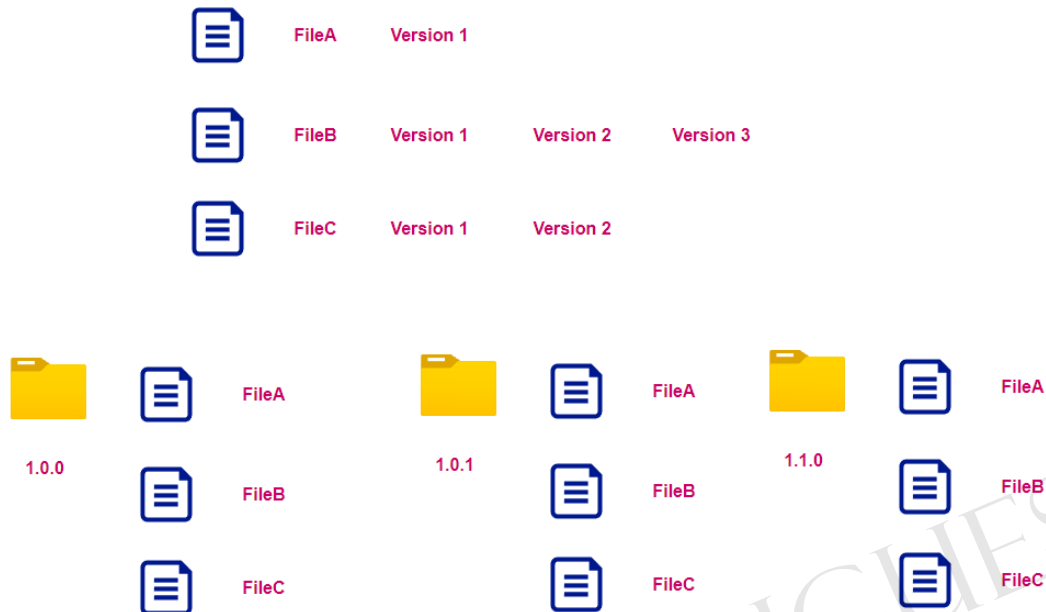


Design and implement source control

What is source control

Source Control

Versioning control system



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



What is Git

Source Control

Versioning control system



FileA

Version 1



FileB

Version 1

Version 2

Version 3



FileC

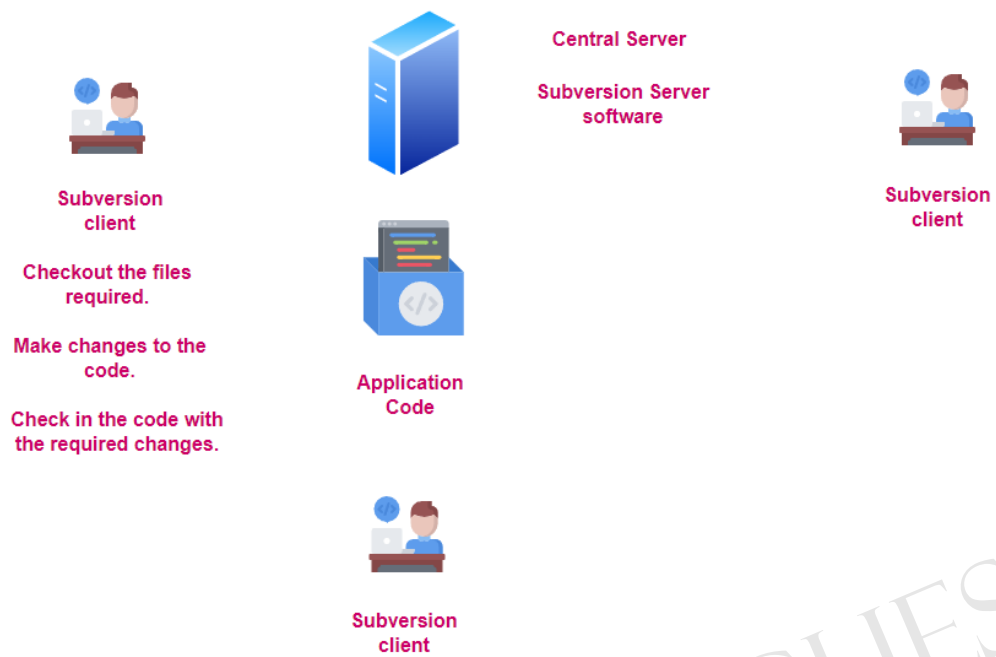
Version 1

Version 2

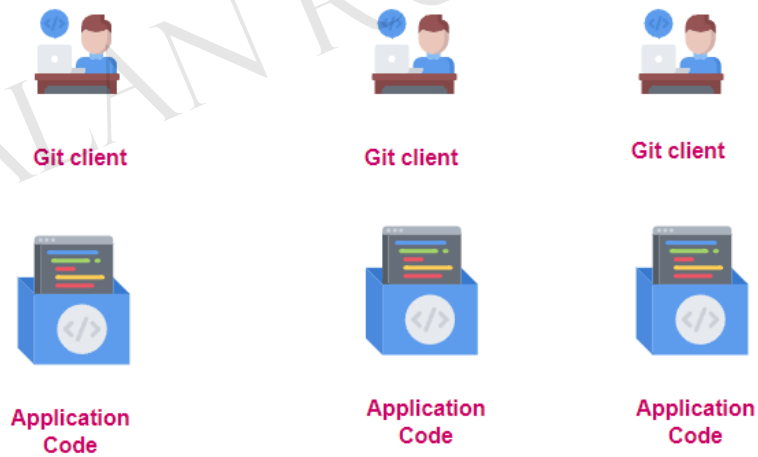
Git is the most commonly used version control system

This is because of its distributed nature.

Subversion



Git



Other advantages

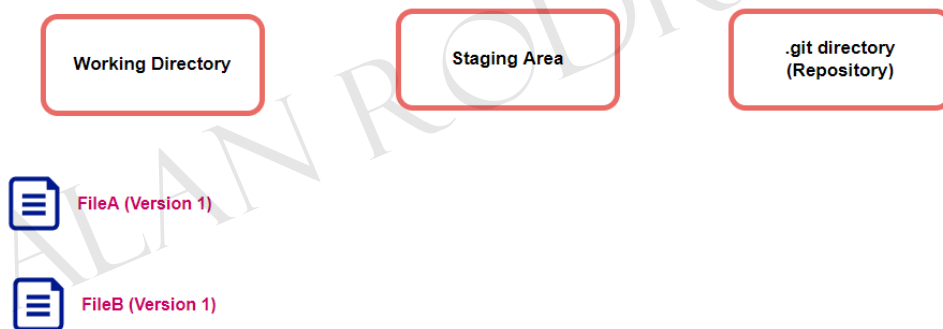
Better overall performance

Easily manage aspects like creating branches

Adding to the git repository



1. Install the git tool
2. Initialize an empty repository onto your file system
3. Start working with your files on your local system

