

**Azure DevOps HLD Document**

July 22, 2022

2Table of Contents

[1 Introduction 3](#_Toc77883639)

[1. Executive Summary 3](#_Toc77883640)

[2 Prerequisites 3](#_Toc77883641)

[2.1 DevOps Techstack 3](#_Toc77883642)

[2.2 Build and Deployment tools Integration (Azure DevOps with AWS) 4](#_Toc77883643)

[2.3 Build and Deployment tools Authentication and Authorization 4](#_Toc77883644)

[2.3.1 Azure Agent access using PAT 4](#_Toc77883645)

[2.3.2 Build Agent 4](#_Toc77883646)

[2.3.3 Deployment 4](#_Toc77883647)

[2.4 Azure DevOps Organization Configuration 5](#_Toc77883648)

[2.5 Project configurations 5](#_Toc77883649)

[2.6 Users management 6](#_Toc77883650)

[2.7 Notifications 6](#_Toc77883651)

[3 Source Code Management 6](#_Toc77883652)

[3.1 Azure Git Repository Configuration 6](#_Toc77883653)

[3.1.1 Azure Git Repository Branching Strategy 6](#_Toc77883654)

[4 Azure Pipeline Build and Deployment 8](#_Toc77883655)

[4.1 Azure Build and Deployment Solution Architecture 8](#_Toc77883656)

[5 Azure CI Pipeline 9](#_Toc77883657)

[6 Continuous Deployment strategy 9](#_Toc77883658)

[7 Configuration Management 9](#_Toc77883659)

[8 Promotion Strategy 10](#_Toc77883660)

[8.1 Azure Pipeline Promotion strategy design & workflow 10](#_Toc77883661)

[Approval groups 10](#_Toc77883662)

[9 Logs and Monitoring Management 10](#_Toc77883663)

[10 Database Deployments Strategy 11](#_Toc77883664)

# **Introduction**

# Executive Summary

Aspire Systems would like to thank the CPM team for the facilitation and cooperation extended to assess their environment.

The objective of this documents is to assess the current devops maturity, deployment efficiency and to provide betterment of agility, branching strategy, build tools and process and deployment strategy and significant improvements of Infrastructure governance and deployment applications management.

We have come up with this report based on the interactions with CPM team.

# **Prerequisites**

## **DevOps Techstack**

|  |  |  |
| --- | --- | --- |
| **Solution Element** | **Technology** | **Licensed/Open source** |
| **Source Code Version Control** | Azure DevOps - Git Repo | Licensed |
| **ALM** | Azure Board | Licensed |
| **Continuous Integration** | Azure DevOps – CI | Licensed as part of Azure DevOps |
| **Artifacts Storage** | Azure Artifacts | Licensed as part of Azure DevOps |
| **Code Quality** | SonarQube | Licensed |
| **Unit Testing** | XUnit | Open source |
| **Build Tools** | .NET 5 | Open source |
| **Docker Image Registry** | Azure Container Registry (ACR) | Azure PaaS |
| **Package Artifacts** | Azure Artifacts/Azure Blob Storage | Azure PaaS |
|  |  |  |
| **Configuration & Secret Management** | Azure Key Vault | Azure PaaS |
| Ansible |
| **Continuous Delivery Pipeline Tools** | Azure DevOps | Licensed |
| **API Gateway** | Azure API Gateway | Azure PaaS |
| **Application Monitoring** | Azure Monitoring | Azure PaaS |
| **Infra Monitoring** | Azure Monitoring,Insights | Azure PaaS |
| **Azure Account / Azure DevOps integration** | Azure Cloud, Azure Service Connection (PAT) | Open-source |

## Build and Deployment tools Integration (Azure DevOps with AWS)

AWS Toolkit for Microsoft Azure DevOps is an extension for Microsoft Azure DevOps. It contains tasks you can use in build and release definitions in Azure DevOps and Microsoft Azure DevOps Server to interact with AWS services. AWS Toolkit for Azure DevOps is available through the Visual Studio Marketplace.

AWS Toolkit can be installed through Azure DevOps extensions. once the AWS Toolkit is installed successfully, To allow the AWS Toolkit for Azure DevOps to access AWS services, you need an AWS account and AWS credentials.

## Build and Deployment tools Authentication and Authorization

### Azure Agent access using PAT

A personal access token (PAT) is used as an alternate passw**o**rd to authenticate into Azure DevOps.

To create PAT go to Users settings and select Personal Access Tokens and then select 'new Tokens", provide necessary details to create tokens. Once your PAT is created, you can use it anywhere your user credentials are required for authentication in Azure DevOps.

### Build Agent

Out of the box Azure DevOps provides free hosted agents that have a predefined set of tools installed and configured for building and deploying your apps.