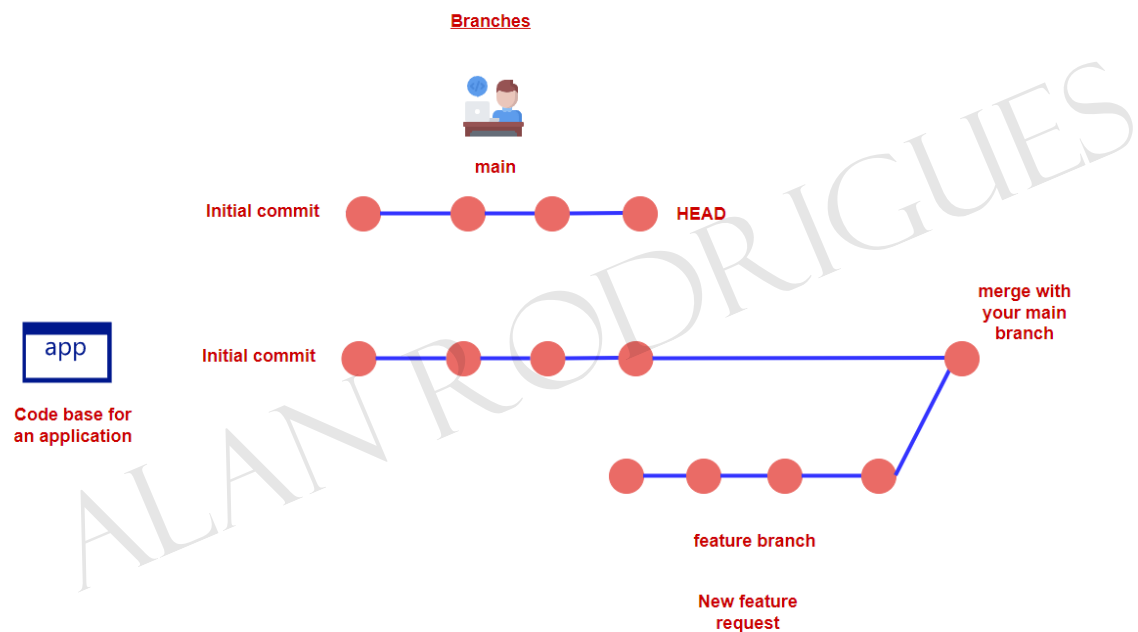


git add command





Git branches



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

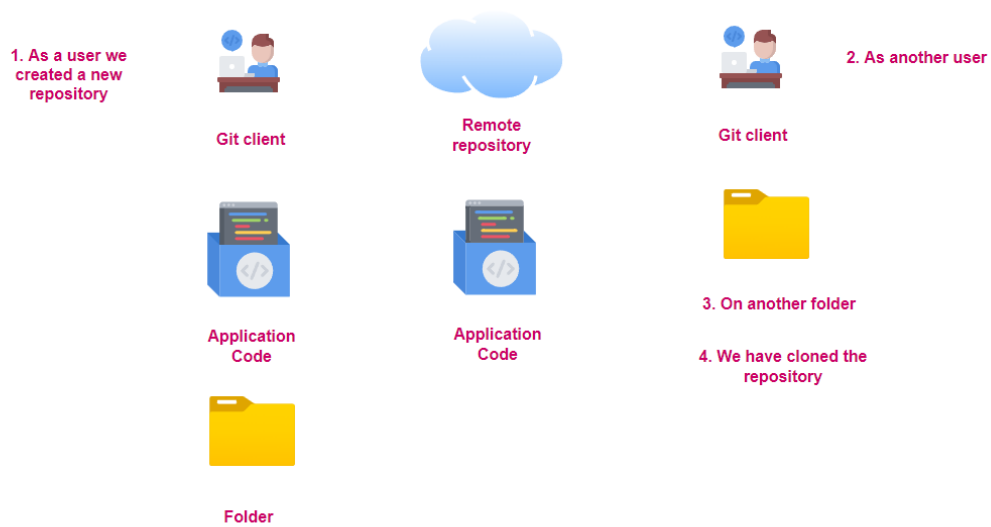
3. Merge the changes
from the local to the
remote repository

1. Clone the remote
repository

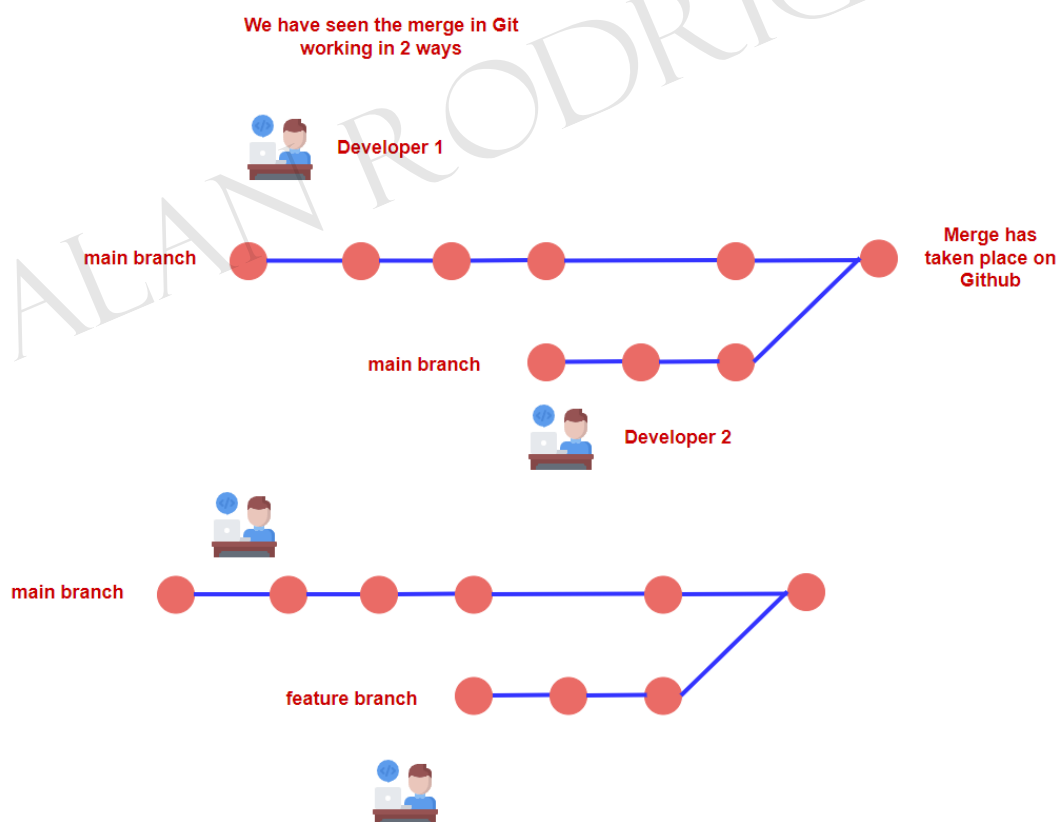
2. Make changes to
the local repository

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

Now let's clone the repository as another user



More on merges

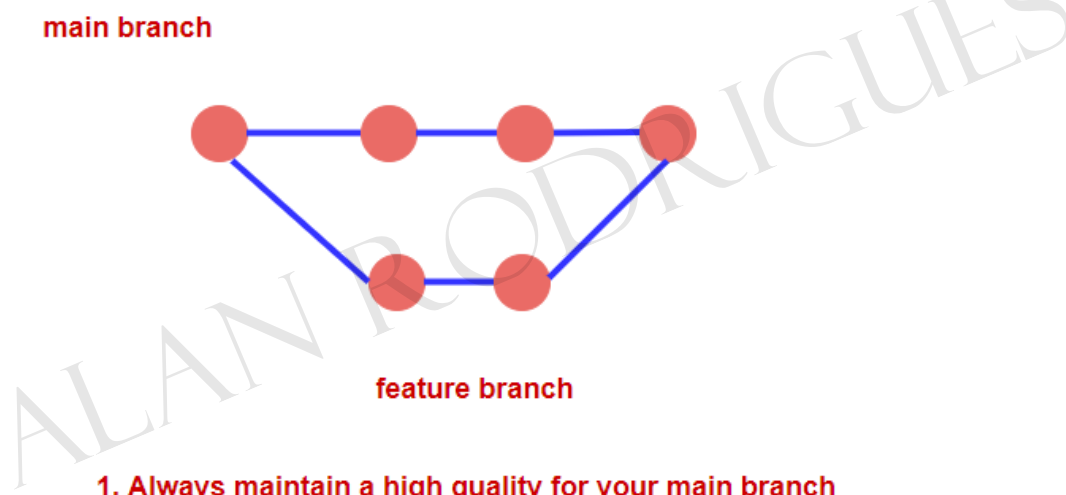


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Application> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Application> git log --pretty=format:"%h %s" --graph
* d8af16d Added FileC
* cc636ac Made a change to FileA
* 872b55b This is the initial commit
PS C:\Application> git log --graph
* commit d8af16d7f892531b2507ef9315524b2477929bd (HEAD -> main, origin/main, FeatureA)
| Author: Alan <techsup400@gmail.com>
| Date: Sat Jul 16 14:58:41 2022 +0400
|
| Added FileC
|
| * commit cc636aca1f8ad304bc92c7fcf88e7b2d2557ef30
| | Author: Alan Rodrigues <techsup400@gmail.com>
| | Date: Sat Jul 16 13:01:26 2022 +0400
| |
| | Made a change to FileA
| |
| * commit 872b55b8641cb4a4de54c89565e9c6faab20c481
| | Author: Alan Rodrigues <techsup400@gmail.com>