To build your code or deploy your software using Azure Pipelines, you need at least one agent. As you add more code and people, you'll eventually need more.

When your pipeline runs, the system begins one or more jobs. An agent is a computing infrastructure with installed agent software that runs one job at a time.

**Self-hosted agents**

There are several options available to you to be able to run self-hosted agents. You can quite simply download the agent package, and run it on your local machine. There are packages available for Windows, Linux, and macOS.

An agent that you set up and manage on your own to run jobs is a self-hosted agent, can use self-hosted agents in Azure Pipelines. Self-hosted agents give you more control to install dependent software needed for your builds and deployments. Also, machine-level caches and configuration persist from run to run, which can boost speed, you can install the agent on Linux, macOS, or Windows machines. You can also install an agent on a Docker container.

Build agents running on Azure instances, the tasks can automatically obtain credential and Region information from instance metadata associated with the Azure instance.

### Deployment

* 1. Naming conventions
  2. Available Environments

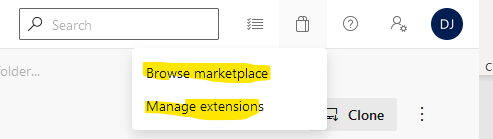
In line with the current setup, we propose to have 4 different environments,

* Development
* Staging
* Production

Dev, Staging and production environments will have dedicated environments that are always running. **Azure** **Boards**

## Azure DevOps Organization Configuration

An organization in Azure DevOps is a mechanism for organizing and connecting groups of related projects. Examples include business divisions, regional divisions, or other enterprise structures. You can choose one organization for your entire company, one organization just for you, or separate organizations for specific business units.

**

Your organization includes five free users with Basic access, plus unlimited Visual Studio subscribers and Stakeholders at no extra charge. Your organization also includes free monthly amounts of additional services such as build and deployment. The following tasks can be obtained in the Azure

DevOps organization.

* Collaborate with others to develop applications by using our cloud service
* Plan and track your work and code defects and issues
* Set up continuous integration and deployment
* Integrate with other services by using service hooks
* Obtain additional features and extensions
* Create one or more projects to segment work.

## Project configurations

Azure DevOps project contains the following set of features:

* Boards and backlogs for agile planning
* Pipelines for continuous integration and deployment
* Repos for version control and management of source code and artifacts
* Continuous test integration throughout the project life cycle Each organization contains one or more projects

The Project board consists of Iterations and Areas, each project can have only one iteration and areas, but multiple sprints and nodes can add based on the iterations.

Within organizations can have multiple projects and each project contains many repos and teams. To manage more granular security between your software projects and their teams, consider using many projects. At the highest level of isolation is an organization, where each organization is connected to a single Azure AD tenant. A single Azure AD tenant, however, can be connected to many Azure DevOps organizations.

## Users management

* Teams:
* Multiple teams can be associated with each project. Groups or Users level permissions would be managed through permissions sections.
  + 1. Developer Group
    2. BI Group
    3. Project Management Users Group
    4. Testing Group
    5. DevOps and IT admin Group

## Notifications

* team or group can receive email notifications as changes occur to the following items in Azure DevOps:
* work items
* code reviews
* pull requests
* source control files
* builds
* releases

# **Source Code Management**

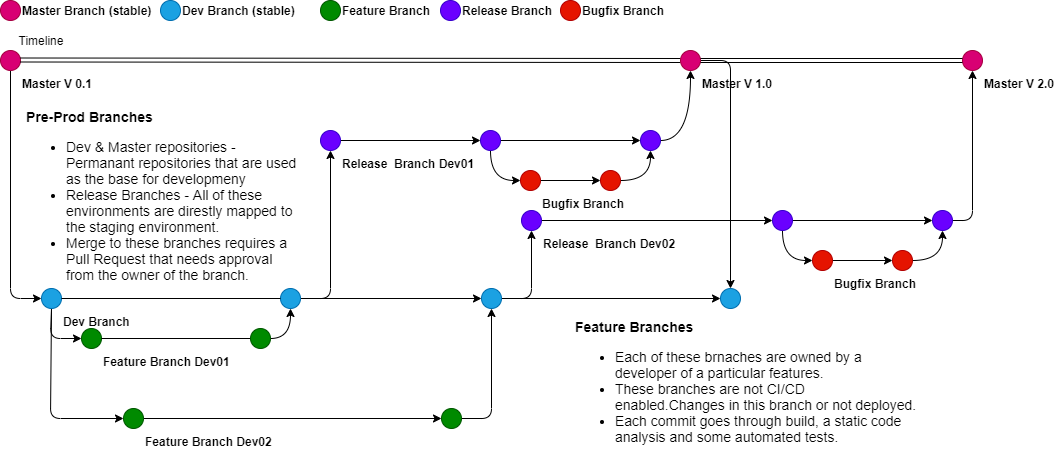
## Azure Git Repository Configuration

### Azure Git Repository Branching Strategy

The source code is maintained in an Azure DevOps GIT Repository. The branching strategy, as explained in the below picture, uses 2 main branches (Dev and Master). Apart from these main branches, every feature is created as a new branch. And there are branches created for each release.