| | Date: Sat Jul 16 11:25:16 2022 +0400
| |
| | This is the initial commit
|
PS C:\Application> []
```

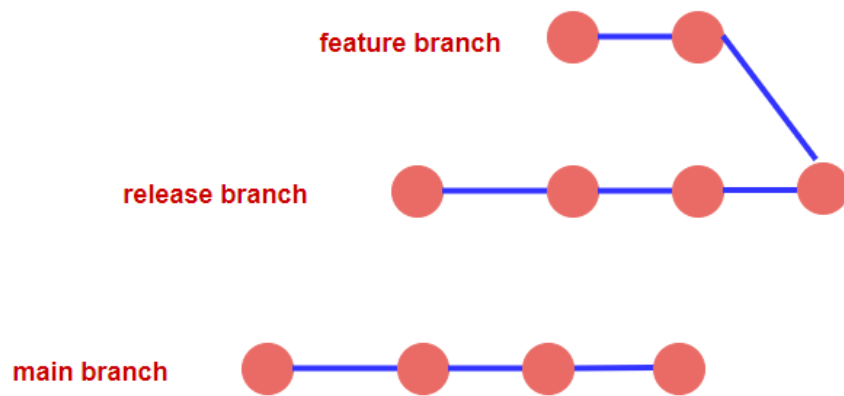
Earlier example

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



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.



**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

ALAN RODRIGUES

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 dependencies into a set of binaries.

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



Coded an application using the .NET 6.0 SDK



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



You normally have multiple developers working on an application

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



8 changes

3 weeks



Developer publishes the updates.

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



1 change

2 days



Developer publishes the update to the code base

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:
18   - task: UseDotNet@2
19     displayName: Install .NET 6
20     inputs:
21       - packageType: 'sdk'
22       - version: '6.x'
23
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



Azure Repos



Jenkins

Continuous Integration
tool

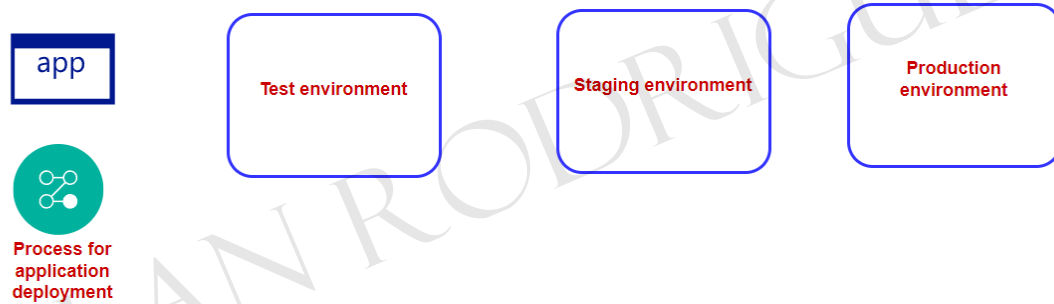
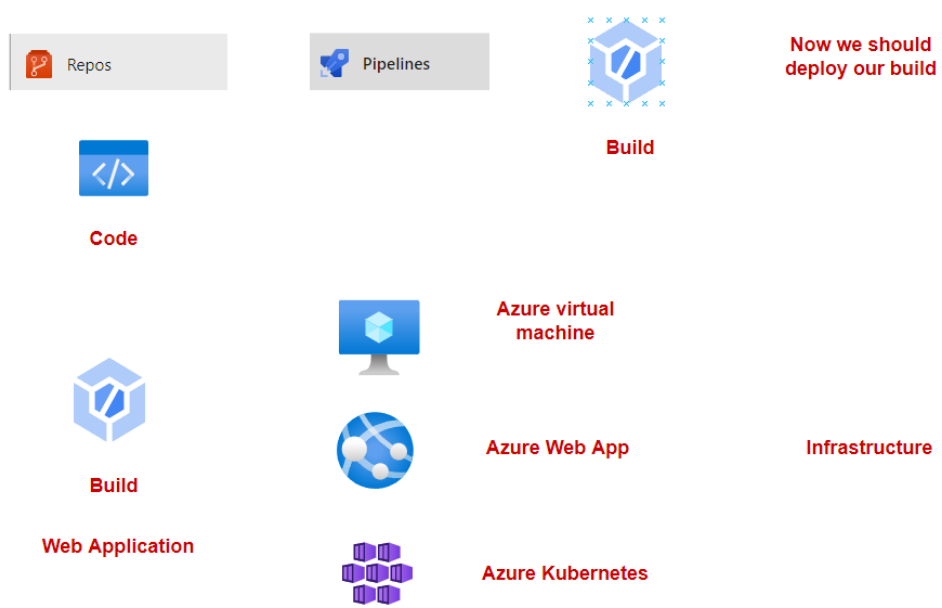
The pipeline can build
the application



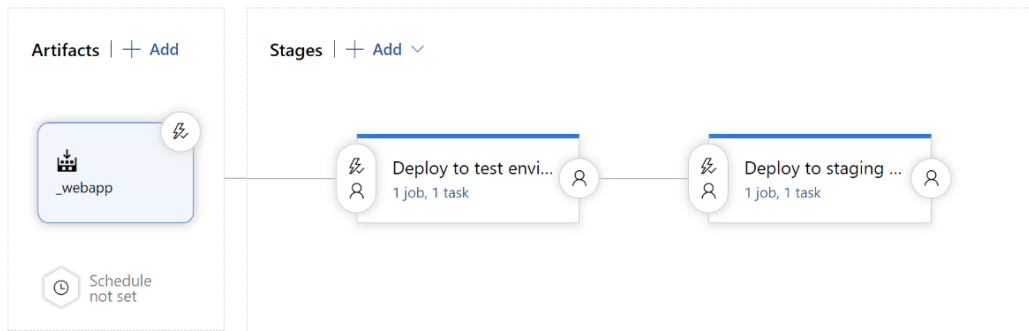
Virtual Machine

Design and implement release pipelines

Understanding deployment



Multiple stages in the pipeline



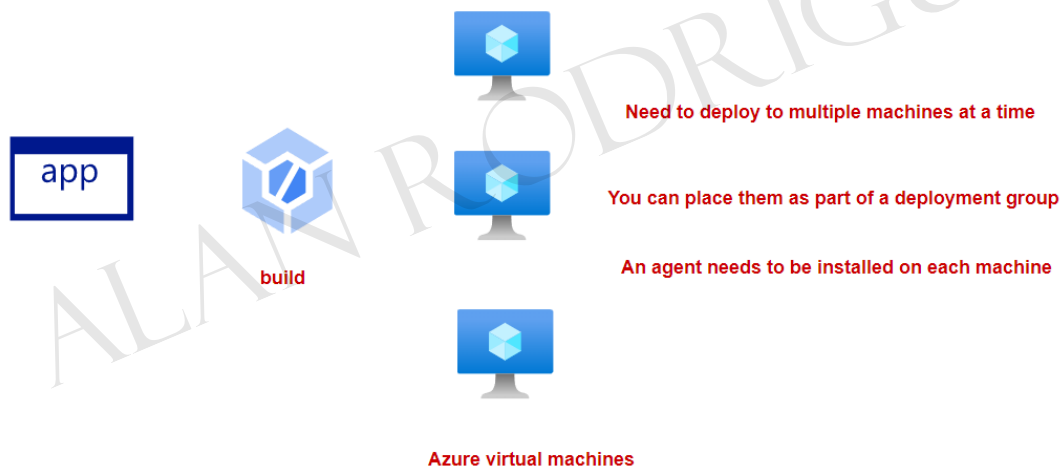
Pre-deployment approval
A user needs to validate the change request so that the deployment can be done to that stage

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

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

Post-deployment gate
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



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.**

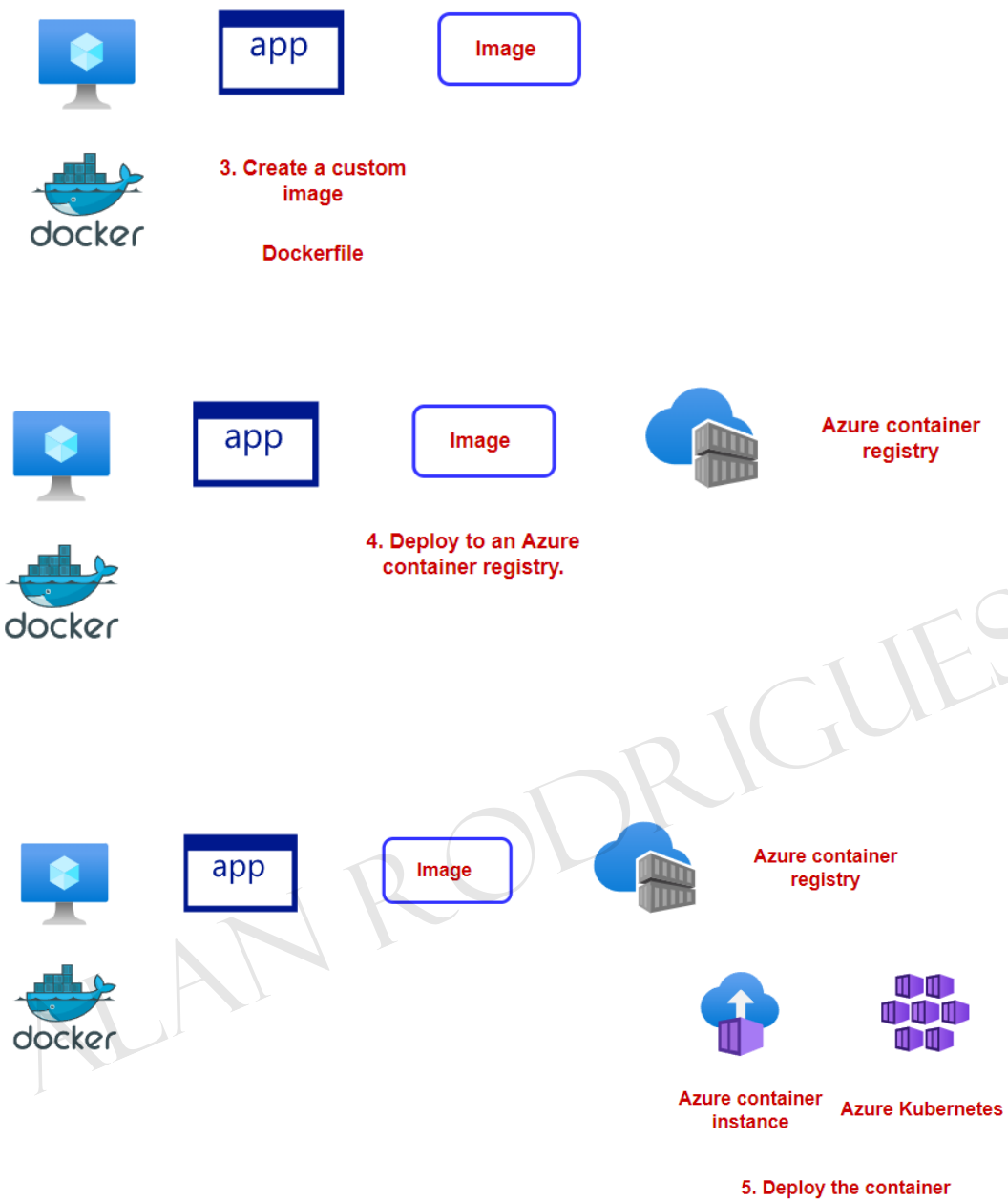
The next step – Containers



**1. Install Docker on a
virtual machine**



**2. Publish the .NET
application onto the
VM.**



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