Some important points to note are below:

* On these main branches, very minimal testing is done. Testing is done mostly on the feature branches.
* If a feature spans more than 1 service, a branch with the same name is created for both the services.



* **Permanent Branches**

Master and Dev branches are the stable branches. Code from feature branches will be merged on to Dev branch first. From Dev-Branch, Release branches are created that are eventually deployed to staging and production.

* **Feature branch creation**

Feature branches are created from a portal to standardize the naming of the branches.

* **Release Branch**

Each release branch is related to planned releases. Release Branches are created on the date of the planned release in the same name as the previously decided release number. These release branches are directly pushed to staging first. After sanity testing, the application is then pushed to production.

* **Hot Fix Branches**

Hot Fix branches are directly taken from Master and are deployed to Staging and Production eventually. These branches are treated like Release Branches.

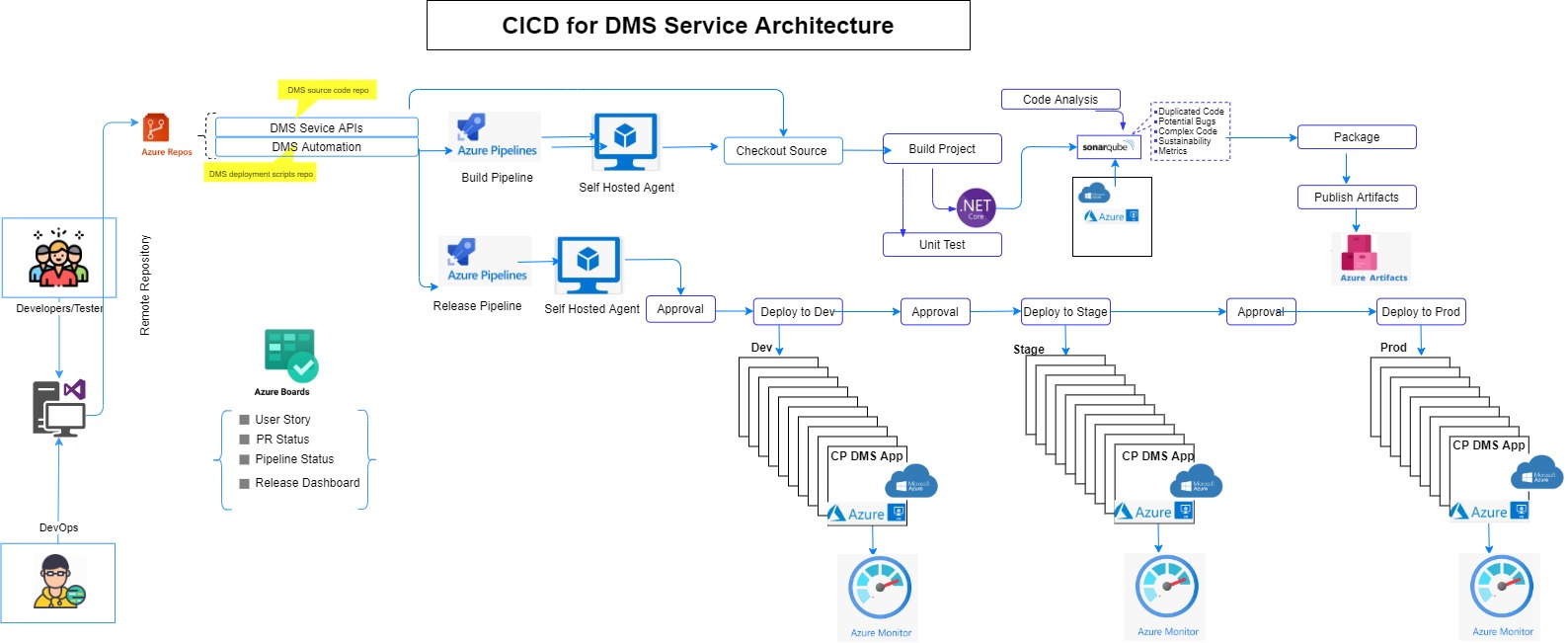
* 1. Pipeline Workflow Design
  2. Azure Git Repository Pull Request and Approval
  3. Azure Git Repository Merging and Release Strategy
* **Branching and merging principles:**

Each branch must have a defined policy about how to integrate code into this branch. For example, in the branch structure of the previous illustration, you can assign a team member to own and manage the MAIN branch. This member is responsible for performing the initial branch operation, reverse integrating changes from the DEVELOPMENT branch to the MAIN branch, and forward integrating changes from the MAIN branch to the DEVELOPMENT branch. Forward integration is important when the MAIN branch also integrates changes from other branches.

The MAIN branch must contain code that has passed integration tests so that it is always ready for a release.

The DEVELOPMENT (or work) branch constantly evolves because team members' check-in changes periodically.

# **Azure Pipeline Build and Deployment**



## Azure Build and Deployment Solution Architecture

* Azure DevOps CI/CD pipeline implementation is the backbone of the modern DevOps environment. It bridges the gap between development and operations teams by automating the building, testing, and deployment of applications.
* We have chosen Azure DevOps Pipeline as a Continuous Integration tool that governs the entire build and deployments automation. The planned build flow and prod deployment flow for each environment is explained in the below picture

**The cicd pipeline consists of the following steps:**

* A developer commits the code changes to the Azure DevOps git source repository
* Code change triggers a build pipeline in Azure DevOps. note: based on the process build trigger by manual or schedule time.
* build pipeline performs a code analysis in Sonnar Qube and test results will be shared by teams channel/email.
* based on the threshold limit the test will be passed and proceed to build.
* Azure build pipeline builds the microservice application then docker builds the image packages deploy to ACR.
* ACR image scanning falls under static image scanning which performs prior to deployments with the implication that developers can detect vulnerabilities before a container is launched.
* once the build is successful an alert notification sends to the MS teams channel along with the users' approval.

**Release pipeline consists of the following steps:**

* When the user approves the request, the Azure Release pipeline pull the latest artifacts images to the Azure deployment agent and execute deployment package.
* Azure release builds execute the deployment package to Azure VM’s.
* DMS service APIs will be deployed on a Windows Azure VM’s in Dev, Stage & Prod Env’s.
* AI/ML service API will be hosted in a separate Windows Azure VM’s in Dev, Stage & Prod
* Cosmos DB (NoSQL) will be used as a backend to the AI/ML APIs.
* Redis cache is used it to create cloud deployments that handle millions of requests per second at sub-millisecond latency all with the configuration, security and availability benefits of a managed service.
* Microsoft Azure Service Bus is used with message queues and publish-subscribe topics it is fully managed enterprise message broker .
* Azure Pipelines enable Continuous Delivery (CD) to Azure VM’s in Dev , Stage and Production through a secure service connection.
* Azure Key Vault is used to securely inject secrets and credentials into an application at runtime, abstracting sensitive information away from developers.

# **Continuous Deployment strategy**

* 1. Deployment Agent Integration
  2. Deployment pipeline configurations
  3. Deployment Artifacts
  4. Environment specific Configurations
  5. Non-Prod deployment strategy
  6. Production deployment strategy
  7. Notifications
* DMS Front-end will be hosted in an Azure Static web app with storage blob as content storage. Private Link will be configured for the DMS front end. All the external/client applications will access the private endpoint only
* AKS cluster will be created with 2 agent pools. Each agent pool will have 1 or worker nodes based on the bandwidth required
* AKS cluster & Agent pools will be created in a same resource group
* DMS service APIs will be deployed on a Linux worker nodes (e.g. agent pool name - coreapi)
* AI/ML service API will be hosted in a separate Linux worker nodes (e.g. agent pool name - aimlapi)
* AI/ML worker node spec will be bit higher than coreapi agent pool since it might require more compute resource
* Cosmos DB (NoSQL) will be used as a backend to the AI/ML APIs. This will be created in a same VNET as the AKS cluster
* Private link will be configured for the Cosmos DB. Cluster resource should be able to access the DB through private endpoint

# **Configuration Management**

Variables and secrets related to the various environments will be managed on Azure.

* Azure Key Vault – to manage secrets, passwords, and other confidential info.

# **Promotion Strategy**

## Azure Pipeline Promotion strategy design & workflow

## Approval groups

We need the following approval groups. Each group will own a Teams Channel and anyone in the group can approve or reject. All approval requests will be sent to the corresponding “Teams” Channel.

1. Dev Deployment Approvers
   * Dev Leads will approve the deployment of new code to any existing Dev environments
2. QA Deployment Approvers
   * Testing Leads will approve the deployment of new release image id for QA environments
3. Staging Deployment Approvers
   * Project Lead will approve the deployment of new code to staging environments
   * Testing Lead will approve the change of the new release image id for the Staging Environment
4. Product Deployment Approvers
   * Project Lead will approve the change of the new release image id for the Prod Environment
   * Azure Pipeline promotion and approval

Approvals are required for the below actions

1. Provisioning new environment
2. Trigger of deployment to any environment.

# **Logs and Monitoring Management**

* 1. Azure Monitor & Insights

# **Database Deployments Strategy**