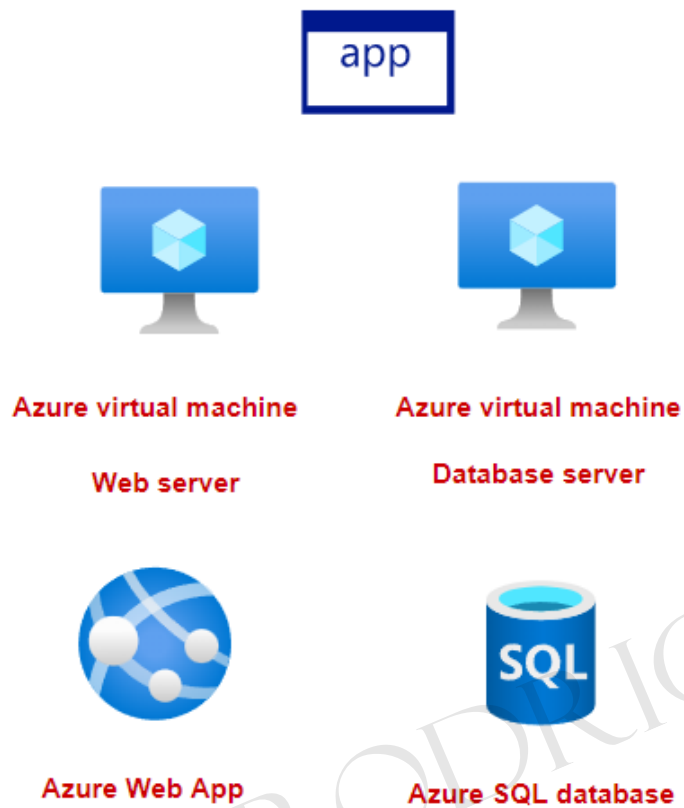
Container



Design and Implement Infrastructure as Code

About managing infrastructure

Managing infrastructure



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



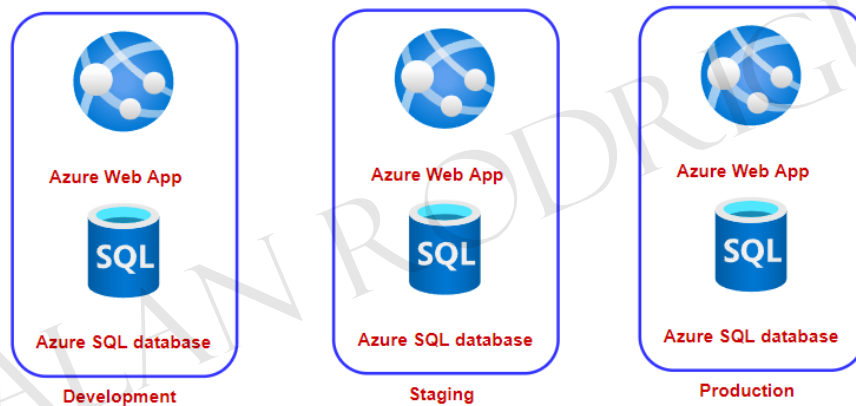
You develop your infrastructure as code

This helps clearly define how your 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 infrastructure across multiple environments



ARM Templates

Bicep

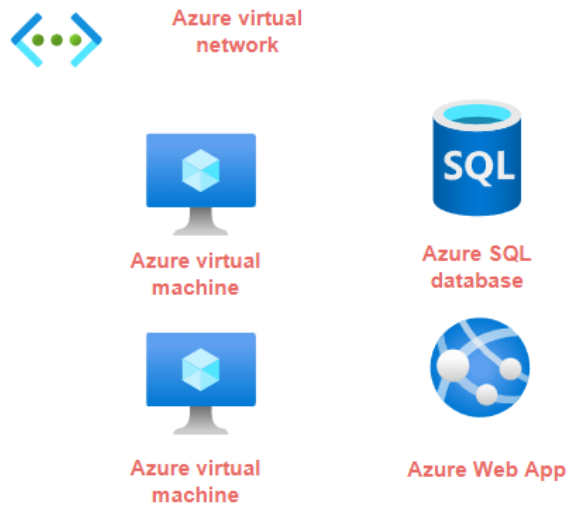
Terraform

Remember you can also create scripts to manage your Azure resources

Azure CLI

Azure PowerShell
modules

About ARM templates



You define your infrastructure 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 released 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.

ALAN RODRIGUES

template



Here the definition of the nested resource is defined within 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

ALAN RODRIGUES

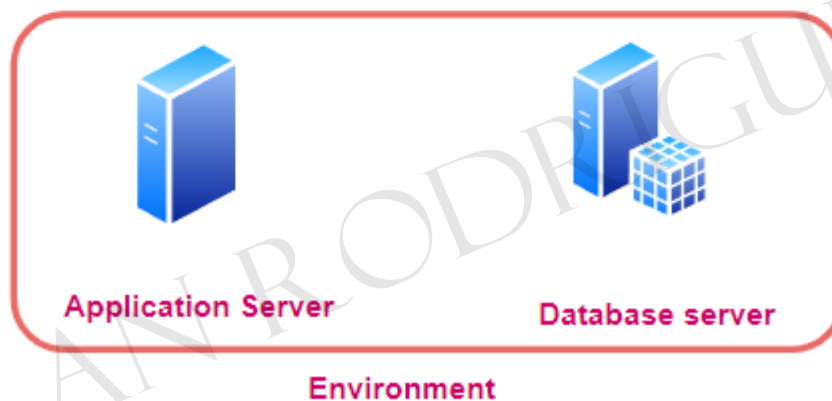
2. Terraform plan

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



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



Desired State Configuration

Desired State Configuration



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

Azure virtual 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.

ALAN RODRIGUES

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 Load Balancer



Public IP Address



Azure virtual machine scale set

Condition

Metrics



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.

ALAN RODRIGUES

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



Version 1

Version 2



Production Slot

Staging slot

Standard , Premium and
Isolated App Service Plan

Applications in
deployment slots have
their own host names

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

Canary deployments



Production Environments

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

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.



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.



Azure Traffic Manager Profile

Priority Routing Method



North Europe

Azure Web App



UK South

Azure Web App



Azure Traffic Manager Profile

Weighted Routing method



North Europe

Azure Web App

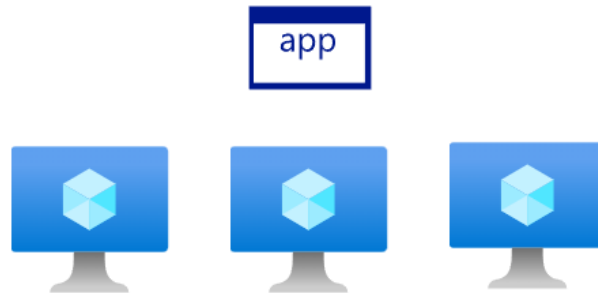


UK South

Azure Web App

Using a rolling deployment

Rolling deployments



Azure virtual machines

Here the application is running
on a set of virtual machines

Application - Version 1



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



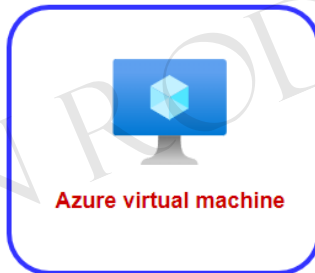
Azure Load Balancer

Health probe

Load Balancing Rules

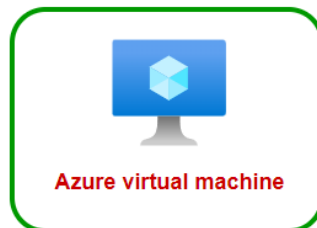
Azure virtual machines

Backend pool



Blue environment

All users are directed to the current production environment



Green environment

New application version

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



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

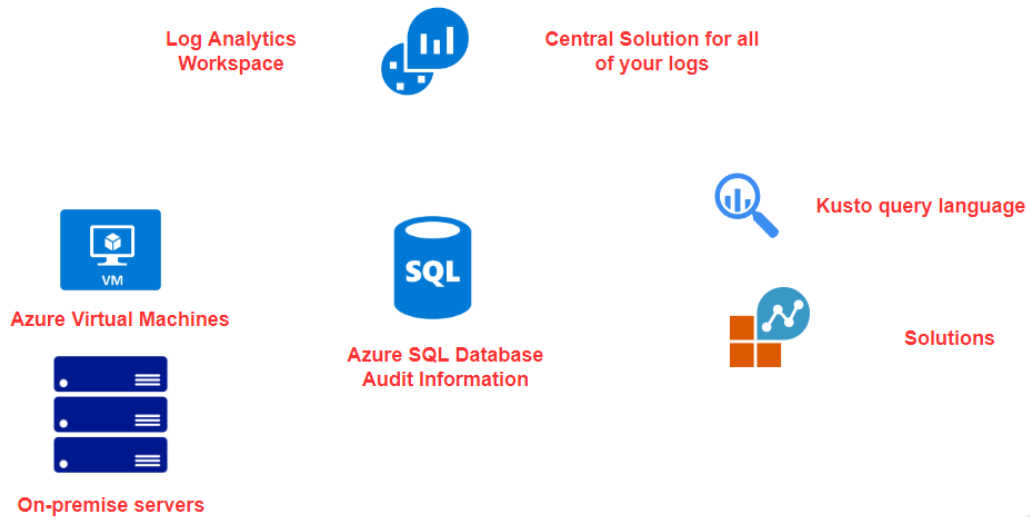


Azure Storage Account -
File share

The results reports will be available in the file share

Implement an instrumentation strategy

Log Analytics workspace



ALAN RODRIGUES